BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Sistem adalah suatu jaringan kerja dari prosedur - prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan keigatan atau untuk melakukan sasaran tertentu (J. Hutahaean, 2015 : 1).

II.1.1. Karakteristik Sistem

Supaya sistem itu dikatakan sistem yang baik harus memiliki karakteristik yaitu :

1. Komponen

Suatu sistem terdiri dari sejumlah komponen-komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk suatu kesatuan. Komponen sistem terdiri dari komponen berupa subsistem atau bagian-bagian dari sistem.

2. Batasan sistem (*boundary*)

Batasan sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lain atau dengan lingkungan luarnya. Batasan sistem ini memungkinkan suatu sistem dipandang sebagai suatu kesatuan. Batasan suatu sistem menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

3. Lingkungan luar sistem (*environment*)

Lingkungan luar sistem (*environtment*) adalah diluar batas sistem yang mempengaruhi operasi sistem. Lingkungan dapat bersifat menguntungkan yang

harus tetap dijaga dan yang merugikan yang haris dijaga dan dikendalikan, kalau tidak akan mengganggu kelangsungan hidup dari sistem.

4. Penghubung sistem (*interface*)

Penghubung sistem merupakan media penghubung antara satu subsistem dengan subsistem lainnya. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari subsistem ke subsistem lain. Keluaran (*output*) dari subsistem akan menjadi masukan (*input*) untuk subsistem lain melalui penghubung.

5. Masukan sistem (*input*)

Masukan adalah energi yang dimasukkan kedalam sistem, yang dapat berupa perawatan (*maintenance input*), dan masukan sinyal (*signal input*). *Maintenance input* adalah energi yang dimasukkan agar sistem dapat beroperasi. *Signal input* adalah energi yang diproses untuk mendapatkan keluaran. Contoh dalam sistem komputer, program adalah *maintenance input* sedangkan data adalah *signal input*.

6. Keluaran sistem (*output*)

Keluaran sistem adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan. Contoh komputer menghasilkan panas yang merupakan sisa pembuangan, sedangkan informasi adalah keluaran yang dibutuhkan (J. Hutahaean, 2015 : 3-4).

II.2. Sistem Pakar

Sistem pakar (*expert system*) adalah sistem yang berusaha mengadopsi pengetahuan manusia ke komputer, agar dapat menyelesaikan masalah kedokteran

seperti yang biasa dilakukan oleh para ahli. Tujuan pengembangan sistem pakar sebenarnya bukan untuk menggantikan peran manusia, tetapi untuk mensubstitusikan pengetahuan manusia ke dalam bentuk sistem, sehinggan dapat digunakan oleh orang banyak. Sistem pakar disusun oleh dua bagian utama, yaitu lingkungan pengembangan (*development environment*) dan lingkungan konsultasi (*consultation environment*). Lingkungan pengenmbangan sistem pakar digunakan untuk memasukkan pengetahuan pakar ke dalam lingkungan sistem pakar, guna memperoleh pengetahuan pakar. Komponen dalam sistem pakar yaitu *user interface* (antarmuka pengguna), basis pengetahuan, akuisisi pengetahuan, mesin inferensi, *workplace*, fasilitas penjelasan dan perbaikan pengetahuan.

1. *User interface* (antarmuka pengguna)

   User interface merupakan mekanisme yang digunakan oleh pengguna dan sistem pakar untuk berkomunikasi.

2. Basis pengetahuan

   Basis pengetahuan mengandung pengetahuan untuk pemahaman, formulasi dan penyelesaian masalah.

3. Akuisisi pengetahuan

   Akuisisi pengetahuan adalah akumulasi, transfer dan transformasi keahlian dalam menyelesaikan masalah dari sumber pengetahuan ke dalam program komputer.

4. Mesin inferensi

   Mesin inferensi adalah program komputer yang memberikan metodologi untuk penalaran tentang informasi yang ada dalam basis pengetahuan dan dalam *workplace* dan untuk memformulasikan kesimpulan.

5. *Workplace*

   Workplace merupakan area dari sekumpulan memori kerja (*working memory*).

6. Fasilitas

   Penjelasan fasilitas adalah komponen tambahan yang akan meningkatkan kemampuan sistem pakar.

7. Perbaikan pengetahuan

   Pakar memiliki kemampuan untuk menganalisis dan meningkatkan kinerjanya serta kemampuan untuk belajar dari kinerjanya (Rachmawatii, D.J. Damiri, A. Susanto; 2012 : 2-3).

II.2.1. Manfaat Sistem Pakar

Sistem pakar menjadi sangat popular karena sangat banyak kemampuan dan manfaat yang diberikannya, diantaranya :

1. Meningkatkan produktivitas, karena sistem pakar dapat bekerja lebih cepat dari manusia.
2. Membuat seseorang yang awam bekerja seperti layaknya seorang pakar.
3. Meningkatkan kualitas, dengan memberi nasehat yang konsisten dan mengurangi kesalahan.
4. Mampu menangkap pengetahuan dan kepakaran seseorang.

5. Memudahkan akses pengetahuan seorang pakar.
6. Bisa digunakan sebagai media pelengkap dalam pelatihan
7. Meningkatkan kemampuan untuk menyelesaikan masalah karena sistem pakar mengambil sumber pengetahuan dari banyak pakar (Hayadi B. Herawan, 2016 : 2-3).

II.2.2. Kekurangan Sistem Pakar

Selain manfaat, ada juga beberapa kekurangan yang ada pada sistem pakar, diantaranya :

1. Biaya yang sangat mahal untuk membuat dan memeliharanya.
2. Sulit dikembangkan karena keterbatasan keahlian dan ketersediaan pakar.
3. Sistem pakar tidak 100% bernilai benar (Hayadi B. Herawan; 2016 : 3).

II.2.3. Ciri – Ciri Sistem Pakar

Ciri – ciri sistem pakar adalah sebagai berikut :

1. Terbatas pada domain keahlian tertentu.
2. Dapat memberikan penalaran untuk data yang tidak pasti.
3. Dapat mengemukakan rangkaian alasan yang diberikannya dengan cara yang dapat dipahami.
4. Berdasarkan pada kaidah atau rule tertentu.
5. Dirancang untuk dapat dikembangkan secara bertahap.
6. Pengetahuan dan mekanisme inferensi jelas terpisah.
7. Keluarannya bersifat anjuran.
8. Sistem dapat mengaktifkan kaidah secara searah yang sesuai dituntun oleh dialog dengan pemakai (Hayadi B. Herawan, 2016 : 3- 4).

II.3. Penyakit Asam Lambung

Dalam sains konvensional, maag tergolong penyakit yang tidak bisa disembuhkan secara total. Penyakit maag atau tukak lambung merupakan suatu penyakit psikosomatis (penyakit pikiran tubuh) atau bisa juga penyakit infeksi yang disebabkan oleh bakteri gram negatif helicobacter pylori, efek samping obat-obatan dan pola makan yang salah (Danton Awan; 2016 : 3).

II.4. Certainty Factor

Faktor kepastian (*certainty factor*) diperkenalkan oleh Shortliffe Buchanan dalam pembuatan MYCIN pada tahun 1975 untuk mengakomodasi ketidakpastian pemikiran (*inexact reasoning*) seorang pakar. Teori ini berkembang bersamaan dengan pembuatan sistem pakar MYCIN. Tim pengembang MYCIN mencatat bahwa dokter sering kali menganalisa informasi yang ada dengan ungkapan seperti misalnya : mungkin, kemungkinan besar, hampir pasti. Untuk mengakomodasi hal ini, tim MYCIN menggunakan *certainty factor* (CF) guna menggambarkan tingkat keyakinan pakar terhadap permasalahan yang sedang terjadi. Secara umum, rule direpresentasikan dalam bentuk sebagai berikut :

IF E1 [AND / OR] E2 [AND / OR] ... En..... (1)

THEN H (CF = Cfi)..... (2)

Dimana :

E1 .... En : fakta – fakta (*evidence*) yang ada.

H : hipotesa atau konklusi yang dihasilkan.

CF : tingkat keyakinan (*certainty factor*) terjadinya hipotesa H akibat adanya fakta – fakta E1 s/d En.

Definisi menurut David McAllister, *certainty factor* adalah suatu metode untuk membuktikan apakah suatu fakta itu pasti ataukah tidak pasti yang berbentuk metrik yang biasanya digunakan dalam sistem pakar (Weni Wilda; 2013 : 104).

II.4.1. Kelebihan dan Kekurangan Certainty Factor

Kelebihan dari metode *certainty factor* adalah :

1. Metode ini cocok dipakai dalam sistem pakar untuk mengukur sesuatu apakah pasti atau tidak pasti dalam mendiagnosis dan mengidentifikasi hama atau penyakit sebagai salah satu contohnya.
2. Perhitungan dengan metode ini dalam sekali proses perhitungan hanya dapat mengolah 2 data saja sehingga keakuratan data dapat terjaga.

Adapun kekurangan metode *certainty factor* adalah :

1. Ide umum dari pemodelan ketidakpastian manusia dengan menggunakan *numeric certainty factor* biasanya diperdebatkan. Sebagian orang akan membantah pendapat bahwa formula untuk metode *certainty factor* diatas memiliki sedikit kebenaran.
2. Metode ini dapat mengolah ketidakpastian / kepastian hanya dua data saja perlu dilakukan beberapa kali pengolahan data untuk data yang lebih dari dua buah (Dodi Harto; 2013 : 24).

II.5. Visual Basic 2010

Visual basic 2010 adalah bahasa pemrograman yang memungkinkan pengembang untuk dengan mudah membangun aplikasi *windows* yang kompleks dan aplikasi web, serta perangkat lunak lainnya. Visual basic 2010 didasarkan

pada bahasa pemrograman visual basic yang dikembangkan microsoft pada awal 1990 an. Visual basic, pada awalnya, adalah bahasa BASIC (*Beginner's All Purpose Symboic Instructional Code*) yang dikembangkan pada 1960 an.

Popularitas visual basic berevolusi dari berbagai fitur produktivitas yang memungkinkan pengembang dengan cepat menghasilkan aplikasi perangkat lunak berkualitas tinggi untuk *windows*. Hari ini, visual basic adalah bahasa pemrograman yang paling banyak digunakan di dunia karena seperti bahasa inggris dan dianggap sebagai salah satu bahasa pemrograman *enterprise* yang saling mudah untuk dipelajari. Visual basic adalah satu-satunya bahasa di visual studio yang tidak bersifat *case sensitive*, yang membuatnya mudah bagi *programmer* pemula. Bahasa ini sangat bagus dibandingkan bahasa lain pemrograman di visual studio seperti C++ atau C# (Shelly, Hoishington; 2011 : 19).

II.6. SQL Server 2008

SQL Server 2008 adalah sebuah terobosan baru dari microsoft dalam bidang database. SQL Server adalah 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.

Microsoft merilis SQL Server 2008 dalam beberapa versi yang disesuaikan dengan segmen – segmen pasar yang dituju. Versi – versi tersebut

adalah sebagai berikut. Menurut cara pemrosesan data pada prosesor maka microsoft mengelompokkan produk ini berdasarkan 2 jenis yaitu :

1. Versi 32-bit (x86), yang biasanya digunakan untuk komputer dengan single prosesor (pentium 4) atau lebih tepatnya prosesor 32 bit dan sistem operasi windows XP.
2. Versi 64-bit (x64), yang biasanya digunakan untuk komputer dengan lebih dari satu prosesor (misalnya core 2 duo) dan sistem operasi 64 bit seperti windows xp 64, vista dan windows 7.

Sedangkan secara keseluruhan terdapat versi – versi seperti berikut ini :

1. Versi *compact*, ini adalah versi "tipis" dari semua versi yang ada. Versi ini seperti versi *desktop* pada SQL Server 2000. Versi ini juga digunakan pada *handled drive* seperti Pocket PC, PDA, Smartphone, Tablet PC.
2. Versi *Express*, ini adalah versi "ringan" dari semua versi yang ada (tetapi versi ini berbeda dengan versi *compact*) dan paling cocok untuk latihan para pengembang aplikasi. Versi ini membuat *Express Manager Standard*, integrasi dengan CLR dan XML (Wenny Widya; 2013 : 3-4).

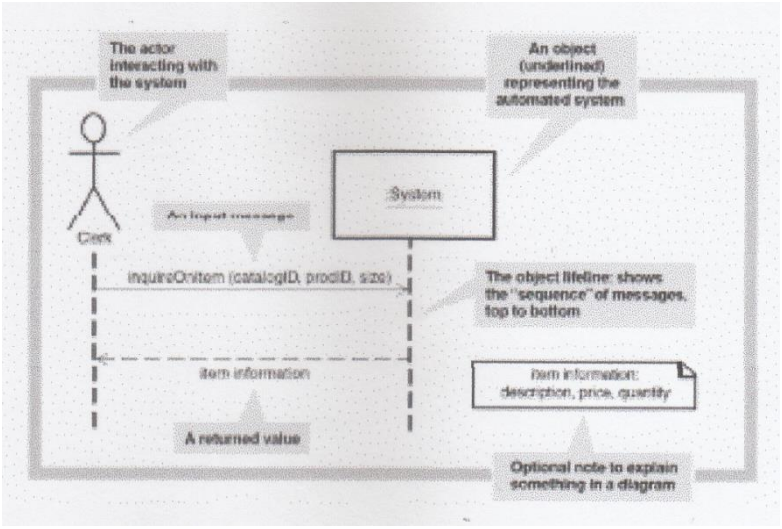
II.7. UML (*Unified Modelling Language*)

Berikut adalah diagram-diagram unified modelling language (UML) yang digunakan penulis yaitu :

1. Sequence Diagram

System Sequence Diagram (SSD) adalah diagram yang digunakan untuk mendefinisikan *input* dan *output* serta urutan instruksi antara pengguna dan sistem untuk sebuah *use case*.

a. Aktor : mewakili seorang aktor (orang atau peran yang berinteraksi dengan sistem).
b. Kotak berlabel: Sistem adalah objek yang mewakili keseluruhan sistem yang terotomatisasi.
c. Garis putus – putus vertikal (*lifelines*) adalah perpanjangan objek tersebut, baik aktor maupun objek, sepanjang durasi dari *sequence diagram*.
d. Anak panah antara *lifeline* mewakili *message* yang dikirim atau diterima oleh aktor dari sistem.
e. *Message* diberi label untuk menggambarkan maksud *message* dan *input* apapun yang sedang dikirim. *Message* dipertimbangkan sebagai sebuah aksi yang diminta pada tujuan objek, kebanyakan seperti perintah.



Gambar II.1. Notasi *Sequence* Diagram

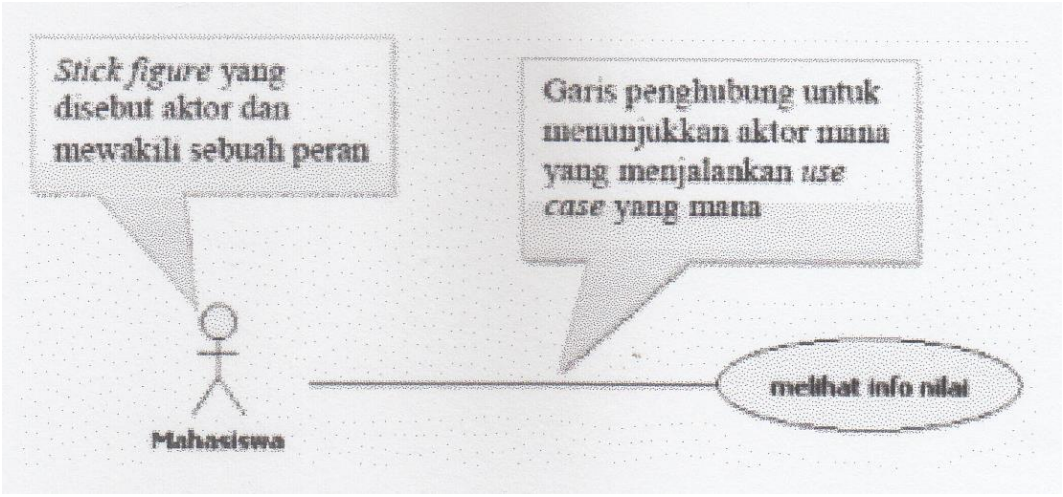
Sumber : E. Triandini & G Suardika; 2012 : 71

2. *Use Case Diagram*

Use case adalah sebuah kegiatan yang dilakukan oleh sistem, biasanya dalam menanggapi permintaan dari pengguna sistem. Salah satu langkah awal untuk membuat diagram *use case* adalah dengan mengidentifikasi aktor dan proses bisnis dasar.

Langkah-langkah membuat use case :

a. Mengidentifikasi aktor. Perhatikan bahwa aktor sebenarnya adalah peran yang dimainkan oleh pengguna. Alih-alih menyusun daftar aktor sebagai bob, maria atau tuan hendricks, sebaiknya identifikasi peran spesifik yang dimainkan oleh orang – orang tersebut. Ingatlah bahwa orang yang sama mungkin memainkan berbagai peran karena ia menggunakan sistem. Sistem lain juga dapat menjadi aktor dari sistem. Contoh aktor : mahasiswa, dosen, *order clerk*, *department manager*, auditor dsb.
b. Setelah peran aktor teridentifikasi, langkah berikutnya adalah menyusun tujuan – tujuan yang ingin dicapai oleh peran – peran tersebut dalam penggunaan sistem. Tujuan tersebut merupakan tugas yang dilakukan oleh aktor untuk mencapai beberapa fungsi bisnis yang memberikan nilai tambah bagi bisnis. Contoh : melihat infor biodata, menyimpan data login, mengirim testimoni.



Gambar II.2. Notasi *Use Case* Diagram

Sumber : E. Triandini & G Suardika; 2012 : 17

3. *Activity Diagram*

Activity Diagram adalah sebuah diagram alur kerja yang menjelaskan berbagai kegiatan pengguna (atau sistem), orang yang melakukan masing – masing aktivitas, dan aliran sekuensial dari aktivitas – aktivitas tersebut.

Notasi umum yang sering digunakan dalam *activity diagram* adalah sebagai berikut :

a. *Swimlane* : mewakili agen yang melakukan aktivitas. Karena dalam alur kerja umumnya mempunyai agen yang berbeda yang melakukan langkah yang berbeda dari proses alur kerja. Simbol *swimlane* membagi aktivitas alur kerja ke dalam kelompok yang menunjukkan agen mana yang menjalankan aktivitas yang mana. Ada dua jenis *swimlane* yang dapat digunakan sesuai dengan kebutuhan, yaitu *swimlane vertical* dan *swimlane horizontal*.
b. *InitialState* : awal dari alur kerja

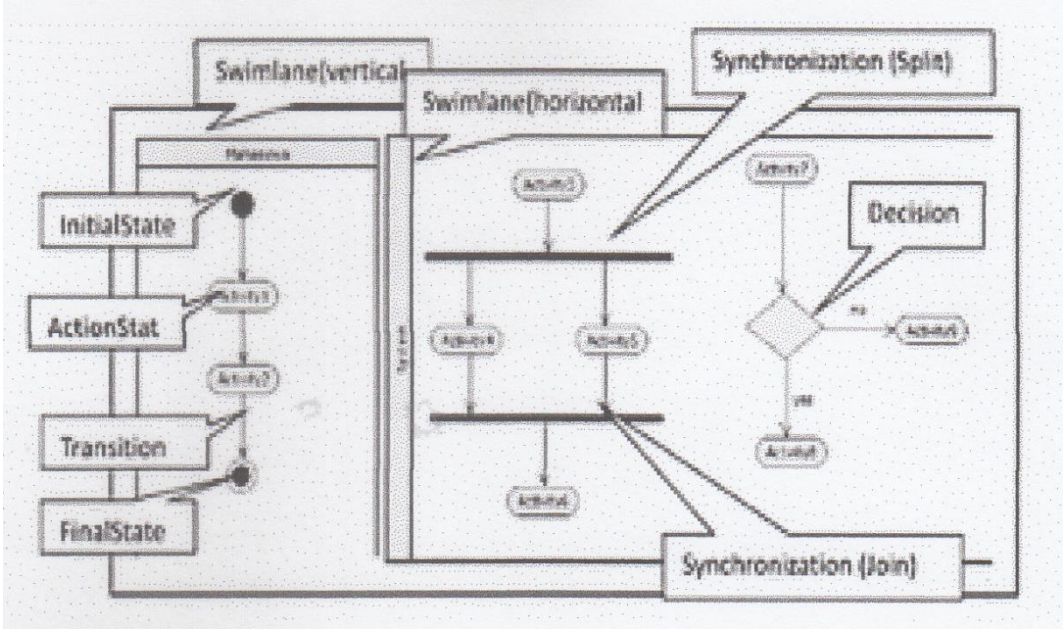
c. *ActionState* : melambangkan aktivitas tersendiri dalam alur kerja.

d. *Transition* : melambangkan urutan di antara aktivitas.

e. *FinalState* : akhir dari alur kerja.

f. *Synchronization* : membagi alur kerja menjadi beberapa alur yang berbarengan ataupun menggabungkan lagi alur yang berbarengan.

g. *Decision* : titik pengambilan keputusan dimana aliran proses tersebut akan mengikuti satu jalur atau jalur lainnya.



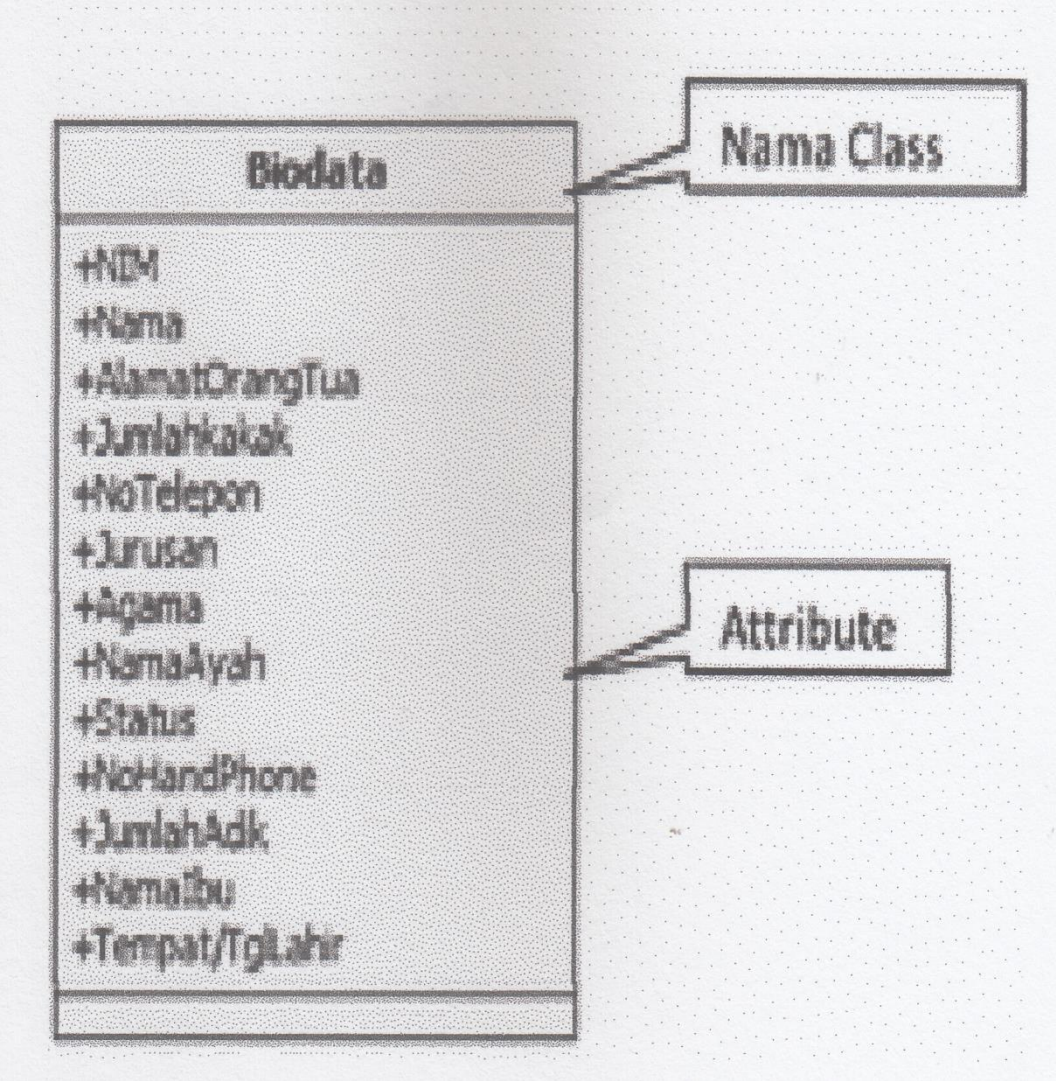
Gambar II.3. Notasi *Activity* Diagram

Sumber : E. Triandini & G Suardika; 2012 : 37

4. *Class Diagram*

Dalam UML, ada dua jenis *class diagram* yaitu : *domain class diagram* dan *design class diagram*. Tujuan utamanya adalah untuk mendokumentasikan dan menggambarkan kelas – kelas dalam pemrograman yang nantinya akan dibangun. *Design class diagram* menggambarkan kelas berorientasi objek yang

dibutuhkan dalam pemrograman, navigasi diantara kelas, *attribute names*, dan propertinya, serta *method name* dan propertinya.



Gambar II.4. Notasi *Class* Diagram

Sumber : E. Triandini & G Suardika; 2012 : 49