BAB II
TINJAUAN PUSTAKA

II.1. Konsep Dasar Sistem

Sistem merupakan kumpulan dari unsur atau elemen-elemen yang saling berkaitan/berinteraksi dan saling mempengaruhi dalam melakukan kegiatan bersama untuk mencapai suatu tujuan tertentu.

Menurut Jerry FithGerald, sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan dan berkumpul bersama-sama untuk melakukan suatu kegiatan atau menyelesaikan suatu sasaran tertentu.

Menurut Ludwig Von Bartalanfy, sistem merupakan seperangkat unsure yang saling terkait dalam suatu antar-relasi di antara unsur-unsur tersebut dengan lingkungan.

Menurut Anatol Raporot, sistem adalah suatu kumpulan kesatuan dan perangkat hubungan satu sama lain.

Menurut L. Ackof, sistem adalah setiap kesatuan secara konseptual atau fisik yang terdiri dari bagian-bagian dalam keadaan saling tergantung satu sama lainnya (Asbon Hendra ; 2012 : 157).

II.1.1. Syarat - syarat Sistem

a. Sistem harus dibentuk untuk menyelesaikan tujuan.
b. Elemen sistem harus mempunyai rencana yang ditetapkan.
c. Adanya hubungan di antara elemen sistem.

d. Unsur dasar dari proses (arus informasi, energy, dan material) lebih penting dari pada elemen sistem.
e. Tujuan organisasi lebih penting dari pada tujuan elemen (Asbon Hendra ; 2012 : 157).

II.1.2. Karakteristik Sistem

a. Komponen (Component)

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, bekerja sama membentuk satu kesatuan. Komponen-komponen sistem dapat berupa suatu subsistem atau bagian-bagian dari sistem. Setiap sistem, tidak peduli betapa pun kecilnya, selalu mengandung komponen-komponen atau subsistem-subsistem. Setiap subsistem mempunyai sifat-sifat dari sistem untuk menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan. Suatu sistem dapat mempunyai suatu sistem yang lebih besar yang disebut *supra sistem*. Sebagai contoh suatu perusahaan dapat disebut dengan suatu sistem dan industri yang merupakan sistem yang lebih besar dapat disebut dengan supra sistem. Kalau dipandang industri sebagai suatu sistem, maka perusahaan dapat disebut sebagai subsistem,. Demikian juga bila perusahaan dipandang sebagai suatu sistem, maka sistem akuntansi adalah subsistemnya.

b. Batas Sistem (*Boundary*)

Batas sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipandang sebagai suatu kesatuan, karena dengan batas sistem ini fungsi dan tugas dari subsistem yang satu dengan lainnya berbeda

tetapi tetap saling berinteraksi. Batas suatu sistem menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

c. Lingkungan Luar Sistem (*Environment*)

Environmen merupakan segala sesuatu di luar batas sistem yang mempengaruhi operasi dari suatu sistem. Lingkungan luar sistem ini dapat bersifat menguntungkan atau merugikan. Lingkungan luar yang menguntungkan harus dipelihara dan dijaga agar tidak hilang pengaruhnya, sedangkan lingkungan luar yang merugikan harus dimusnakan atau dikendalikan agar tidak mengganggu operasi sistem.

d. Penghubung Sistem (*Interface*)

Merupakan media penghubung antara satu subsistem dengan subsistem yang lainnya untuk membentuk satu kesatuan sehingga sumber-sumber daya mengalir dari subsistem yang satu ke subsistem yang lainnya. Dengan kata lain, *output* dari suatu subsistem akan menjadi *input* dari subsistem yang lainnya.

e. Masukan Sistem (*Input*)

Merupakan energy yang dimasukkan ke dalam sistem. Masukan dapat berupa Masukan Perawatan (***Maintenance Input***) adalah energy yang dimasukkan supaya sistem tersebut dapat beroperasi. Masukan sinyal (***Signal Input***) adalah energy yang diproses untuk didapatkan keluaran. Sebagai contoh, di dalam sistem computer, program adalah *maintenance input* yang digunakan untuk mengoperasikan komputernya dan data adalah signal input untuk diolah menjadi informasi.

f. Keluaran Sistem (*Output*)

Merupakan hasil dari energi yang diolah oleh sistem, meliputi output yang berguna, contohnya informasi yang dikeluarkan oleh computer. Dan output yang tidak berguna dikenal sebagai sisa pembuangan, contohnya panas yang dikeluarkan oleh komputer.

g. Pengolah Sistem (*Process*)

Merupakan bagian yang memproses masukan untuk menjadi keluaran yang diinginkan. Contoh CPU pada computer, bagian produksi yang mengubah bahan baku menjadi barang jadi, serta bagian akuntansi yang mengolah data transaksi menjadi laporan keuangan.

h. Tujuan Sistem (Goal)

Setiap sistem pasti mempunyai tujuan ataupun sasaran yang mempengaruhi input yang dibutuhkan dan output yang dihasilkan. Dengan kata lain, suatu sistem akan dikatakan berhasil kalau pengoperasian sistem itu mengenai sasaran atau tujuannya. Jika sistem tidak mempunyai sasaran, maka operasi sistem tidak aka nada gunanya (Asbon Hendra ; 2012 : 156-162).

II.1.3. Sistem

Secara umum sistem dapat didefinisikan sebagai sekumpulan objek, ide, berikut saling ketergantungan (inter-relasi) di dalam usaha mencapai suatu tujuan atau dengan kata lain, sistem disebutkan sebagai kumpulan komponen (subsistem fisik maupun non-fisik/logika) yang saling berhubungan satu sama lainnya dan bekerja sama secara harmonis untuk mencapai suatu tujuan (Eddy Prahasta ; 2009 : 89).

II.1.4. Sistem Informasi

Sistem informasi adalah sekumpulan komponen- komponen yang saling berhubungan dan bekerja sama untuk mengumpulkan , memproses, menyimpan, dan mendistribusikan informasi terkait untuk mendukung proses pengambilan keputusan, koordinasi, dan pengendalian (Eddy Prahasta ; 2009 : 93).

II.1.5. Akuntansi

Akuntansi adalah seni daripada pencatatan, penggolongan dan peringkasan daripada peristiwa-peristiwa dan kejadian-kejadian yang setidaknya sebagian bersifat keuangan dengan cara yang setepa-tepatnya dan dengan penunjuk atau dinyatakan dalam uang, serta penafsiran terhadap hal-hal yang timbul daripadanya (Drs. S. Munawir. Akuntan ; 2010: 05).

II.1.6. Sistem Informasi Akuntansi

Kumpulan beberapa sub sistem yang saling berhubungan satu sama lain dan bekerja sama dengan harmonis dalam mengolah data keuangan sedemikian rupa hingga menghasilkan informasi keuangan bagi (pihak) manajemen untuk membantu pengambilan keputusan dibidang keuangan (Eddy Prahasta ; 2009 : 108).

II.1.7. Laporan Keuangan

Laporan keuangan pada dasarnya adalah hasil dari proses akuntansi yang dapat digunakan sebagai alat untuk berkomunikasi antara data keuangan atau

aktivitas suatu perusahaan dengan pihak- pihak yang berkepentingan dengan data atau aktivitas perusahaan tersebut (Drs. S. Munawir. Akuntan ; 2010: 02). Adapun tujuan dari pembuatan laporan keuangan sebagai berikut :

1. Untuk memberikan informasi keuangan yang dapat dipercaya mengenai sumber-sumber ekonomi, dan kewajiban serta modal suatu perusahaan.

2. Untuk mamberikan informasi penting lainnya mengenai perubahan dalam sumber-sumber ekonomi dan kewajiban, sperti informasi mengenai aktivitas pembelanjaan (Rudianto ; 2009 : 19)

II.2. Basis Data (*Database*)

Database atau basis data adalah sekumpulan data yang memiliki hubungan secara logika dan diatur dengan seusunan tetentu serta disimpan dalam media penyimpanan komputer. dat itu sendiri adalah representasi dari semua fakta yang ada pada dunia nyata. database sering digunakan untuk melakukan proses terhadap data-data tesebut untuk menghasilkan informasi tetentu. misalnya dari data nama siswa yang berulang tahun pada hari ini. Tentu saja informasi tersebut akan anda dapatkan dari sofware pemroses database dengan cara anda memberikan perintah dalam bahasa tertentu yaitu SQL (*Structured Query Language*).

Pada era kemajuan teknologi seperti sekarang ini, nilai informasi sangatlah penting, terlebih bagi kemajuan perusahaan. Oleh karena itu penggunaan dan

penguasaan database sangat penting. Dalam database ada sebutan-sebutan utnuk satuan data yaitu:

1. Karakter, ini adalah satuan data terkecil. data terdiri atau susunan karakter yang pada akhirnya memawakili data yang memiliki arti dari sebuah fakta.
2. *Field*, adalah kumpulan dari karakter yang mewakili fakta tertentu misalnya seperti nama siswa, tanggal lahir, dan lain-lain. Dalam dunia perancangan database, feld juga disebut atribut. Bila dipandang dari sudut pemrogaman berorientasi obyek maka name dan properti tipe. Properti *name* atau nama adalah properti dari *field* yang berisi *field* yang mewakili data sejenis yang disimpannya. Sedangkan properti tipe adalah properti yang mengatur tipe data dari data yang akan ditampungnya. Misalnya nama fieldnya adalah nama siswa maka tipe datanya adalah *char*, bila nama fieldnya adalah tanggal lahir maka tipe datanya adalah *date*. *Field* dilihat seperti kolom.
3. *Record*, adalah kumpulan dari field. Pada record anda dapat menemukan banyak sekali informasi penting dengan cara mengombinasikan field-field yang ada.
4. Tabel, adalah sekumpulan dari record-record yang memiliki kesamaan entity dalam dunia nyata. Kumpulan dari tabel adalah database, wujud fisik sebuah database dalam komputer adalaha sebuah file yang didalamnya terdapat berbagai tingkatan data yang telah disebutkan di atas.
5. File, adalah bentuk fisik dari penyimpanan data. File database berisi semua data yang telah disusun dan diorganisasikan sedemikian rupa sehingga memudahkan pemberian informasi. (Wahana Komputer : 2010 : 24)

II.3. Entity Relationship Diagram

Pada dasarnya ERD (*Entity Relationship Diagram*) adalah sebuah diagram yang secara konseptual memetakan hubungan antar penyimpanan pada diagram DFD di atas. ERD ini digunakan untuk melakukan permodelan terhadap struktur data dan hubungannya. Penggunaannya ERD ini dilakukan untuk mengurangi tingkat kerumitan penyusunan sebuah database yang baik.

Entity dapat berarti sebuah obyek yang dapat dibedakan dengan obyek lainnya. Obyek tersebut dapat memiliki komponen-komponen data (atribut atau field) yang membuatnya dapat dibedakan dari obyek yang lain. Dalam dunia database *entity* memiliki atribut yang menjelaskan karakteristik dari entity tersebut. Ada dua macam atribut yang di kenal deskriptif. Hal ini berarti setiap entity memiliki himpunan yang diperlukan sebuah primary key untuk membedakan anggota-anggota dalam himpunan tersebut.

Atribut dapat memiliki sifat-sifat sebagai berikut:

1. *Atomic*, atomik adalah sifat dari atribut yang menggambarkan bahwa atribut tesebut berisi nilai yang spesifik dan tidak dapat dipecah lagi. Contoh dari sifat atomik adalah field status dari tabel karyawan yang hanya berisi menikah atau single
2. *Multivalued*, sifat ini menandakan atribut ini bisa memiliki lebh dari satu nilai untuk tiap entity tertentu. Misalnya adalah field hobi, hodi dari tiap karyawan mungkin dan hampir pasti lebih dari satu. Misalnya karyawan A mimiliki hobi membaca dan bersepeda.
3. *Composite*, atribut yang bersifat komposit adalah atribut yang nilainya adalah gabungan dari beberapa atribut yang bersifat atomik. Contohnya adalah

atribut alamat yang dapat dipecah menjadi atribut atomik berupa alamt, kode pos, no telepon, dan kota. (Wahana Komputer : 2010 : 30)

Ada beberapa derajat relasi yang terjadi, yaitu :

1. *One to one*, menggambarkan bahwa antara 1 anggota entity A hanya dapat berhubungan dengna 1 anggota entity B. Biasanya derajat relasi ini digambarkan dengna simbol 1-1.
2. *One to many*, menggambarkan bahwa 1 anggota entity A dapat memiliki hubungan dengan lebih dari 1 angota entity B. Biasanya derajat relasi ini digambarkan denga simbol 1-N.
3. *Many to many*, menggambarkan bahwa lebih dari satu anggota A dapat memiliki hubungan dengan lebih dari satu angota entity B. Simbol yang digunakan adalah N-N. (Wahana Komputer : 2010 : 31).

II.3.1. Normalisasi

Setelah melalui tahapan di atas atau ERD, maka hasil pada diagram tersebut mulai direlasasikan pada tabel-tabel database. Untuk itu dperlukan sebuah tahapan yang disebut normalisasi. Normalisasi data adalah proses di mana tabel-tabel pada database dites dalam hal kesalingtergantungan di antara field-field pada sebuah tabel. Misalnya jika pada sebuah tabel terdapat ketergantungan terhadap lebih dari satu field dalam tabel tersebut, maka tabel tersebut harus dipecah menjadi banyak tabel.

Pada proses normaslisasi data, aturan yang dijadikan acuan adalah metode ketergantungan fungsional. Teorinya adalah bahwa tiap kolom dalam sebuah tabel selalu memiliki hubungan yang unik dengan sebuah kolom kunci. Misalnya pada

sebuah tabel data_siswa ada field nomor induk data field nama siswa serta field tanggal lahir. Maka ketergantungan fungsionalnya dapat dinyatakan sebagai berikut: nmr_induk -> nm_siswa dan nmr_induk -> tgl_lahir. Artinya nm_siswa memiliki ketergantungan fungsional terhadap nmr_induk. Field nm_siswa isinya juga ditentukan oleh field nmr_induk. Maksud dari semua itu adalah nmr_induk adalah field kunci yang menentukan karena tidak ada nomor induk yang sama pada satu sekolah, jadi field nmr_induk dapat dijadikan patokan untuk mengisi nm_siswa dan field lainnya.

Ada beberapa langkah dalam normalisasi tabel, yaitu:

1. *Decomposition*, dekomposisi adalah proses mengubah bentuk tabel supaya memenuhi syarat tertentu sebagai tabel yang baik. Dekomposisi dapat dikatakan berhasi jika tabel yang dikenal dekomposisi bila digabungkan kembali dapat menjadi tabel awal sebelum di –dekomposisi. Dekomposisi akan sering dilakukan dalam proses normalisasi untuk memenuhi syarat-syaratnya
2. Bentuk tidak normal, pada bentuk ini semua data yang ada pada tiap entity (diambil atributnya) masih ditampung dalam satu tabel besar. Data yang ada pada tabel ini masih ada yang redundansi dan ada juga yang kosong. Semuanya masih tidak tertata rapi.
3. Normal Form pertama (1^{st} Normal Form), pada tahapan ini tabel di-dekomposisi dari tabel bentuk tidak normal yang kemudian dipisahkan menjadi tabel-tabel kecil yang memiliki kriteria tidak memiliki atribut yang bernilai ganda dan komposit. Semua atribut harus bersifat atomik.

4. Normal Form kedua (2nd Normal Form), pada tahapan ini tabel dianggap memenuhi normal kedua jika pada tabel tersebut semua atribut yang bukan kunci primer bergantung penuh terhadap kunci primer tabel tersebut .
5. Normal Form ketiga (3rd Normal Form), setiap atribut pada tabel selain kunci primer atau kunci utama harus bergantung penuh pada kunci utama. Bentuk normal ketiga biasanya digunakan bila masih ada tabel yang belum efisien. Biasanya penggunaan bentuk normal (normalisasi) hanya sampai pada bentuk ketiga, dan tabel yang dihasilkan telah memiliki kualitas untuk membentuk sebuah database yang dapat diandalkan. Semua tabel diatas juga telah memenuhi bentuk normal tahap ketiga. (Wahana Komputer : 2010 : 32)

II.4. Pemrograman Visual Basic 2010

Visual basic 2010 merupakan salah satu bagian dari produk pemograman terbaru yang dikeluarkan oleh Microsoft, sebagai produk lingkungan pengembangan terintegritasi atau IDE andalan yang dikeluarkan. Visual basic 2010 menambahkan perbaikan – perbaikan filtur dan filtur baru yang lebih lengkap dibandingkan versi visual studio pendahulunya yaitu Vb 2008.

Visual studio merupakan produk pemograman andalan dari Microsoft corporation, yang didalamnya berisi beberapa jenis IDE pemrograman seperti visual basic , visual C ++, visual Web developer, Visual C#, dan Visual f#. semua IDE pemrograman tersebut sudah mendukung penuh implementasi. Net Framework terbaru, yaitu Net Framework 4.0 yang merupakan pengembangan

dari Net Framework 3.5. adapun database standar yang disertakan adalah Microsoft *SQL Server 2008 Express* (Andi Yogyakarta : 2010 : 2).

II.5. SQL Server 2008

SQL Server 2008 adalah sebuah terobosan baru dari Microsoft dalam bidang database. SQL Server adalah sebuah *DBMS (Database Management System)* Yang dibuat oleh *Microsof*t untuk ikut berkecimpung dalam persaingan dunia pengolahan data menyusul pendahulunya seperti IBM dan *Oracle. Sql server 2008* dibuat pada saat kemajuan dalam bidang hardware sedemikian pesat. Oleh karena itu sudah dapat dipastikan bahwa *SQL Server 2008* membawa beberapa terobosan dalam bidang pengolahan dan penyimpanan data (Wahana Komputer ; 2010 : 5).

SQL server 2008 termasuk salah satu mesin database relasional. Ide tentang sistem database relasional pertama kali dicetuskan oleh seorang yang bernama E.F. Codd lewat sebuah artikel yang berjudul " *A Relation Model of Data for Large Shared Data Banks"*. Artikel tersebut ditulis pada tahun 1970. Sistem database relational berdasarkan pada metode matematika. Sistem ini menggantikan metode lama yang berdasarkan *network* dan *hierarki*. Metode relasional ini bekerja berdasarkan keterkaitan antartabel (Wahana Komputer ; 2010 : 25).

II.6. UML

Unified Modeling Language (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membandu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OOP).

Bahasa pemodelan grafis telah ada di industru perangkat lunak sejak lama. Pemicu umum dibalik semuanya adalah bahwa bahasa pemrograman berada pada tingkat abstraksi yang tidak terlalu tinggi untuk memfasilitasi diskusi tentang desain.

UML merupakan standar yang relatif terbuka yang dikontrol oleh *Object Management Group* (OMG), sebuah konsorium terbuka yang terdiri dari banyak perusahaan. OMG dibentuk untuk membuat standar-standar yang mendukung interoperabilitas, khususnya interoperabilitas sistem yang berorientasi objek. OMG mungkin lebih dikenal dengan standar-standar CORBA (*Common Object Request Broker Architecture*).

UML lahir dari penggabungan banyak bahasa permodelan grafis berorientasi objek yang berkembang pesat pada akhir 1980-an dan awal 1990-an. Sejak kehadirannya pada tahun 1997, UML menghancurkan menara Babel tersebut menjadi sejarah. (Prabowo Pudjo Widodo : 2011 : 6-9)

II.6.1. Diagram-Diagram UML

Beberapa *literature* menyebutkan bahwa UML menyediakan Sembilan jenis diagram, yang lain menyebutkan delapan karena ada beberapa

diagram yang digabung menjadi diagram interaksi. Namun model-model itu dapat dikelompokan berdasarkan sifatnya yaitu statis atau dinamis. Jenis diagram itu antara lain:

1. Diagram kelas bersifat statis. Diagram ini memperlihatkan himpuan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi. Diagram ini umumnya dijumpain pada pemodelan system berorientasi objek. Meskipun bersifat statis, sering pula diagram kelas memuat kelas-kelas aktif.

2. Diagram paket (package diagram) bersifat statis. Diagram ini memperlihatkan kumpulan kelas-kelas, merupakan bagian dari diagram komponen.

3. Diagram *use case* bersifat statis. Diagram ini memperlihatkan himpunan use-case dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini sangat penting untuk mengorganisasi dan memodelkan perilaku suatu sistem yang membutuhkan serta diharapkan pengguna.

4. Diagram interaksi dan Squence (urutan) bersifat dinamis. Diagram urutan adalah diagram interaksi yang menekankan pada pengiriman pesan dalam suatu waktu tertentu.

5. Diagram komunikasi *(Communication Diagram)* bersifat dinamis. Diagram sebagai pengganti diagram kolaborasi UML 1.4 yang

menekankan organisasi structural dari objek-objek yang menerima serta mengirim pesan.

6. Diagram *Statechart* bersifat dinamis. Diagram status memperlihatkan keadaan-keadaaan pada sistem, memuat status (state), tranisi, kejadian serta aktifitas. Diagram ini terutama penting untuk memperlihatkan sifat dinamis dari antarmuka *(interface),* kelas, kolaborasi dan terutama penting pada pemodelan sistem-sistem yang reaktif.

7. Diagram aktivitas (*Activity Diagram*) bersifat dinamis. Diagram aktivitas adalah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem. Diagram ini terutama penting dalam pemodelan fungsi-fungsi suatu sistem dan memberi tekanan pada aliran kendali antar objek.

8. Diagram komponen (*component diagram*) bersifat statis. Diagram komponen ini memperlihatkan organisasi ketergantungan sistem/perangkat lunak pada komponen-komponen yang telah ada sebelumnya. Diagram ini berhubungan dengan diagram kelas dimana komponen secara tipikal dipetakan ke dalam satu atau lebih kelas-kelas antarmuka-antarmuka serta kolaborasi-kolaborasi.

9. Diagram deployment *(Deployment diagram*) bersifat statis. Diagram ini memperlihatkan konfigurasi saat alikasi dijalankan (run-time). Memuat simpul-simpul berserta komponen-komponen yang ada di dalamnya. Diagram ini berhubungan erat dengan komponen dimana diagram ini

memuat satu atau lebih komponen-komponen. Diagram ini sangat berguna saat aplikasi kita berlaku sebagai yang dijalankan pada banyak mesin (*Distibuted Computing)*, (Prabowo Pudjo Widodo : 2011 : 10-12).