BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Kata sistem mempunyai beberapa pengertian, tergantung dari sudut pandang mana kata tersebut didefinisikan. Secara garis besar ada dua kelompok, yaitu : (Kusrini, dkk ; 2010 : 5).

1. Pendekatan sistem yang lebih menekankan pada elemen-elemen atau kelompoknya, yang dalam hal ini sistem itu didefinisikan sebagai suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu aturan tertentu.
2. Pendekatan sistem sebagai jaringan kerja dari prosedur, yang lebih menekankan urutan operasi di dalam sistem. Prosedur (procedure) didefinisikan oleh Richard F. Neushl sebagai urutan operasi kerja (tulis menulis), yang biasanya melibatkan beberapa orang di dalam satu atau lebih departemen, yang diterapkan untuk menjamin penanganan yang seragam dari transaksi bisnis yang terjadi.

Pendekatan sistem yang lebih menekankan pada elemen-elemen atau komponennya mendefinisikan sistem sebagai sekumpulan elemen yang saling terkait atau terpadu yang dimaksudkan untuk mencapai suatu tujuan. Dengan demikian di dalam suatu sistem, komponen-komponen ini tidak dapat berdiri sendiri-sendiri, tetapi sebaliknya, saling berhubungan hingga membentuk satu kesatuan sehingga tujuan sistem itu dapat tercapai.

II.1.1. Karakter Sistem

Sistem mempunyai beberapa karakteristik atau sifat-sifat tertentu, antara lain : (Kusrini, dkk ; 2010 : 6).

1. Komponen sistem (*component*)

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang saling bekerja sama membentuk suatu komponen sistem atau bagian-bagian dari sistem.

2. Batasan sistem (*boundary*)

   Merupakan daerah yang membatasi suatu sistem dengan sistem yang lain atau dengan lingkungan kerjanya.

3. Subsistem

   Bagian-bagian dari sistem yang beraktivitas dan berinteraksi satu sama lain untuk mencapai tujuan dengan sasarannya masing-masing.

4. Lingkungan Luar Sistem (*Environment*)

   Suatu sistem yang ada di luar dari batas sistem yang dipengaruhi oleh operasi sistem.

5. Penghubung Sistem (*interface*)

   Media penghubung antara suatu subsistem dengan subsistem lain. Adanya penghubung ini memungkinkan berbagai sumber daya mengalir dari suatu subsistem ke subsistem lainnya.

6. Masukan Sistem (*input*)

   Energi yang masuk ke dalam sistem, berupa perawatan dan sinyal. Masukan perawatan adalah energy yang dimasukan supaya sistem tersebut dapat berinteraksi.

7. Keluaran sistem (*output*)

Hasil energy yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan.

8. Pengolahan Sistem (*process*)

Suatu sistem dapat mempunyai suatu bagian pengolah yang akan mengubah masukan menjadi keluaran

9. Sasaran Sistem (*object*)

Tujuan yang ingin dicapai oleh sistem, akan dikatakan berhasil apabila mengenai sasaran atau tujuan. (Kusrini, dkk ; 2010 : 7).

II.1.2. Klasifikasi Sistem

Suatu sistem dapat diklasifikasikan menjadi seperti berikut : (Kusrini, dkk ; 2010 : 7).

1. Sistem abstrak dan sistem fisik.

Sistem abstrak adalah suatu sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik, sedangkan sistem fisik adalah sistem yang ada secara fisik.

2. Sistem alamiah dan sistem buatan manusia

Sistem alamiah adalah sistem yang terjadi melalui proses alam sedangkan sistem buatan adalah sistem yang dirancang oleh manusia.

3. Sistem tertentu dan sistem tak tentu

Sistem tertentu adalah suatu sistem yang operasinya dapat diprediksi secara tepat sedangkan sisten tak tertentu adalah sistem dengan perilaku ke depan yang tidak dapat diprediksi.

4. Sistem tertutup dan sistem terbuka

Sistem tertutup adalah sistem yang tidak terpengaruh oleh lingkungan luar atau otomatis, sedangkan sistem terbuka adalah sistem yang berhubungan dan terpengaruh oleh lingkungan luar.

II.2. Pengertian Informasi

Informasi adalah data yang sudah diolah menjadi sebuah bentuk yang berarti bagi pengguna, yang bermanfaat dalam pengambilan keputusan saat ini atau mendukung sumber informasi. Data belum memiliki nilai sedangkan informasi sudah memiliki nilai. Informasi dikatakan bernilai bila manfaatnya lebih besar disbanding biaya untuk mendapatkannya. (Kusrini, dkk ; 2010 : 7).

II.3. Kualitas Informasi

Informasi yang berkualitas memiliki 2 kriteria, yaitu : (Kusrini, dkk ; 2010 : 8).

1. Akurat (*accurate*)

Informasi harus bebas dari kesalahan, tidak bisa ataupun menyesatkan. Akurat juga berarti bahwa informasi itu harus dapat dengan jelas mencerminkan maksudnya.

2. Tepat pada waktunya (*timelines*)

Informasi yang datang pada penerima tidak boleh terlambat. Di dalam pengambilan keputusan, informasi yang sudah using tidak lagi bernilai. Bila informasi datang terlambat sehingag pengambilan keputusan terlambat dilakukan, hal ini dapat berakibat fatal bagi perusahaan.

3. Relevan (*relevance*)

Informasi yang disampaikan harus mempunyai keterkaitan dengan masalah yang akan dibahas dengan informasi tersebut. Informasi harus bermanfaat bagi pemakainya. Di samping karakteristik, nilai informasi juga ikut menentukan kualitasnya. Nilai informasi (value of information) ditentukan oleh dua hal, yaitu manfaat dan biaya untuk mendapatkannya. Suatu informasi dikatakan bernilai bila manfaatnya lebih besar disbanding biaya untuk mendapatkannya.

II.4. Sistem Pakar

Sistem pakar adalah aplikasi berbasis komputer yang digunakan untuk menyelesaikan masalah sebagaimana yang dipikirkan oleh pakar. Pakar yang dimaksud di sini adalah orang yang mempunyai keahlian khusus yang dapat menyelesaikan masalah yang tidak dapat diselesaikan oleh orang awam. Sebagai contoh, dokter adalah seorang pakar yang mampu mendiagnosis penyakit yang diderita pasien serta dapat memberikan penatalaksanaan terhadap penyakit tersebut. Tidak semua orang dapat mengambil keputusan mengenai diagnosis dan memberikan penatalaksanaan suatu penyakit. Contoh yang lain, montir adalah seorang yang punya keahlian dan pengalaman dalam menyelesaikan kerusakan mesin motor atau mobil. Psikolog adalah orang yang ahli dalam memahami kepribadian seseorang, dan lain-lain. (Kusrini ; 2010 ; 3).

Sistem pakar adalah suatu program komputer cerdas yang menggunakan *knowledge* (pengetahuan) dan prosedur inferensi untuk menyelesaikan masalah yang cukup sulit sehingga membutuhkan seorang yang ahli untuk menyelesaikannya. Pengetahuan adalah sebuah kekuatan yang dapat memecahkan

suatu masalah yang kita temui sehari-hari. Sistem pakar adalah program *Artificial Intellenge* yang menggabungkan pangkalan pengetahuan (*knowledge base*) dengan sistem inferensi. Kecerdasan buatan atau *Artificial Intellenge* (AI) dapat didefinisikan sebagai sub bidang pengetahuan komputer yang khusus ditujukan untuk membuat *software* dan *hardware* yang sepenuhnya biasa menirukan beberapa fungsi otak manusia. Karena itu diharapkan komputer bisa membantu manusia didalam berbagai masalah yang sangat rumit. (Wenny Widiastuti, dkk, jurnal.sttgarut.ac.id ; 2012 : 2).

Sistem pakar yang mencoba memecahkan masalah yang biasanya hanya bisa dipecahkan oleh seorang pakar, dipandang berhasil ketika mampu mengambil keputusan seperti yang dilakukan oleh pakar aslinya baik dari sisi proses pengambilan keputusannya maupun hasil keputusan yang diperoleh.

Sebuah sistem pakar memiliki 2 komponen utama yaitu basis pengetahuan dan mesin inferensi. Basis pengetahuan merupakan tempat penyimpanan pengetahuan dalam memori computer, di mana pengetahuan ini diambil dari pengetahuan pakar.

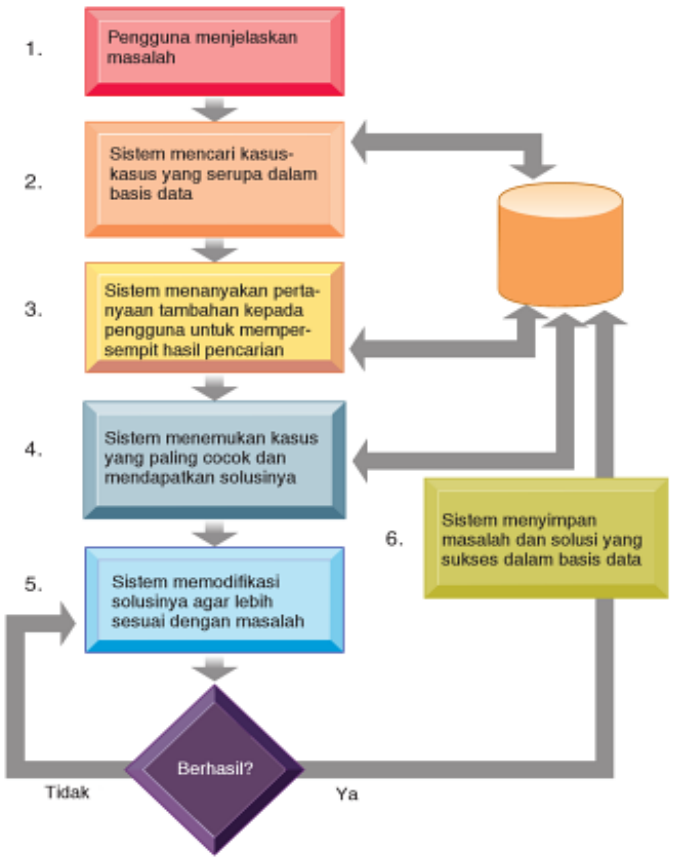
Ada banyak cara untuk merepresentasikan pengetahuan, di antaranya adalah logika (*logic*), jaringan semantic (*semantic nets*), *Object atribut value* (OAV), bingkai (*frame*), dan kaidah produksi (*production rule*).

Mesin inferensi merupakan otak dari aplikasi sistem pakar. Bagian inilah yang menuntun *user* untuk memasukkan fakta sehingga diperoleh suatu kesimpulan. Apa yang dilakukan oleh mesin inferensi ini didasarkan pada pengetahuan yang ada dalam basis pengetahuan.

II.5. Sistem Logika Fuzzy

Kebanyakan orang tidak berpikir dengan menggunakan aturan IF THEN yang tradisional atau dengan angka-angka yang tepat. Manusia cenderung mengategorikan hal-hal secara tidak tepat, dengan menggunakan aturan pengambilan keputusan yang mungkin memiliki banyak perbedaan pengertian. Sebagai contoh, seorang laki-laki atau perempuan dapat saja kuat atau cerdas. Sebuah perusahaan dapat digolongkan sebagai besar, sedang, atau kecil. Suatu temperature dapat digolongkan sebagai panas, sejuk, dingin, atau hangat. Kategori-kategori ini merepresentasikan suatu rentang nilai. (Kennec, dkk ; 2010 ; 124).

Logika fuzzy (*fuzzy logic*) adalah teknologi berbasis aturan yang dapat merepresentasikan ketidak presisian seperti yang telah disebutkan, dengan menciptakan aturan yang menggunakan nilai subjektif atau nilai yang mendekati. *Logika fuzzy* dapat menjelaskan fenomena atau proses tertentu secara linguistic, kemudian merepresentasikannya dalam sejumlah kecil aturan yang fleksibel. Organisasi dapat menggunakan *logika fuzzy* untuk menciptakan sistem piranti lunak yang menangkap pengetahuan tersirat yang mengandung *ambiguitas linguistic*. (Kennec, dkk ; 2010 ; 125).



Gambar II.1. Cara kerja penalaran berbasis kasus
(Sumber : Kennech C, dkk ; 2010 : 125)

Lihat bagaimana *logika fuzzy* merepresentasikan temperature yang bervariasi dalam aplikasi computer untuk mengendalikan temperature ruangan secara otomatis. Istilah-istilah atau dikenal sebagai fungsi keanggotaan didefinisikan secara tidak presisi, sejuk adalah antara 50 derajat dan 70 derajat (*Fahrenheit*), padahal temperature sejuk lebih tepatnya berkisar antara 60 dan 67 derajat. Ingatlah bahwa kategori sejuk bertumpang tindih dengan kategoi dingin atau normal. Untuk mengendalikan lingkungan ruangan menggunakan logika fuzzy, programmer juga perlu mengembangkan definisi ketidakpresisian yang serupa untuk kelembapan dan faktor-faktor lainnya, seperti misalnya angin dan temperature di luar ruangan.

Aturan-aturan tersebut mungkin mencakup satu pernyataan berikut jika temperature sejuk atau dingin dan kelembapan rendah, sementara angin di luar ruangan tinggi dan temperature luar ruangan rendah, naikkan panas dan kelembapan dalam ruangan. Komputer akan mengombinasikan pembacaan fungsi keanggotan dengan melakukan pembobotan dan, menggunakan semua aturannya, menaikkan atau menurunkan temperature dan kelembapan.

Logika fuzzy menyediakan solusi bagi masalah-masalah yang sulir dipecahkan hanya dengan menggunakan aturan IF THEN. Di jepang, sistem kereta bawah tanah sendai menggunakan control *logika fuzzy* untuk mengatur percepatan kereta dengan mulus sehingga para penumpang yang berdiri tidak perlu perpengangan saat kereta itu berjalan. Mitsubishi Heavy Industries di Tokyo telah berhasil mengurangi pemakaian listrik untuk penyejuk ruangan sampai 20 persen dengan mengimplementasikan program-program control *logika fuzzy*. Perangkat focus otomatis pada kamera hanya mungkin dibuat dengan menggunakan *logika fuzzy*. Pada contoh tersebut, *logika fuzzy* telah memungkinkan perubahan setahap demi setahap pada masukan menghasilkan perubahan yang mulus pada keluaran, dan bukan keluaran yang putus-putus. Hal tersebut membuat *logika fuzzy* sangat bermanfaat bagi konsumen aplikasi elektronik dan rekaya. (Kennec, dkk ; 2010 : 126).

Logika fuzzy juga dapat bermanfaat di bidang manajemen untuk pengambilan keputusan dan control organisasi. Sebuah perusahaan di Wall Street menciptakan sistem yang memilih perusahaan-perusahaan yang berpotensi untuk diakuisisi dengan menggunakan bahasa yang dimengerti oleh para pedagang saham. Sistem logika fuzzy juga telah dikembangkan untuk mendeteksi

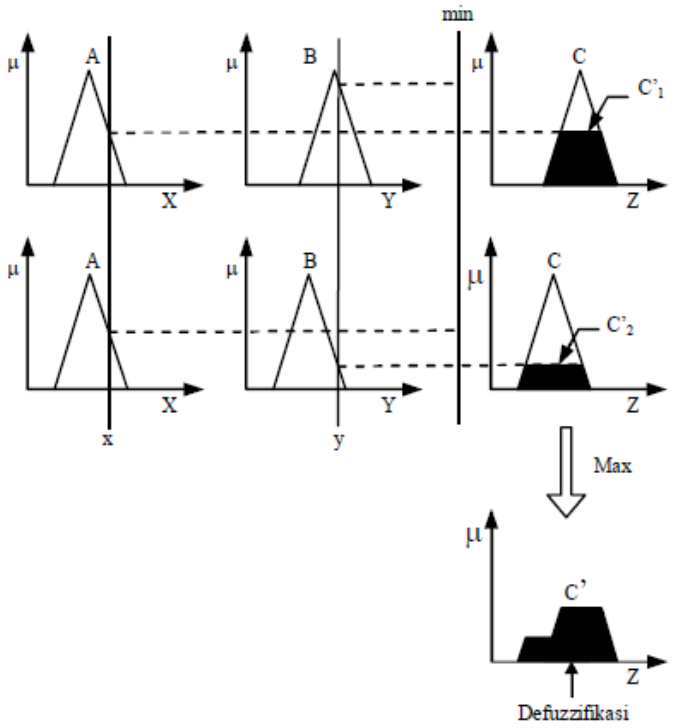
kemungkinan kebohongan dalam klaim kesehatan yang diajukan oleh penyedia perawatan kesehatan di Amerika Serikat. (Kennec, dkk ; 2010 : 127).

II.5.1. Logika Pengambilan Keputusan

Logika pengambilan keputusan atau dapat disebut penyimpulan fuzzy (*fuzzy inference*) mengaplikasikan aturan-aturan fuzzy pada masukan fuzzy kemudian mengevaluasi setiap aturan. Prinsip logika fuzzy digunakan untuk mengkombinasi aturan-aturan JIKA-MAKA (IFTHEN) yang terdapat dalam basis aturan kedalam suatu pemetaan dari suatu himpunan *fuzzy input* kesuatu himpunan *fuzzy output*. Logika pengambilan keputusan merupakan langkah kedua dalam pemrosesan logika fuzzy. Terdapat beberapa metode pengambilan keputusan dalam logika fuzzy diantaranya yaitu metode Mamdani. (Sutikno, eprints.undip.ac.id ; 2010 : 2, diakses tanggal 25-05-2014).

Fungsi implikasi yang digunakan pada pengambilan keputusan dengan metode Mamdani dengan menggunakan MIN dan dalam melakukan komposisi dengan menggunakan MAX. Metode komposisi ini sering disebut MAX-MIN. Contoh dalam penggunaan pengambilan keputusan dengan metode Mamdani ditunjukkan pada gambar II.2. Dengan memisalkan fungsi keanggotaan masukan dan keluaran menggunakan fungsi segitiga dan mempunyai 2 aturan fuzzy, yaitu:

a. IF Kesalahan adalah Nol dan Beda kesalahan adalah Positif maka Keluaran adalah Positif.
b. IF Kesalahan adalah Nol dan Beda kesalahan adalah Nol maka Keluaran adalah Nol.



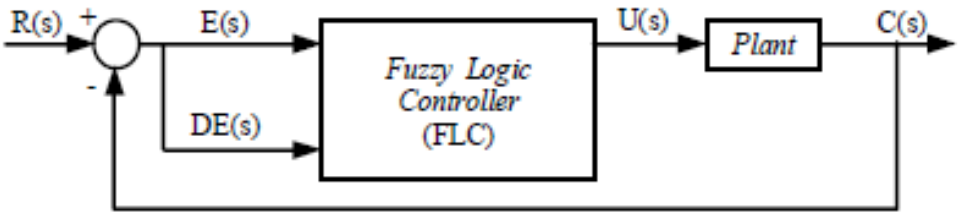
Gambar II.2. Proses pengambilan keputusan metode Mamdani
(Sumber : Sutikno ; 2010 : 2)

Langkah pertama pengambilan keputusan metode Mamdani adalah melakukan proses fuzzifikasi untuk memetakan data tegas masukan kesalahan dan beda kesalahan kedalam data fuzzy sesuai dengan tipe dan bentuk fungsi keanggotaan. Langkah kedua adalah melakukan proses terhadap kedua data fuzzy tersebut dengan operator AND yang akan mengambil nilai paling minimal dari dua data tersebut. Langkah ketiga dengan implikasi MIN akan memotong fungsi keanggotaan keluaran setelah melalui operator AND sehingga didapatkan daerah fuzzy. Ketiga proses tersebut juga diterapkan pada aturan-aturan fuzzy berikutnya. Setelah semua aturan fuzzy telah dieksekusi, dilakukan proses komposisi dengan metode MAX yaitu solusi himpunan fuzzy diperoleh dengan cara mengambil nilai maksimum aturan, kemudian menggunakannya untuk memodifikasi daerah fuzzy, dan mengaplikasikan ke *output* dengan menggunakan operator OR (*union*). Jika

proposisi telah dievaluasi, maka *output* akan berisi suatu himpunan fuzzy yang merefleksikan kontribusi dari tiap-tiap proposisi. Setelah proses implikasi dan komposisi telah dilakukan maka proses selanjutnya adalah proses defuzzifikasi. (Sutikno, eprints.undip.ac.id ; 2010 : 3, diakses tanggal 25-05-2014).

II.5.2. Kendali Logika Fuzzy

Dalam pendekatan kendali berbasis logika fuzzy, masukan, keluaran, dan tanggapan kendali dispesifikasikan sesuai dengan keahlian seorang pakar serta pemodelan matematika terhadap suatu sistem kendali tidak dibutuhkan. Prinsip dalam mendesain kendali logika fuzzy adalah mengaturatur parameter fungsi keanggotaan dan kaidah atur fuzzy. (Sutikno, eprints.undip.ac.id ; 2010 : 3, diakses tanggal 25-05-2014).



Gambar II.3. Kendali logika fuzzy pada sistem kalang tertutup (Sumber : Sutikno ; 2010 : 3)

Sebuah pengendali logika fuzzy umumnya dapat dimasukkan dalam sistem pengendali kalang tertutup. Pada Gambar 5 memperlihatkan system kendali kalang tertutup dengan pengendali logika fuzzy dimana E (*error*) dan dE (*delta error*/ perubahan *error*) merupakan masukan pengendali logika fuzzy dan U adalah besaran yang diberikan pada *plant*. Masukan *error* didapatkan dari nilai

referensi dikurangi dengan nilai keluaran dari *plant* yang dinotasikan oleh persamaan 5.

e(k) = r(k) – c(k) …… (5)

dimana:

e(k) adalah besar nilai kesalahan diskrit.

r(k) adalah besar nilai referensi diskrit. c(k) adalah besar nilai keluaran *plant* diskrit. Nilai masukan beda kesalahan (perubahan *error* pada sistem kontinyu) didapatkan dari nilai kesalahan sekarang dikurangi dengan nilai kesalahan sebelumnya yang dinotasikan oleh persamaan 6.

dE(k) = e(k) – e(k-1) ……(6)

dimana:

dE(k) adalah besar nilai beda kesalahan diskrit.

e(k) adalah besar nilai kesalahan diskrit.

e(k-1) adalah besar nilai kesalahan diskrit sebelumnya. (Sutikno, eprints.undip.ac.id ; 2010 : 4, diakses tanggal 25-05-2014).

II.6. Prosesor dan Motherboard

Komponen pertama yang paling penting dari penyusun *hardware* komputer adalah prosesor. Prosesor ini fungsinya sebagai otak computer. Manusia jika tidak punya otak akan mati. Ini juga benar di computer, tanpa prosesor, computer tak bisa hidup. Jika computer dihidupkan, prosesor selalu aktif. Secara teknis, prosesor adalah sebuah prosesor mikro atau disebut mikroprosesor, artinya prosesor ukuran mikro. (Tim E-Media Solusindo ; 2010 : 69)

Semua jenis komputer, apakah desktop, server atau laptop, memerlukan prosesor. Ada banyak produsen prosesor, yang paling popular tentunya adalah seri Pentium buatan intel dan produk dari AMD. Prosesor juga sering sebagai CPU (central processing unit), artinya unti pemrosesan utama di mesin computer. Prosesor sendiri adalah sebuah mesin yang cukup canggih dan rumit, namun dipaket dalam satu chip kecil yang mudah digunakan.

Prosesor sebenarnya belum terlalu lama ditemukan dalam ukuran paket kecil atau dikenal dengan mikro prosesor tersebut. Karena prosesor canggih yang menjadi pendahulu dari prsesor modern baru ada di tahun 1971. Ketika itu intel mengluarkan versi prosesor inter 4004.

Prosesor intel 4004 ketika itu hanya bisa melakukan hal-hal simple seperti operasi aritmatika. Selain itu kapasitasnya juga Cuma bisa mengakomodasi 4 bit saja. Tapi di zaman itu sudah dianggap hebat karena bentuknya yang dalam bentuk chip kecil.

Sebelum intel 4004, para pembuat prosesor harus membuatnya dalam ukuran besar. Baru di intel 4004 ini para pembuat prosesor bisa mem-packing dalam 1 chip.

Mereka harus mengonfigurasikan banyak chip sekaligus dalam satu tempat yang kecil. Di mana rangkaian-rangkaian di dalamnya seperti transistor dihubungkan satu dengan lainnya membnetuk rangkaian kompleks. (Tim E-Media Solusindo ; 2010 : 70)

II.7. Bahasa Pemrograman Visual Basic 2010

Visual Basic merupakan salah satu bahasa pemrograman yang andal dan banyak digunakan oleh pengembang untuk membangun berbagai macam aplikasi Windows. Visual Basic 2010 atau Visual Basic 9 adalah versi terbaru yang telah diluncurkan oleh Microsoft bersama C#, visual C++, dan Visual Web Developer dalam satu paket Visual Studio 2010 (Wahana Komputer; 2010: 2).

Visual Basic 2010 merupakan aplikasi pemrograman yang menggunakan teknologi *.NET Framework*. Teknologi *.NET Framework* merupakan komponen Windows yang terintegrasi serta mendukung pembuata, penggunaan aplikasi, dan halaman web. Teknologi *.NET Framework* mempunyai 2 komponen utama, yaitu CLR (*Common Language Runtime*) dan *Class Library*, CLR digunakan untuk menjalan aplikasi yang berbasis .NET, sedangkan *Library* adalah kelas pustaka atau perintah yang digunakan untuk membangun aplikasi.

Sebelum menginstall komputer harus memenuhi beberapa persyaratan agar Visual Basic 2010 dapat dijalankan dengan baik. Adapun, persyaratan *(System Requirements*) yang harus dipenuhi dapat Anda lihat pada Tabel II.1.

Tabel II.1. Sistem Requirements Visual Basic 2010

Sistem	Syarat Minimal	Syarat yang direkomendasikan
Arsitektur	X86 dan x64 (WOW)	
Sistem Operasi	Microsoft Windows XP Service Pack 2 Microsoft Windows Server 2003 Windows Vista	
Prosesor	CPU 1.6 GHz (Giga Hertz)	Windows XP dan Windows Server 2003:CPU 2,2 GHz atau yang lebih tinggi. Windows Vista : CPU 2,4 GHz
RAM	Windows XP dan Windows Server	RAM 1024 MB / 1

	2003 384 MB (Mega byte) Windows Vista : 768 MB	GB atau yang lebih besar.
Harddisk	Tanpa MSDN Ruang Kosong harddisk pada drive penginstalan 2 GB. Sisa ruang harddisk kosong 1 GB Dengan MSDN Ruang kosong harddisk pada drive penginstalan 3,8 GB (MSDN diinstal full) 2,8 GB untuk menginstal MSDN default. Kecepatan Harddisk 5400 RPM.	Kecepatan harddisk 7200 RPM atau yang lebih tinggi.
Display Layar	1024 x 768 display	1280 x 1024 display

(Sumber : Wahana Komputer ; 2010 : 2)

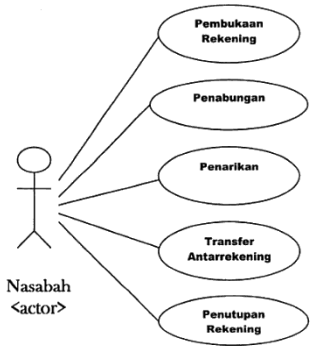
II.8. UML

Pemodelan perangkat lunak bekerja dengan cara yang cukup serupa layaknya seorang arsitek atau insinyur teknik sipil yang akan membuat sebuah bangunan, ia biasanya membuat denah-denah yang menggambarkan bentuk jadi dari bangunan. Sebagai seorang perancang sistem perangkat lunak juga bertindak dengan cara yang serupa, hanya saja yang dirancang bukan bangunan, melainkan sistem perangkat lunak. Menggambarkan komponen sistem perangkat lunak dalam bentuk-bentuk geometri tertentu dalam aplikasi (*Adi Nugroho; 2009:6*).

II.8.1. Diagram Use Case

Membuat *use case* diagram yang komprehensif merupakan hal yang sangat penting dilakukan pada tahap analisis. Dengan menggunakan *use case* diagram, akan mendapatkan banyak informasi yang sangat penting yang berkaitan dengan aturan-aturan bisnis yang coba ditangkap. Dalam hal ini, setiap objek yang berinteraksi dengan sistem perangkat lunak misalnya, orang, suatu perangkat keras, sistem lain, dan sebagainya merupakan *actor* untuk sistem perangkat lunak,

sementara *use case* merupakan deskripsi lengkap tentang bagaimana sistem perangkat lunak berperilaku untuk para *actor*nya. Dengan demikian, *use* cara diagram merupakan deskripsi lengkap tentang interaksi yang terjadi antara para *actor* dengan sistem perangkat lunak yang sedang kita kembangkan. Untuk lebih jelas dapat dilihat pada Gambar II.4.



Gambar II.4. Diagram *Use Case*
(Sumber : Adi Nugroho; 2010 : 8)

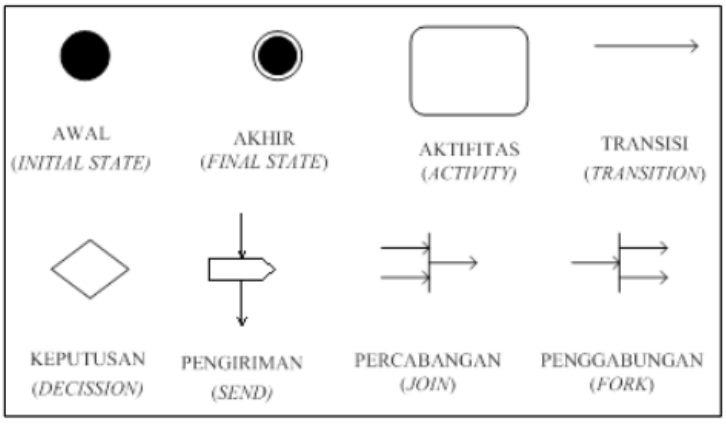
Actor pada dasarnya ditentukan berdasarkan perannya (role) pada program /aplikasi yang sedang kita kembangkan, bukan sebagai objek-objek secara mandiri. Sebagai contoh, jika mengambil kasus ATM (Anjungan Tunai Mandiri), seseorang (objek tunggal) mungkin bisa dikelompokkan sebagai actor Karyawan Bank serta Nasabah jika oarng tersebut merupakan karyawan bank yang bersangkutan sekaligus sebagai nasabah karena memiliki tabungan di bank tersebut. Sementara itu, Adi, Ana Geuis, dan beberapa orang lainnya dapat dikelompokkan menjadi actor nasabah jika mereka semua masing-masing memiliki tabungan di bank tersebut. Dalam hal ini, kita akan coba mengambil contoh *actor* nasabah untuk menentukan *use case*nya.

UML (*Unified Modeling Language*) adalah metode pemodelan secara visual sebagai sarana untuk merancang dan atau membuat *software* berorientasi

objek, karena UML ini merupakan bahasa visual untuk pemodelan bahasa berorientasi objek, maka semua elemen dan diagram berbasiskan pada paradigma *object oriented*. UML juga memberikan standar penulisan sebuah sistem blueprint, yang meliputi konsep bisnis proses, penulisan kelas-kelas dalam bahasa program yang spesifik, skema *database*, dan komponen-komponen yang diperlukan dalam sistem *software*. UML terdiri dari beberapa diagram, yaitu *use case* diagram, *class* diagram, *state* diagram, *activity* diagram, *sequence* diagram, *collaboration* diagram, *component* diagram, dan *deployment* diagram. (MHD Imam Alfarisyi, dkk ; 2011 : 55).

III.8.2. Diagram *Activity*

Activity diagram menggambarkan urutan aktifitas yang digunakan untuk menjelaskan aktifitas dari sebuah operasi. Pada activity diagram terdapat keadaan aksi yang berisi spesifikasi dari aktifitas tertentu. Diagram ini berisi, pilihan keputusan dan kondisi serta spesifikasi message yang dikirim atau diterima sebagai gambaran dari aksi.



Gambar II.5. Diagram Activity
(Sumber : MHD Imam Alfarisyi, dkk ; 2011 : 56)

III.8.3. *Sequence*

Diagram interaksi yang menekankan pada waktu pengiriman *message*. *Sequence* diagram menunjukkan sekumpulan objek dan pengiriman serta penerimaan *message* antar objek. Objek yang umumnya memiliki nama atau instansiasi dari *class*, tapi dapat pula merupakan turunan dari *things* lain, seperti *collaboration*, *component* dan *node*. Diagram ini digunakan untuk mengilustrasikan *dynamic view* dari sistem.

III.8.4. *State*

Sebuah *state* diagram menggambarkan keadaan mesin, transisi, *event* dan *activity*. Diagram ini adalah pelengkap khusus untuk mendeskripsikan sebuah *class* yang menggambarkan *state* dari objek dari *class* dan *event* yang menyebabkan *state* berubah. *Event* tersebt dapat berasal dari objek yang mengirimkan suatu *message* atau dari kondisi yang terpenuhi.

II.9. Perancangan

Untuk membuat tampilan yang menarik memang tidak mudah dilakukan. Seorang perancang tampilan selain harus mempunyai jiwa seni yang memadai, ia juga harus mengerti selera pengguna secara umum. Hal lain yang pelu disadari oleh seorang perancang tampilan adalah bahwa ia harus bisa meyakinkan pemrogramnya bahwa apa yang ia bayangkan dapat diwujudkan dengan peranti bantu yang tersedia (Insap Santoso ; 2010 : 185).

Perancangan merupakan proses pengolahan hasil analisis perangkat lunak menjadi rencana pengembangan perangkat lunak dan batasan-batasan perangkat lunak atau masalah yang mungkin dihadapi dalam pengembangan perangkat

lunak. Perancangan yang dilakukan meliputi perancangan arsitektur, perancangan modul, dan perancangan antarmuka. (Y. Yohakim Marwanta ; 2010 : 5).

Bagi perancang antarmuka, hal yang sangat penting untuk ia perhatikan adalah mendokumentasikan semua pekerjaan yang dilakukan. Dokumentasi rancangan dapat dikerjakan atau dilakukan dengan beberapa cara :

1. Membuat sketsa pada kertas
2. Menggunakan peranti purwarupa GUI
3. Menuliskan keterangan yang menjelaskan tentang kaitan antara jendela.
4. Menggunakan peranti bantu CASE (*Computer Aided Software Engineering*).

Cara kedua dan keempat tidak selalu dapat diterapkan, karena peranti tersebut biasanya harus dibeli dan seringkali cukup mahal. Cara ini kebanyakan diterapkan pada pembuatan antarmuka grafis untuk suatu jenis pekerjaan berskala besar.

II.9.1. Cara Pendekatan

Sebuah program aplikasi pastilah ditujukan kepada pengguna, yang utama, bukan perancangan program aplikasi tersebut. Program aplikasi pada dasarnya dapat dikelompokkan dalam dua kategori besar, yakni program aplikasi untuk keperluan khusus dengan pengguna yang khusus pula dan program aplikasi yang akan digunakan oleh pengguna umum, yang juga sering dikenal dengan sebutan public software. Karena perbedaan pada calon pengguna, maka perancang program antarmuka perlu memperhatikan hal ini (Insap Santoso ; 2009 : 186).

Pada kelompok pertama, yakni pada program aplikasi untuk keperluan khusus, misalnya program aplikasi untuk inventori gudang, pengelolaan data

akademis mahasiswa, pelayanan reservasi hotel, dan program-program aplikasi yang serupa, kelompok calon pengguna yang akan memanfaatkan program aplikasi tersebut dapat dengan mudah diperkirakan, baik dalam hal keahlian pengguna maupun ragam antarmuka yang akan digunakan. Untuk kelompok ini ada satu pendekatan yang dapat dilakukan, yakni pendekatan yang disebut dengan pendekatan perancangan berpusat ke pengguna (*user centered design approach*). Cara pendekatan ini berbeda pendekatan perancangan oleh pengguna (*user design approach*).

Pendekatan perancangan berpusat ke pengguna adalah perancangan antarmuka yang melibatkan pengguna. Pelibatan pengguna di sini tidak diartikan bahwa pengguna harus ikut memikirkan bagaimana implementasinya nanti, tetapi pengguna diajak untuk aktif berpendapat ketika perancangan antarmuka sedang menggambar wajah antarmuka yang mereka inginkan. Dengan kata lain, perancangan dan pengguna duduk bersama-sama untuk merancang wajah antarmuka yang diinginkan pengguna. Pengguna menyampaikan keinginannya. Sementara perancangan menggambar keinginan pengguna tersebut sambil menjelaskan keuntungan dan kerugian wajah antarmuka yang diinginkan oleh pengguna, seolah-olah sudah mempunyai gambaran nyata tentang antarmuka yang nanti akan mereka gunakan (Insap Santoso ; 2010 : 187).

Pada perancangan oleh pengguna, pengguna sendirilah yang merancang wajah antarmuka yang diinginkan. Di satu sisi, cara ini akan mempercepat proses pengimplementasian modul antarmuka. Tetapi di sisi yang lain, hal ini justru sangat memberatkan pemrogram karena apa yang diinginkan pengguna belum

tentu dapa diimplementasikan dengan mudah, atau bahkan tidak dapat dikerjakan dengan menggunakan peranti bantu yang ada.

Perancang program aplikasi yang dimasukkan dalam kelompok kedua, aau public *software,* perlua menganggap bahwa program aplikasi tersebut akan digunakan oleh pengguna dengan berbagai tingkat kepandaian dan karakteristik yang sangat beragam. Di satu sisi keadaan ini dapat ia gunakan untuk memaksa pengguna menggunakan antarmuka yang ia buat, tetapi pada sisi lain pemaksaan itu akan berakibat bahwa program aplikasinya menjadi tidak banyak penggunanya. Satu kunci penting dalam pembuatan modul antarmuka untuk program-program aplikasi pada kelompok ini adalah dengan melakukan customization. Dengan customization pengguna dapat menggunakan program aplikasi dengan wajah antarmuka yang sesuai dengan selera masing-masing pengguna.

Salah satu contoh dari adanya kemampuan yang dimiliki oleh sebuah program aplikasi atau sistem operasi yang dapat disesuaikan dengan karakteristik pengguna adalah pengaturan desktop pada OS X versi 10.5 favoritnya, sehingga pengguna dapat mengubahnya sesuai keinginan justru akan membuat mata pengguna itu sakit, dikarenakan mata harus melakukan akomomasi maksimum terus menerus untuk menyesuaikan dengan warna tampilan yang ada.

Selain cara pendekatan yang dijelaskan di atas, Anda yang terbiasa menulis program-program aplikasi mungkin mempunyai cara khusus untuk berhadapan dengan pengguna. Tetapi perlu Anda ingat bahwa apapun cara yang Anda gunakan, Anda tetap harus mempunyai pedoman bahwa pada akhirnya program itu bukan untuk Anda sendiri, tetapi akan digunakan oleh orang lain.

Dengan kata lain, jangan pernah mengabaikan pendapat (calon) pengguna program aplikasi Anda. (Insap Santoso ; 2010 : 188).

II.9.2. Prinsip Dan Petunjuk Perancangan

Antarmuka pengguna secara alamiah terbagi menjadi empat komponen model pengguna, bahasa perintah, umpan balik, dan penampilan informasi. Model pengguna merupakan dasar dari tiga komponen yang lain (Insap Santoso ; 2010 : 188).

Model mental pengguna merupakan model konseptual yang dimiliki oleh pengguna ketika ia menggunakan sebuah sistem atau program aplikasi. Model ini memungkinkan seorang pengguna untuk mengembangkan pemahaman mendasar tentang bagian yang yang dikerjakan oleh program, bahkan oleh pengguna yang sama sekali tidak mengetahui teknologi komputer. Dengan pertolongan model itu pengguna dapat mengantisipasi pengaruh suatu tindakan yang dilakukan dan dapat memilih strategi yang cocok untuk mengoperasikan program tersebut. Model pengguna dapat berupa suatu simulasi tentang keadaan yang sebenarnya dalam dunia nyata, sehingga ia tidak perlu mengembangkannya sendiri dari awal.

Setelah pengguna mengetahui dan memahami model yang di inginkan, dia memerlukan peranti untuk memanipulasi model itu. Peranti pemanipulasian model ini sering disebut dengan bahasa perintah (command language), yang sekaligus merupakan komponen kedua dari antarmuka pengguna. Idealnya program komputer kita mempunyai bahasa perintah yang alami, sehingga model pengguna dengan cepat dapat dioperasionalkan. (Insap Santoso ; 2010 : 189).

Komponen ketiga adalah umpan balik. Umpan balik di sini diartikan sebagai kemampuan sebuah program yang membantu pengguna untuk

mengoperasikan program itu sendiri. Umpan balik dapat berbentuk pesan penjelasan, pesan penerimaan perintah, indikasi adanya obyek terpilih, dan penampilan karakter yang diketikkan lewat papan ketik. Beberapa bentuk umpan balik terutama ditujukan kepada pengguna pengguna yang belum berpengalaman dalam menjalankan program sebuah aplikasi. Umpan balik dapat digunakan untuk member keyakinan bahwa program telah menerima perintah pengguna dan dapat memahami maksud perintah tersebut.

Komponen keempa adalah tampilan informasi. Komponen ini digunakan untuk menunjukkan status informasi atau program ketika pengguna melakukan suatu tindakan. Pada bagian ini perancang harus menampilkan pesan-pesan tersebut seefektif mungkin sehingga mudah dipahami oleh pengguna. Setelah memahami beberapa prinsip dalam perancangan antarmuka pengguna. Pada bagian berikut ini akan diberikan petunjuk singkat tentang perancangan anarmuka yang akan Anda lakukan sebagai seorang perancang tampilan.

II.9.3. Urutan Perancangan

Perancangan dialog, seperti halnya perancangan sistem yang lain, harus dikerjakan secara aas ke bawah. Proses perancangannya dapat dikerjakan secara bertahap sampai rancangan yang diinginkan terbentuk, yaitu sebagai berikut (Insap Santoso ; 2010 : 190).

1. Pemilihan ragam dialog

Untuk suatu tugas tertentu, pilihlah ragam dialog yang menurut perkiraan cocok untuk tugas tersebut. Ragam dialog dapat dipilih dari sejumlah ragam dialog yang telah dijelaskan pada bab-bab sebelumnya. Pemilihan ragam dialog dipengaruhi oleh karakterisiktik populasi pengguna, tipa dialog yang diperlukan,

dan kendala teknologi yang ada untuk mengimplementasikan ragam dialog tersebut. Ragam dialog yang terpilih dapat berupa sebuah ragam tunggal, atau sekumpulan ragam dialog yang satu sama lain saling mendukung.

2. Perancangan Struktur Dialog

Tahap kedua adalah melakukan analisis tugas dan menentukan model pengguna dari tugas tersebut untuk membentuk struktur dialog yang sesuai. Dalam tahap ini pengguna sebaiknya banyak dilibatkan, sehingga pengguna langsung mendapatkan umpan balik dari diskusi yang terjadi. Pada tahap ini suatu purwarupa dialog seringkali dibuat untuk memberik gambarang yang lebih jelas kepada calon pengguna.

3. Perancangan format pesan

Pada tahap ini tata letak tampilan dan keterangan tekstual secara terinci harus mendapat perhatian lebih. Selain itu, kebutuhan data masukan yang mengharuskan pengguna untuk memasukkan data ke dalam komputer juga harus dipertimbangkan dari segi efisiensinya. Salah satu contohnya adalah dengan mengurangi pengetikan yang tidak perlu dengan cara mengefektifkan pengguna tombol.