BAB 2

LANDASAN TEORI

2.1 Teori Umum

2.1.1 Analisis dan Perancangan Sistem

Sebuah sistem, dalam hal ini sistem informasi, didefinisikan oleh Whitten et al. (2001, p8) sebagai sebuah pengaturan dimana manusia, data, proses, penampilan informasi, dan teknologi informasi berinteraksi untuk mendukung dan meningkatkan proses bisnis serta mendukung kebutuhan pengguna akan *problem – solving* dan *decision –making.*

Whitten (2001, p165) menuliskan bahwa analisis sistem merupakan sebuah teknik *problem–solving* yang menguraikan sebuah sistem menjadi komponen-komponen untuk mempelajari seberapa baik kinerja komponen tersebut dan bagaimana interaksinya untuk mencapai tujuan sistem tersebut.

Sedangkan Whitten (2001, p166) mendefinisikan perancangan sistem sebagai sebuah teknik *problem–solving* yang akan menyusun kembali komponen–komponen sistem menjadi sebuah sistem yang utuh, yang diharapkan adalah sebuah sistem yang lebih baik. Proses perancangan ini bisa meliputi penambahan, penghapusan, atau penggantian komponen dari sistem aslinya.

2.1.2 Aplikasi

Kata “aplikasi”, dalam hal ini aplikasi perangkat lunak, didefinisikan Ensiklopedia Wikipedia sebagai *subclass* dari perangkat lunak komputer yang

menggunakan kemampuan komputer secara langsung untuk melakukan suatu perintah yang diinginkan oleh pengguna.

Aplikasi ini berbeda dengan sistem perangkat lunak yang bekerja untuk mengintegrasikan kemampuan yang berbeda–beda dari sebuah komputer, tetapi secara umum tidak mengaplikasikannya secara langsung untuk meningkatkan keuntungan pengguna. Dalam konteks ini, kata “aplikasi” merujuk kepada aplikasi perangkat lunak maupun implementasinya (Anonim 1, 2006).

2.1.3 *Software Development Life Cycle* (SDLC)

Menurut ensiklopedia *online* Wikipedia.org, SDLC adalah proses pengembangan piranti lunak, walaupun SLDC itu sendiri adalah proses yang tidak tergantung pada piranti lunak. SDLC digunakan oleh para *system analyst* untuk mengembangkan sebuah sistem informasi, termasuk kebutuhan, validasi, pelatihan, dan kepemilikan pengguna melalui investigasi, analisis, perancangan, implementasi, dan perawatan. SDLC juga dikenal sebagai pengembangan sistem informasi atau pengembangan aplikasi. Sebuah SDLC harus menghasilkan sistem berkualitas tinggi yang memenuhi atau bahkan melebihi harapan pelanggan dalam hal perkiraan waktu dan harga, dapat bekerja dengan efektif dan efisien dalam infrastruktur teknologi informasi yang ada sekarang atau yang sedang dirancang, serta membutuhkan biaya yang rendah dalam perawatannya dan *cost-effective* dalam pengembangan selanjutnya. (Anonim3, 2007)

Selain itu, SDLC juga merupakan model konseptual yang digunakan dalam manajemen proyek yang mendeskripsikan tahapan-tahapan yang dilakukan dalam proyek pengembangan sistem informasi, mulai dari studi kelayakan pertama

sampai perawatan aplikasi yang telah sempurna. Ada banyak metode yang dikembangkan untuk menuntun jalannya proses pengembangan yang harus dilalui, misalnya *Rapid Application Development (RAD), Joint Application Development (JAD), fountain model, spiral model,* dan *waterfall model* (metode SDLC klasik). (Anonim1, 2007)

Menurut Pressman (1987, pp20-21), paradigma siklus hidup klasik dalam perancangan piranti lunak, yang terkadang disebut sebagai model *waterfall,* menuntut pendekatan yang sistematis dan terurut pada pengembangan piranti lunak yang dimulai pada perancangan sistem dan berlanjut ke analisis, perancangan, *coding, testing,* sampai perawatan sistem. Paradigma siklus hidup *waterfall* ini meliputi aktivitas berikut:

1. *System Analysis and Design*

   Karena piranti lunak selalu merupakan bagian dari sistem yang lebih besar, tahap ini dimulai dengan membuat daftar kebutuhan untuk setiap elemen sistem dan mengalokasikan sebagian subset dari kebutuhan tersebut ke dalam piranti lunak. Tahapan ini sangat penting ketika piranti lunak harus bekerjasama dengan elemen lain seperti piranti keras, masyarakat, dan basis data. Analisis dan perancangan sistem meliputi pengumpulan kebutuhan pada tingkat sistem ditambah dengan sedikit analisis dan perancangan tingkat akhir.

2. *Software Requirements Analysis*

   Proses pengumpulan kebutuhan dilakukan dengan lebih intens dan lebih terfokus pada piranti lunak. Untuk mengetahui sifat dari program yang akan dibuat, perancang piranti lunak (*analyst*) harus

mengerti baik ruang lingkup informasi piranti lunak maupun fungsi-fungsi, fitur, dan antarmuka yang dibutuhkan. Kebutuhan yang harus dipenuhi baik sistem maupun piranti lunak akan didokumentasikan dan kemudian di-*review* bersama-sama dengan pelanggan.

3. *Design*

   Perancangan piranti lunak adalah proses yang terfokus pada 3 atribut yang berbeda dari program: struktur data, arsitektur piranti lunak, dan detil prosedur. Proses perancangan ini menerjemahkan kebutuhan menjadi sebuah representasi piranti lunak yang dapat diuji kualitasnya sebelum proses *coding* dimulai. Sama halnya dengan kebutuhan sistem, hasil perancangan ini juga didokumentasikan dan menjadi bagian dari konfigurasi piranti lunak.

4. *Coding*

   Hasil rancangan dari tahap di atas harus diterjemahkan menjadi bahasa yang dapat dibaca mesin, yang akan dilakukan dalam tahap *coding.* Jika perancangan dilakukan dengan detil, *coding* akan dapat dilakukan secara mudah.

5. *Testing*

   Sesudah *code* dihasilkan, *testing* program dimulai. Proses *testing* difokuskan pada logika piranti lunak, memastikan bahwa setiap pernyataan sudah diuji, dan pada fungsi eksternal, yang dilakukan dengan menjalankan tes untuk memastikan *input* yang dimasukkan akan menghasilkan hasil yang sama dengan hasil yang diinginkan.

6. *Maintenance*

Piranti lunak pasti akan mengalami perubahan setelah dikirimkan ke pelanggan (salah satu pengecualian adalah *embedded software*). Perubahan akan terjadi karena *error* telah ditemukan, karena piranti lunak harus disesuaikan dengan lingkungan eksternal, atau karena pelanggan membutuhkan peningkatan fungsi atau performa. Perawatan sistem mengaplikasikan tahapan siklus hidup piranti lunak yang sebelumnya ke program yang sudah ada, bukan ke program yang baru.

2.1.4 *Structured System Analysis and Design Method* (SSADM)

Menurut Laws (1996), metodologi adalah kumpulan prosedur, teknik, peralatan, dan dokumentasi yang akan membantu para pengembang sistem dalam mengimplementasikan sebuah sistem informasi baru. Sebuah metodologi terdiri atas fase-fase, yang masing-masing terdiri dari banyak subfase, yang akan membimbing para pengembang sistem untuk memilih teknik yang cocok untuk setiap tahapan pengembangan proyek, dan juga membantu mereka untuk merencanakan, memanajemen, mengontrol, dan mengevaluasi proyek sistem informasi. Terdapat 2 metodologi yang sering digunakan dalam proses pengembangan piranti lunak, yaitu *Object Oriented Analysis and Design* (OOAD) dan *Structured System Analysis and Design Method* (SSADM).

Dalam OOAD, pertama-tama sebuah obyek harus didefinisikan. Obyek ini adalah sesuatu yang menjadi ‘pusat’ sistem, yaitu apa yang informasinya harus disimpan. Sebuah obyek dibentuk dengan membuat sebuah struktur data yang mampu menyimpan data yang berkaitan dengan obyek tersebut. Struktur data ini

lalu menyimpan data mengenai sebuah obyek, dan lalu disebut *instance* dari obyek tersebut. Obyek ini lalu dikelilingi dengan segmen-segmen kode yang dikenal sebagai metode, yang melindungi struktur data tersebut dari operasi ilegal dengan cara membuat suatu antarmuka yang mengontrol akses ke obyek tersebut. Tiga prinsip utama dari OOAD adalah *Generalization, Inheritance*, dan *Polymorphism.* Menurut Mason (1999), beberapa keuntungan dari OOAD adalah kemungkinan menggunakan piranti lunak berulang kali dan kemudahan mengubah atau mengembalikan *component library* ke asalnya, struktur kode yang lebih menggambarkan ruang lingkup masalah dengan jelas, dan pengurangan resiko dengan memperkenalkan proses desain formal yang seringkali belum pernah ada sebelumnya. Namun kerugian yang paling jelas terlihat dari pendekatan *Object Oriented* adalah bahwa walaupun *Object Oriented* sangat baik dalam menjelaskan *class* / obyek, *Object Oriented* tidak dapat menjelaskan perilaku sistem secara keseluruhan (Anonim1, 2001).

Unified Modelling Language (UML) sangat efektif digunakan dalam *Object Oriented Languages,* tetapi ada beberapa area yang merupakan kelemahannya, misalnya dalam hubungannya dengan basis data. Pada kasus ini, lebih baik menggunakan ERD seperti yang biasa digunakan dalam SSADM, daripada *Class Diagram,* atau DFD untuk proses yang tidak dapat dijelaskan secara *Object Oriented* (Anonim1, 2003). Martin (2004) menuliskan bahwa UML bukanlah sebuah bahasa pemrograman tetapi sebuah alat dalam OOAD. Yang paling sering digunakan adalah *class diagram* dan *use case diagram.* UML sudah merupakan standar internasional untuk OOAD.

Menurut Downs (1992, p1), SSADM adalah sebuah cara untuk mengorganisir bagian analisa dan perancangan sistem dari sebuah proyek yang bertujuan untuk membuat sebuah sistem informasi berbasis komputer. Metode seperti ini memberikan tuntunan bagi orang-orang yang melakukan hal sebagai berikut:

1. Menentukan apa yang pengguna potensial dari sebuah sistem informasi akan butuhkan dari sistem yang akan dibangun.
2. Merancang sistem informasi yang dibutuhkan.

SSADM terdiri dari aktivitas dan produk. Aktivitas dideskripsikan SSADM dalam 2 cara. Pertama, “kapan” sesuatu harus dilakukan, seperti model struktur; kedua, “bagaimana” sesuatu harus dilakukan, seperti deskripsi teknik. Kumpulan komponen ketiga yang mendefinisikan SSADM adalah produknya, yaitu “apa” yang dibuat oleh SSADM.

Menurut Jadalowen (2007), *Structured Analysis* adalah sekumpulan teknik dan peralatan grafis yang memungkinkan *system analyst* untuk mengembangkan spesifikasi sistem baru yang dapat dimengerti pengguna dengan mudah. Menurut Schumacher (2001), SSADM menspesifikasikan alur dan tugas dari proyek pengembangan piranti lunak dan membuat dokumentasi yang detil dari proyek tersebut. SSADM mengikuti model *waterfall,* yang memungkinkan adanya *review* terhadap tahapan-tahapan sebelumnya tetapi menuntut sebuah tahapan diselesaikan sebelum tahapan selanjutnya bisa dimulai. SSADM lebih terfokus pada tahapan analisa dan perancangan dari SDLC. SSADM menggabungkan 3 metode yang saling melengkapi satu sama lain, yaitu *Logical Data Modelling, Data Flow Modelling,* dan *Entity Event Modelling.*

1. *Logical Data Modelling*

   Dalam proses ini, kebutuhan data dari sistem informasi diinvestigasi, diidentifikasi, digambarkan, dan didokumentasikan. Dan oleh karena itu, terbentuklah *Logical Data Structure* (LDS). LDS member informasi tentang entitas yang harus ada dan hubungan antar entitas tersebut. *Logical Data Modelling* biasanya dilakukan pada 2 tahap awal pengembangan dengan SSADM. Oleh karena itu, dalam tahap uji kelayakan, sebuah *Data Flow Diagram* (DFD) atau *Entity Relationship Diagram* (ERD) biasanya akan terbentuk.

2. *Data Flow Modelling*

   Sementara *Logical Data Modelling* lebih terfokus pada kebutuhan sistem informasi, *Data Flow Modelling* terfokus pada identifikasi, penggambaran, dan dokumentasi alur data dalam sebuah sistem informasi. Dan pada akhirnya, alur data antara proses, penyimpanan data, dan entitas eksternal dapat digambarkan.

3. *Entity Event Modelling*

   Proses *entity event modeling* berkaitan dengan proses bisnis yang mempunyai dampak pada setiap entitas dan lingkungannya. Pada akhirnya, untuk setiap proses, alur data, penyimpanan data, dan entitas eksternal terdapat sebuah *entry* dalam kamus data. Kamus data adalah katalog data utama dalam sebuah sistem informasi yang memperlihatkan struktur, penyimpanan, hubungan, asal, dan kegunaan data. Kamus data ini bertujuan untuk memberi kemudahan

mendeskripsikan data yang sederhana dalam suatu bahasa *verbal* yang pantas.

Keuntungan dari SSADM antara lain:

1. Waktu

   SSADM memungkinkan seseorang untuk merencanakan, mengatur, dan mengontrol sebuah proyek dengan baik. Hal ini penting agar proyek dapat selesai tepat waktu.

2. *Usability*

   Penekanan dalam SSADM terletak pada kebutuhan pelanggan. Pengembangan sistem dan analisa kebutuhan akan kemudian dijalankan secara bersamaan. Keduanya dijalankan untuk melihat apakah sistem dan kebutuhan dapat saling melengkapi.

3. Respon terhadap perubahan dalam lingkungan bisnis

   Karena dokumentasi proyek dilakukan dengan baik dalam SSADM, *issue* seperti tujuan dan kebutuhan bisnis dipertimbangkan dalam pengembangan proyek.

4. Penggunaan kemampuan yang efektif

   SSADM tidak menuntut kemampuan yang spesial dan dapat dengan mudah diajarkan kepada para karyawan.

5. Kualitas yang lebih baik

   SSADM mengurangi tingkat kesalahan dalam sebuah sistem informasi dengan mendefinisikan sebuah tingkatan kualitas pada awal pengembangan proyek dan mengecek sistem secara berkala.

6. Peningkatan produktivitas

   Dengan memungkinkan pengiriman tepat waktu, pemenuhan kebutuhan bisnis, memastikan kualitas yang lebih baik, menggunakan sumber daya manusia dengan efektif sekaligus menghindari birokrasi, SSADM meningkatkan produktivitas keseluruhan baik untuk proyek dan perusahaan.

7. Mengurangi biaya

   SSADM memisahkan desain sistem fisikal dan logikal. Oleh karena itu, sistem tersebut tidak memerlukan piranti keras ataupun piranti lunak baru dalam implementasinya.

Jadalowen (2007) juga menulis bahwa SSADM baik untuk digunakan pada sistem *real-time* atau sistem transaksi. SSADM juga disarankan untuk mengembangkan proyek kontrak yang mana kebutuhan sistemnya telah diketahui dengan jelas, dan juga untuk masalah yang ruang lingkupnya sudah diketahui dengan jelas.

2.2 Teori Khusus

2.2.1 E–business

Chaffey et al. (2000, p6) menuliskan bahwa sebutan *e–business* merujuk kepada semua penggunaan teknologi elektronik yang terhubung ke Internet dalam fungsi bisnis, mulai dari *Human Resource Management* (HRM) sampai strategi perusahaan.

Belakangan ini istilah *e-commerce* banyak disebut, dan terkadang menjadi rancu dengan pengertian *e-business.* Banyak pihak yang menganggap

bahwa *e-commerce* itu sama dengan *e-business,* dan beberapa pihak lain mengatakan bahwa *e-commerce* adalah bagian dari *e-business.* Strauss et al. (2001, p6) menuliskan bahwa EB = EC + BI + CRM + SCM + ERP. Artinya adalah bahwa *e-business* terdiri dari beberapa bagian, yaitu *e-commerce, Business Intelligence, Customer Relationship Management, Supply Chain Management,* dan *Enterprise Resource Planning.* Dari definisi ini sudah jelas terlihat bahwa *e-commerce* tidak sama dengan *e-business* dan bahwa *e-commerce* merupakan bagian dari *e-business.*

Menurut Chaffey et al. (2000, p325), kata *commerce* didefinisikan New Oxford English Dictionary sebagai "aktivitas membeli dan menjual, khususnya dalam skala besar". Aktivitas jual–beli ini bisa terjadi mulai dari sebuah pembelian di sebuah supermarket lokal sampai ke pembelian antar perusahaan internasional.

Dari beberapa definisi tersebut, bisa disimpulkan bahwa *e-commerce* merupakan aktivitas jual–beli melalui Internet, sedangkan *e-business* adalah segala proses bisnis yang dilakukan di Internet, termasuk proses jual–beli dan pemasaran.

2.2.2 E–marketing

Strauss et al. (2001, p8) menuliskan bahwa *marketing* atau pemasaran adalah proses merencanakan dan melakukan konsep, distribusi, promosi, dan penentuan harga sebuah ide, barang, atau jasa untuk menciptakan sebuah proses pertukaran yang memenuhi kebutuhan individual dan organisasi. Sedangkan menurut Chaffey et al. (2001, p5) kata *marketing* meliputi seluruh proses dan kegiatan organisasi yang dilakukan dengan tujuan untuk menemukan kebutuhan

target pasar dan memberi produk serta jasa kepada konsumen dan para pelaku bisnis lain, seperti karyawan dan institusi keuangan.

Dalam konteks pemasaran, dikenal suatu istilah *Marketing Mix*. *Marketing Mix* ini telah diterima secara luas sebagai 4P yang menjelaskan posisi produk di dalam pasar. *Marketing Mix* ini meliputi:

1. *Product*

   Product adalah suatu benda atau jasa yang diproduksi secara massal atau diproduksi pabrik dengan skala besar dengan volume atau unit yang spesifik.

2. *Price*

   Price adalah harga yang harus dibayar konsumen untuk sebuah produk. Harga ditentukan oleh beberapa faktor, termasuk pangsa pasar, kompetisi, modal awal, identitas produk, dan *value* produk di mata konsumen.

3. *Place*

   Place merepresentasikan tempat dimana produk dapat dibeli. Seringkali disebut sebagai jalur distribusi. *Place* dapat meliputi semua toko fisik maupun yang sifatnya *virtual* seperti toko di Internet.

4. *Promotion*

   Promotion merepresentasikan semua bentuk komunikasi yang digunakan oleh *marketer* di dalam pasar. *Promotion* memiliki empat elemen, yaitu *advertising, public relations, word of mouth,* dan *point of sale. Advertising* meliputi semua bentuk komunikasi yang dibayar, mulai dari iklan televisi dan bioskop, radio, Internet, sampai

pembuatan *billboard. Public relations* adalah komunikasi yang tidak langsung dibayar, termasuk konferensi pers, kerjasama dengan sponsor, pameran, seminar, atau *event* perusahaan. *Word of mouth* adalah komunikasi informal mengenai sebuah produk oleh individu tertentu, atau konsumen yang puas terhadap produk.

(Anonim4, 2007)

Sedangkan *e-marketing,* menurut Chaffey et al. (2001, p6) adalah penggunaan Internet dan teknologi digital yang terkait untuk mencapai tujuan pemasaran dan mendukung konsep pemasaran modern. Strauss et al. (2001, p8) juga menuliskan bahwa *e-marketing* mempengaruhi konsep pemasaran tradisional dengan 2 cara:

1. Meningkatkan efisiensi dari fungsi pemasaran yang dilakukan secara tradisional
2. Teknologi yang digunakan di *e-marketing* merubah banyak strategi pemasaran. Perubahan ini menghasilkan suatu model bisnis baru yang menambah *customer value* dan / atau meningkatkan keuntungan perusahaan.

Menurut Chaffey (2001, pp5-7), Internet dapat diaplikasikan sebagai bagian penting dalam konsep pemasaran modern karena:

1. Dapat mendukung seluruh proses dan fungsi organisasi yang memberi produk atau jasa kepada konsumen dan pelaku bisnis lain.

2. Merupakan media komunikasi yang dapat mengintegrasikan bagian-bagian fungsional yang berbeda-beda dari organisasi tersebut.
3. Memfasilitasi manajemen informasi, yang telah dianggap sebagai alat bantu perancangan dan implementasi strategi pemasaran yang penting.
4. Peran Internet di masa yang akan datang akan mewujudkan sebagian visi perusahaan karena dampak Internet di masa yang akan datang akan membawa perubahan yang signifikan pada banyak bisnis.

Selain itu Internet dipandang baik untuk tujuan pemasaran karena bisa menjangkau 4 posisi strategis berikut:

1. Menembus pasar

   Internet dapat digunakan untuk menjual lebih banyak produk ke dalam pasar yang sudah ada. Tujuan ini dapat dicapai dengan cara menggunakan Internet untuk memasang iklan, sehingga meningkatkan *customer awareness* terhadap produk dan profil perusahaan.

2. Pengembangan pasar

   Menjual ke pasar yang baru dengan mengambil keuntungan dari rendahnya biaya untuk memasang iklan secara internasional tanpa keharusan untuk mempunyai infrastruktur penjualan di negara pelanggan.

3. Pengembangan produk

Produk atau jasa baru dikembangkan, untuk kemudian dapat disajikan melalui Internet.

4. Diversifikasi

Produk baru dikembangkan, untuk kemudian dijual ke dalam pasar yang baru.

2.2.3 *Build–to–Order* (BTO)

Situs ensiklopedia *online*, Wikipedia.org, mendefinisikan kata *Build–to–Order* (BTO) sebagai sebuah pendekatan produksi dimana produk dibuat setelah pesanan diterima. BTO sendiri dipandang baik untuk produk yang konfigurasinya rumit, semisal server komputer, atau untuk produk yang membutuhkan biaya banyak dalam penyimpanannya (Anonim 2, 2007).

Vigoroso (2001) dalam artikelnya menulis bahwa dengan BTO, perusahaan akan mengurangi tingkat stok barang dan biaya retur produk, serta menambah kepuasan pelanggan yang tentunya akan mengulangi pesanannya seiring dengan berjalannya waktu.

2.2.4 Sistem Basis Data

2.2.4.1 Pengertian Basis Data

Menurut Conolly (2002, p14), basis data atau *database* adalah sebuah kumpulan data yang terhubung dan deskripsi data tersebut, yang dirancang untuk memenuhi kebutuhan informasi oleh sebuah organisasi.

Sistem basis data ini dapat menutupi kekurangan dari penyimpanan data berdasarkan file, antara lain:

1. Definisi data tersimpan di program aplikasi, bukan disimpan secara terpisah.
2. Tidak ada kontrol akan akses dan manipulasi data selain yang ditentukan oleh program aplikasi tersebut.

2.2.4.2 *Database Management System* (DBMS)

Sedangkan sebuah DBMS, menurut Conolly (2002, p16) adalah sebuah sistem software yang memungkinkan pengguna untuk mendefinisikan, membuat, menjaga, dan mengontrol akses ke database. Sebuah DBMS pada umumnya menyediakan fasilitas berikut ini:

1. Memungkinkan pengguna untuk mendefinisikan database, yang biasanya dilakukan menggunakan *Data Definition Language* (DDL).
2. Memungkinkan pengguna memasukkan, merubah, menghapus, dan memanggil data dari database, yang biasanya dilakukan menggunakan *Data Manipulation Language* (DML).
3. Menyediakan akses yang terkontrol ke database.

2.2.4.3 *Structured Query Language* (SQL)

SQL , menurut Ullman (2005, p125) adalah sekumpulan kata-kata yang digunakan hanya untuk berinteraksi dengan database. Semua aplikasi database pada umumnya menggunakan SQL, termasuk MySQL.

SQL diciptakan tidak lama sesudah Dr. E. F. Codd membuat suatu teori *relational database* pada awal tahun 1970an. Pada 1989, American National Standards Institute, organisasi yang bertugas menstandarisasi bahasa, merilis standar SQL pertama, yang saat ini dikenal dengan SQL89. SQL92 (disebut juga SQL92 atau hanya SQL) dirilis pada tahun 1992 dan menjadi versi yang digunakan sampai saat ini.

MySQL adalah aplikasi database *open source* yang paling dikenal di seluruh dunia (berdasarkan *website* MySQL) dan biasanya digunakan bersamaan dengan PHP.

2.2.4.4 *Entity Relationship Diagram* (ERD)

Menurut Conolly (2002,p330), *Entity Relationship Modelling* adalah suatu pendekatan perancangan basis data yang mendefinisikan suatu data penting yang disebut entitas, dan hubungan antara data yang harus direpresentasikan di dalam model tersebut.

Sedangkan ERD menurut situs ensiklopedia online, Wikipedia.org adalah hasil akhir dari proses *Entity Relationship Modelling* dan merupakan suatu tipe model data konseptual atau model data semantik. Menurut Conolly (2002, p359), *Entity Relationship Model* yang ditambahkan dengan konsep semantic disebut *Enhanced Entity Relationship (EER) Model*.

2.2.5 *State Transition Diagram* (STD)

Wiggers (1999) menyebutkan bahwa STD adalah sebuah cara untuk mendeskripsikan perilaku sistem yang bergantung pada waktu. Aturan

konsistensinya adalah: “Perilaku sistem dalam keadaan apapun harus sama dan tidak tergantung dari jalur mana keadaan tersebut diperoleh.”

Dalam situsnya, Wiggers (1999) juga menjelaskan bahwa keadaan sistem adalah:

1. Keadaan sistem adalah mode perilaku sistem yang dapat diamati.
2. Pada suatu waktu, sebuah STD hanya bisa berada pada satu keadaan.
3. Perilaku sebuah sistem dapat dideskripsikan dengan lebih dari 1 STD.

Dan kondisi transisi adalah kejadian internal (*internal events*) atau eksternal sistem. Sedangkan aksi transisi adalah:

1. Aksi respons terhadap kejadian (*events*).
2. Memulai sebuah aksi yang sifatnya sekali jalan.
3. Menghubungkan beberapa STD yang berbeda.
4. Memproduksi kontrol output.

Berikut ini adalah tahapan menggambar STD:

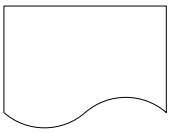
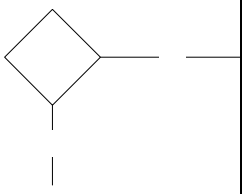
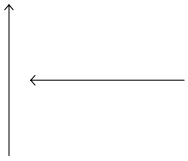
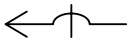
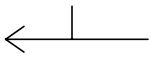
1. Mengidentifikasi keadaan yang dapat diamati dari sebuah sistem.
2. Memilih keadaan dengan perilaku normal.
3. Menspesifikasi ketentuan yang menandakan terjadinya sebuah transisi.
4. Menspesifikasi aksi yang dibutuhkan untuk memproduksi perilaku yang dapat diamati dalam keadaan yang dituju untuk setiap transisi.
5. Jika sistem tersebut kompleks, bagilah diagram menjadi beberapa STD.

2.2.6 Diagram Aliran Dokumen (DAD)

Menurut Mulyadi (2001,pp58-63), diagram aliran dokumen adalah suatu model yang menggambarkan aliran dokumen dan proses untuk mengolah dokumen dalam suatu proses.

Berikut ini adalah tabel yang menjelaskan komponen-komponen dari diagram aliran dokumen :

Tabel 2.1 Tabel Simbol-simbol Diagram Aliran Dokumen

Simbol	Keterangan
	**Dokumen** Simbol ini digunakan untuk menggambarkan semua jenis dokumen, yang merupakan formulir untuk merekam data terjadinya suatu transaksi.
	**Keputusan** Simbol ini menggambarkan keputusan yang harus dibuat dalam proses pengolahan data. Keputusan yang dibuat ditulis dalam simbol.
	**Garis Alir** Simbol ini menggambarkan arah proses pengolahan data.
	**Persimpangan Garis Alir** Jika dua garis alir bersimpangan, untuk menunjukkan arah masing-masing garis, salah satu garis dibuat sedikit melengkung tepat pada persimpangan kedua garis tersebut.
	**Pertemuan Garis Alir** Simbol ini digunakan jika dua garis alir bertemu dan salah satu garis mengikuti garis lainnya.

Simbol	Keterangan
	**Proses** Simbol ini untuk menunjukkan tempat-tempat dalam sistem informasi yang mengolah atau mengubah data yang diterima menjadi data yang mengalir keluar. Nama pengolahan data ditulis didalam simbol.
	**Mulai / Berakhir (terminal)** Simbol ini untuk menggambarkan awal dan akhir suatu sistem akuntansi