BAB II

Landasan Teori

2.1. Definisi Sistem

Menurut Fat, pengertian sistem adalah suatu himpunan suatu "benda" nyata atau abstrak (*a set of thing*) yang terdiri dari bagian-bagian atau komponen-komponen yang sailing berkaitan, berhubungan, berketergantungan, saling mendukung, yang secara keseluruhan bersatu dalam satu kesatuan (*Unity*) untuk mencapai tujuan tertentu secara efisien dan efektif".

2.2. Karakteristik Sistem

Sistem dapat dikatakan sebagai sistem yang baik apabila memiliki karakteristik sebagai berikut:

a) Komponen

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

b) Batasan sistem (*boundary*)

Batasan sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lain atau dengan lingkungan luarnya. Batasan suatu sistem menunjukkan ruang lingkup dari sistem tersebut.

c) Lingkungan luar sistem (*environment*)

Lingkungan luar sistem (*environment*) adalah diluar batas dari sistem yang mempengaruhi operasi sistem.

d) 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 masukkan (*input*) untuk subsistem lain melalui penghubung.

e) Masukkan sistem (*input*)

Masukkan adalah energi yang dimasukkan kedalam sistem, yang dapat berupa perawatan (*maintenance input*), dan masukkan sinyal (*signal input*). *Maintenance input* adalah energi yang dimasukkan agar sistem dapat beroperasi. *Signal input* adalah energi yang diproses untuk didapatkan keluaran.

f) Keluaran sistem (*ouput*)

Keluaran sistem adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan.

g) Pengolah sistem

Suatu sistem menjadi bagian pengolah yang akan merubah masukkan menjadi keluaran.

h) Sasaran sistem

Suatu sistem pasti mempunyai tujuan atau sasaran. Sasaran dari sistem sangat menentukkan *input* yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem.

2.3. Analisis Sistem

Analisis sistem ialah penjabaran dari suatu sistem informasi yang utuh ke dalam berbagai bagian komponennya dengan maksud agar bisa mengidentifikasi dan mengevaluasi berbagai macam masalah atau hambatan yang timbul pada sistem sehingga nantinya bisa dilakukan penanggulangan, perbaikan dan juga pengembangan.

2.4. Pengertian Informasi

Informasi adalah data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti bagi yang menerimanya. Sumber dari informasi adalah data. Data merupakan bentuk jamak dari bentuk tunggal data atau *data item*. Data adalah kenyataan yang menggambarkan suatu kejadian-kejadian dan kesatuan nyata. (**Jogiyanto**,2005:11). Data dapat berbentuk nilai yang terformat, teks, citra, audio, dan video.

2.5. Sistem Informasi

Sistem Informasi adalah kombinasi dari teknologi informasi dan aktivitas orang yang menggunakan teknologi itu untuk mendukung operasi dan manajemen. Tujuan dari sistem informasi adalah menghasilkan informasi.

Dalam bukunya Bambang **Hartono** (2013:16). Menurut **Lippeveld**, **Sauerborn**, dan **Bodart** (2000), sistem informasi adalah seperangkat komponen yang saling berhubungan, yang bekerja untuk mengumpulkan dan menyimpan data serta mengolahnya menjadi informasi yang digunakan.

2.6. Pengertian Rancang Bangun

Rancang bangun merupakan kegiatan menerjemahkan hasil analisa ke dalam bentuk paket perangkat lunak kemudian menciptakan sistem tersebut ataupun memperbaiki sistem yang sudah ada.

2.7. Aplikasi

Aplikasi adalah suatu perangkat lunak (*software*) atau program komputer yang beroperasi pada sistem tertentu yang diciptakan dan dikembangkan untuk melakukan perintah tertentu.

Istilah aplikasi sendiri diambil dari bahasa Inggris "*application*" yang dapat diartikan sebagai penerapan atau penggunaan. Secara harfiah, aplikasi merupakan suatu penerapan perangkat lunak yang dikembangkan bertujuan untuk melakukan tugas-tugas tertentu.

2.8. Sistem Pakar

Sistem pakar merupakan cabang dari *Artificial Itellegence* (AI) yang cukup tua karena sistem ini mulai dikembangkan pada pertengahan 1960. Sistem pakar yang muncul pertama kali adalah *General-purpose Problem Solver* (GPS) yang dikembangkan oleh Newel dan Simon. Sampai saat ini banyak sistem pakar yang dibuat, seperti MYCIN untuk diagnosis penyakit, DENDRAL untuk mengidentifikasi struktur molekul campuran yang tak dikenal, XCON dan XSEL untuk membantu konfigurasi sistem komputer besar, SOPHIE untuk analisis sirkuit elektronik, Prospector digunakan di bidang geologi untuk membantu mencari dan menemukan deposit, FOLIO digunakan untuk membantu memberikan keputusan bagi seorang manajer dalam stok dan investasi, DELTA dipakai untuk pemeliharaan lokomotif listrik diesel, dan sebagainya (Sutojo, dkk., 2011:159).

2.9. Pengertian Sepeda Motor

Berdasarkan KBBI (Kamus Besar Bahasa Indonesia), sepeda motor merupakan sepeda besar yang dijalankan oleh motor. Jika dijabarkan, sepeda merupakan kendaraan beroda dua atau tiga, mempunyai setang, tempat duduk, dan sepasang pengayuh yang digerakkan kaki untuk menjalankannya, sedangkan motor merupakan mesin yang menjadi tenaga penggerak.

2.10. Pengertian RAD (*Rapid Application Development*)

RAD (*Rapid Application Development*) adalah suatu pendekatan berorientasi objek terhadap pengembangan sistem yang mencakup suatu metode pengembangan serta perangkat-perangkat lunak. RAD bertujuan mempersingkat waktu yang

biasanya diperlukan dalam siklus hidup pengembangan sistem tradisional antara perancangan dan penerapan suatu sistem informasi (Kendall 2010).

Menurut Kendall (2010), terdapat tiga fase dalam RAD yang melibatkan penganalisis dan pengguna dalam tahap penilaian, perancangan, dan penerapan. Adapun ketiga fase tersebut adalah requirements planning (perencanaan syarat-syarat), RAD design workshop (workshop desain RAD), dan implementation (implementasi). Sesuai dengan metodologi RAD menurut Kendall (2010), berikut ini adalah tahap-tahap pengembangan aplikasi dari tiap-tiap fase pengembangan aplikasi.

2.10.1. *Requirements Planning* (Syarat-Syarat Perencanaan)

Dalam fase ini, pengguna dan penganalisis bertemu untuk mengidentifikasikan tujuan-tujuan aplikasi atau sistem serta untuk megidentifikasikan syarat-syarat informasi yang ditimbulkan dari tujuan-tujuan tersebut. Orientasi dalam fase ini adalah menyelesaikan masalah-masalah perusahaan. Meskipun teknologi informasi dan sistem bisa mengarahkan sebagian dari sistem yang diajukan, fokusnya akan selalu tetap pada upaya pencapaian tujuan-tujuan perusahaan

2.10.2. *RAD Design Workshop* (*Workshop Desain RAD*)

Fase ini adalah fase untuk merancang dan memperbaiki yang bisa digambarkan sebagai workshop. Penganalisis dan dan pemrogram dapat bekerja membangun dan menunjukkan representasi visual desain dan pola kerja kepada pengguna. Workshop desain ini dapat dilakukan selama

beberapa hari tergantung dari ukuran aplikasi yang akan dikembangkan. Selama workshop desain RAD, pengguna merespon prototipe yang ada dan penganalisis memperbaiki modul-modul yang dirancang berdasarkan respon pengguna. Apabila sorang pengembangnya merupakan pengembang atau pengguna yang berpengalaman, Kendall menilai bahwa usaha kreatif ini dapat mendorong pengembangan sampai pada tingkat terakselerasi.

2.10.3. *Implementation* (Implementasi)

Pada fase implementasi ini, penganalisis bekerja dengan para pengguna secara intens selama workshop dan merancang aspek-aspek bisnis dan nonteknis perusahaan. Segera setelah aspek-aspek ini disetujui dan sistem-sistem dibangun dan disaring, sistem-sistem baru atau bagian dari sistem diujicoba dan kemudian diperkenalkan kepada organisasi.

2.11. Pengertian Certainty Factor

Poni Wijayanti dan Abdul Fadlil (2014:692) mendefinisikan *certainty factor* (faktor kepastian) adalah:

"Faktor kepastian merupakan cara dari penggabungan kepercayaan (*belief*) dan ketidakpercayaan (*unbelief*) dalam bilangan yang tunggal dalam *certainty theory*, data-data kualitatif direpresentasikan sebagai derajat keyakitan (*degree of belief*)".

Pada tahun 1975, teori *certainty factor* awalnya diusulkan oleh Shortlife dan Buchanan untuk mengakomodasi ketidakpastian pemikiran (*inexact reasoning*) seorang pakar. Seorang pakar (misalnya dokter), sering kali menganalisis informasi

pada saat konsultasi diagnosis dengan ungkapan seperti “mungkin”, “kemungkinan besar” atau “hampir pasti”.

Terdapat dua cara untuk mendapatkan nilai *certainty factor* dari sebuah *rule*, yaitu:

1) Metode *Net Belief* yang diusulkan oleh E.H Shortlife dan B.G Buchanan

CF(H,E) = MB(H,E) – MD(H,E)

Dimana:

CF(H,E) = Faktor kepastian dari hipotesis H yang mempengaruhi oleh gejala (*evidence*) E.

MB(H,E) = Ukuran kenaikan kepercayaan (*measure of increased belief*) terhadap hipotesis H yang dipengaruhi oleh gejala E.

MD(H,E) = Ukuran kenaikan ketidakpercayaan (*measure of increased disbelief*) terhadap hipotesis H yang dipengaruhi oleh gejala E.

2) Dengan wawancara seorang pakar/ahli

Nilai CF didapatkan berdasarkan interpretasi “term” dari pakar yang diubah menjadi nilai CF tertentu, seperti yang ditujukan pada tabel berikut:

Tabel 2.1 Nilai CF berdasarkan interprestasi dari pakar

Intepretasi Tidak Pasti (*Uncertain Term*)	**Nilai *Certainty Factor***
Pasti tidak	-1.0
Hampir pasti tidak	-0.8
Kemungkinan besar tidak	-0.6

Mungkin tidak	-0.4
Tidak tahu	-0.2 sampai 0.2
Mungkin	0.4
Kemungkinan besar	0.6
Hampir pasti	0.8
Pasti	1.0

Sedangkan pada proses penerapan sistem pakar ini, *rule* nilai yang akan digunakan ditunjukkan pada tabel berikut.

Tabel 2.2 Nilai CF yang akan digunakan

Keterangan	**CF**
Tidak	0
Tidak Yakin	0,2
Mungkin	0,4
Cukup Yakin	0,6
Yakin	0,8

Dengan menggunakan sistem, pengguna akan memberikan nilai CF pada setiap premis yang ada dalam aturan. Untuk menentukan nilai CF dari suatu aturan, terdapat beberapa premis dengan faktor kepastiannya sebagai berikut:

1) *Rule* dengan *evidence* E tunggal dan hipotesis H tunggal.

   IF E THEN H (CF *rule*)

   CF (H,E) = CF (E) x CF (*rule*)

2) *Rule* dengan *evidence* E ganda dan hipotesis H tunggal.

IF E_1 AND E_2 ..... AND E_n THEN H (CF *rule*)

CF (H,E) = min [CF (E_1), CF (E_2), ..., CF (E_n)] x CF (*rule*)

Atau

IF E_1 OR E_2 ..... OR E_n THEN H (CF *rule*)

CF (H,E) = max [CF (E_1), CF (E_2), ..., CF (E_n)] x CF (*rule*)

3) Kombinasi dua buah *rule* dengan *evidence* berbeda (E_1 dan E_2) tetapi hipotesis H sama.

IF E_1 THEN H Rule 1 CF(H_1E_1) – CF_1 = C(E_1) x CF (Rule 1)

IF E_2 THEN H Rule 2 CF(H_2E_2) – CF_2 = C(E_2) x CF (Rule 2)

$$\mathrm{CF}(\mathrm{C1},\mathrm{C2}) = \begin{cases} \mathrm{CF1} + \mathrm{CF2} \times (1 - \mathrm{CF1}) & Jika\ CF1 > 0\ atau\ CF2 > 0 \\ \dfrac{CF1 + CF2}{1 - \min\left[|CF1| - |CF2|\right]} & Jika\ CF1 < 0\ atau\ CF2 < 0 \\ CF1 + CF2 \times (1 + CF1) & Jika\ CF1 < 0\ atau\ CF2 < 0 \end{cases}$$

2.12. Android

Android adalah sebuah sistem operasi perangkat *mobile* berbasis linux yang mencangkup sistem operasi, *middleware* dan aplikasi. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka.

2.13. UML

UML singkatan dari *Unified Modeling Language* yang berarti bahasa pemrograman pemodelan standar. **Chonoles** (2003) mengatakan sebagai bahasa, berarti UML memiliki sintaks dan semantic. Ketika kita membuat model

menggunakan konsep UML ada aturan-aturan yang harus diikuti Bagaimana elemen pada model-model yang kita buat berhubungan satu dengan lainnya harus mengikuti standar yang ada.

UML memiliki berbagai jenis diagram yaitu :

a) Diagram Kelas, Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi.

b) Diagram Paket (*Package Diagram*). Diagram ini memperlihatkan kumpulan kelas-kelas, merupakan bagian dari diagram komponen

c) Diagram *Use Case*. Diagram ini terutama sangat penting untuk mengorganisasi dan memodelkan perilaku suatu sistem yang dibutuhkan serta diharapkan pengguna.

d) Diagram Interaksi dan *Sequence* (urutan). Diagram ini adalah diagram interaksi yang menekankan pada pengiriman pesan dalam suatu waktu tertentu.

e) Diagram Komunikasi (*Communication Diagram*). Diagram sebagai pengganti diagram kolaborasi UML 1.4 yang menekankan organisasi structural dari objek-objek yang menerima serta mengirim pesan.

f) Diagram *Statechart* (*Statechart Diagram*). Diagram status memperlihatkan keadaan-keadaan pada sistem, memuat status (*state*), transisi, kejadian serta aktivitas.

g) Diagram Aktivitas (*Activity Diagram*). Diagram aktivitas adalah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem.

h) Diagram Komponen (*Component Diagram*). Diagram komponen ini memperlihatkan organisasi serta ketergantungan sistem/perangkat lunak pada komponen-komponen yang telah ada sebelumnya.

Kesembilan diagram ini tidak mutlak harus digunakan dalam pengembangan perangkat lunak, semuanya dibuat sesuai dengan kebutuhan.

2.14. DFD

Menurut **Andri Kristanto** (2003 : 55), menjelaskan : "*Data Flow Diagram* adalah suatu model logika data atau proses yang dibuat untuk menggambarkan dari mana asal data dan kemana tujuan data yang keluar dari sistem, dimana data tersimpan, proses apa yang keluar dari sistem, dimana data tersimpan, proses apa yang menghasilkan data tersebut dan interaksi antara data tersimpan dan proses yang dikenakan pada data tersebut". DFD memiliki beberapa simbol yang digunakan untuk menjelaskan diagram yang dibuat, simbol-simbol yang digunakan yaitu:

a) Proses (*Process*), adalah kegiatan yang dilakukan oleh orang, mesin, atau komputer dari hasil suatu data yang masuk ke dalam proses untuk menghasilkan data yang keluar dari proses.
b) Entitas Eksternal (*External Entity*), mempresentasikan sebuah entitas sebagai sebuah elemen sistem.
c) Penyimpanan Data (*Data Store*), merupakan simpanan dari data yang dapat berupa file komputer, database komputer, arsip atau catatan manual,suatu

kontak tempat data di meja seseorang, suatu label acuan seseorang, suatu agenda, atau buku.

d) Arus Data (*Data Flow*), merupakan panah yang mempresentasikan data atau lebih objek data.

DFD terdapat 3 level, yaitu :

a) Diagram Konteks : menggambarkan satu lingkaran besar yang dapat mewakili seluruh proses yang terdapat di dalam suatu sistem.

b) Diagram Nol : merupakan satu lingkaran besar yang mewakili lingkaran-lingkaran kecil yang ada di dalamnya.

c) Diagram Rinci : merupakan diagram yang menguraikan proses apa yang ada dalam diagram nol.