BAB III
LANDASAN TEORI

3. 1 Inventaris

Ada bermacam-macam gagasan mengenai inventaris. Salah satu pemikiran itu adalah bahwa memiliki inventaris merupakan suatu keharusan dalam suatu perusahaan, karena biaya untuk tidak memiliki suatu inventaris pada saat perusahaan menginginkannya, lebih tinggi dari biaya yang ditanggung untuk kepemilikan inventaris itu sendiri.

Ada beberapa macam klasifikasi inventori, menurut buku yang ditulis oleh Dobler et al (1990), klasifikasi inventori yang digunakan oleh perusahaan, antara lain :

a) Inventori Produksi

Inventori yang termasuk dalam klasifikasi inventori produksi adalah bahan baku dan bahan-bahan lain yang digunakan adalam proses produksi dan merupakan bagian dari produk. Bisa terdiri dari dia tipe yaitu *item* spesial yang dibuat khusus untuk spesifikasi perusahaan dan item standart produksi yang dibeli secara *off-the-self*.

b) Inventori MRO (*Maintenance, Repair, and Operating supplies)*

Inventori yang termasuk dalam katagori ini adalah barang – barang yang digunakan dalam proses produksi namum tidak merupakan bagian dari produk. Seperti pelumas dan pembersih.

c) Inventori *In-Process*

Inventori yang termasuk dalam katagori inventori ini adalah produk setengah jadi.Produk yang termasuk dalam katagori inventori ini bisa ditemukan dalam berbagai proses produksi.

d) Inventori *Finished-goods*

Semua produk jadi yang siap dipasarkan termasuk dalam katagori inventori *finished goods*. PT. ABC adalah sebuah swalayan yang menjual produk – produk yang siap untuk dipakai. Tidak ada proses pengolahan yang ada disana, sehingga semua inventori yang dimilikinya termasuk dalam kategori ini.

Inventaris ada untuk memungkinkan perusahaan memenuhi kebutuhan konsumen. Inventaris biasanya juga ada untuk memperlancar arus barang melalui proses produksi khususnya bagi pusat pekerjaan yang mempunyai ketergantungan. Alasan utama kehadirannya adalah perlindungan terhadap ketidakpastian pemasok. Keberadaann inventaris juga memungkinkan pemanfaatan realistis dan sebesar – besarnya dari perlengkapan dan tenaga kerja.

Biaya untuk menyimpan inventaris mencakup biaya unit bahan, biaya pemesanan atau pemesanan ulang, dan biaya penyimpanan. Bilamana perusahaan menghasilkan bahan yang dibutuhkan untuk membuat produk, biaya pemesanan ulang diganti dengan biaya penyiapan mesin atau melakukan perubahan kegiatan. Biaya penyimpanan pada umumnya mencakup biaya gudang, asuransi kebakaran dan pencurian, serta administrasi gudang. Biaya tidak berwujud yang berkaitan dengan penyimpanan inventaris adalah kerugian peluang yang berhubungan dengan inventaris yang kalau tidak bisa dikeluarkan untuk usaha yang lebih menguntungkan.

3.2 Manajemen Inventaris

Manajemen inventaris berhubungan dengan perencanaan dan pengendalian inventaris. Perencanaan inventaris mencoba untuk mencari jawaban atas dua pertanyaan dasar:

a) Kapan memesan?

Pertanyaan ini berhubungan dengan konsep pemesanan ulang. Ini merupakan sistem dimana setiap bahan yang digunakan secara teratur dipesan ulang kalau persediaan berkurang sampai tingkat tertentu. Tingkat tersebut biasanya merupakan fungsi dari waktu penyelesaian proses, permintaan harian, dan stok yang aman.

b) Berapa banyak yang harus dipesan?

Kuantitas yang dipesan ditentukan melalui kuantitas pesanan ekonomis. Kebijakan inventaris perusahaan yang menggunakan model pesanan dalam jumlah tetap adalah dengan memesan kuantitas standart kalau titik untuk pemesanan ulang tercapai tanpa menghiraukan kapan ini terjadi. Ini dipicu kejadian dan

bergantung kepada permintaan atas barang-barang tersebut. Model ini dapat diterapkan untuk:

a) Barang-barang yang tidak begitu mahal dan tidak begitu penting.
b) *Vendor* atau pembeli bisa mendapatkan pesanan baru jika mereka melakukan kunjungan teratur atau rutin ke para konsumen.
c) *Vendor* atau pembeli akan menggabungkan pesanan untuk mengurangi biaya pemesanan dan pengangkutan.

Sistem pengendalian inventaris dirancang untuk memantau tingkat inventaris dan merancang sistem dan prosedur bagi pengelolaan inventaris secara efektif. Dalam pengadaan sistem untuk mengelola inventaris, ada dua wilayah keputusan penting yaitu penggolongan inventaris dan ketepatan pencatatan inventaris.

Strategi pengendalian inventaris mencakup sebagai berikut :

a) *Analysis ABC*

Ini merupakan teknik yang menggolongkan inventaris perusahaan menurut tiga golongan berdasarkan volume dolar tahunan.

b) *Volume dolar* tahunan dihitung sebagai berikut *:*

ADV = permintaan tahunan atas setiap barang inventaris x biaya per unit. Berdasarkan ADV maka semua barang inventaris dapat digolongkan sebagai berikut :

Penggolongan	**Keterangan**
Kelas A	ADV tinggi biasanya mewakili sekitar 15% dari jumlah barang inventaris, tetapi mencapai sekitar 75 – 80% dari jumlah biaya inventaris.
Kelas B	ADV cukup tinggi yaitu mewakili sekitar 30% dari jumlah barang ,tetapi 15 – 25% dari seluruh nilai.
Kelas C	ADV adalah rendah, mewakili sekitar 55% semua barang , tetapi hanya 5% dari seluruh nilai.

Dalam penjadwalan *supply* barang, hal yang paling penting adalah kecepatan pengiriman barang dan respon yang cepat. Penjadwalan akan kebutuhan barang adalah alternative untuk mengetahui kebutuhan dari inventaris. Teknik yang popular dalam mengatasi mekanisme proses penjadwalan adalah *supply* barang akan tersedia disaat dibutuhkan untuk melakukan suatu kegiatan (Ronald H.Ballou, :427).

3.3 *Data Flow* Diagram

Menurut Jeffrey (2011:230) *data flow* diagram (DFD) merupakan *tools* yang serbaguna. DFD memiliki empat simbol umum yang sering digunakan untuk mewakili sistem informasi. Keuntungan menggunakan *data flow* diagram adalah mudah digunakan, karena hanya melibatkan empat symbol yang berbeda, sehingga mempermudah pemakai yang kurang menguasai bidang komputer untuk mengerti sistem yang akan dikerjakan atau dikembangkan.

Beberapa hal yang perlu mendapat perhatian lebih tentang *data flow* diagram adalah sebagai berikut :

1. Antara sumber data tidak boleh langsung saling berhubungan.
2. Diperbolehkan untuk mengambil sumber data yang sama, dengan tujuan untuk menyederhanakan permodelan.
3. Hindari dialog – dialog yang tidak perlu dalam *data flow* diagram.

Untuk memudahkan membaca DFD, maka penggambaran DFD disusun berdasarkan tingkatan atau level dari atas ke bawah, yaitu :

a. Diagram Konteks

Merupakan diagram paling atas yang terdiri dari suatu proses dan menggambarkan ruang lingkup proses. Hal yang digambarkan dalam diagram konteks adalah hubungan *terminator* dengan sistem dan juga sistem dalam proses. Sedangkan hal yang tidak digambarkan dalam diagram konteks adalah hubungan terminator dan data *store*.

b. Diagram *Zero* (level 0)

Merupakan diagram yang berada diantara diagram konteks dan diagram detail serta menggambarkan proses utama dari DFD. Hal ini yang digambarkan

dalam diagram *zero* adalah proses utama dari sistem serta hubungan *entity*, proses, alur data, dan data *store*.

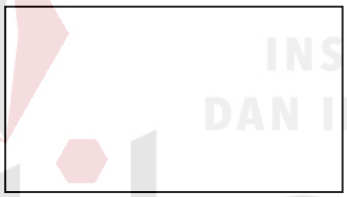
c. Diagram Detail

Merupakan penguraian dalam proses yang ada didalam diagram *zero*. Diagram yang paling rendah dan tidak dapat diuraikan lagi.

Data Flow Diagram (DFD) memiliki empat komponen, yaitu :

a) *Terminator* atau *External Entiry* atau Kesatuan Luar

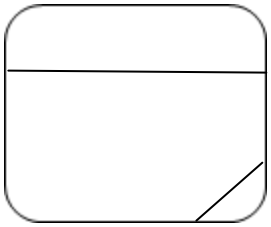
Terminator mewakili entitas *external* yang berkomunikasi dengan sistem yang sedang dikembangkan. *Terminator* merupakan kesatuan di lingkungan sistem. Terminator dapat berupa orang, organisasi atau sistem lainnya yang berada di lingkungan luar sistem. Biasanya *terminator* ini dikenal dengan nama entitas (*external*), sumber atau tujuan (*source* and *sink*). *Terminator* dapat juga berupa departemen, divisi, atau sistem diluar sistem yang berkomunikasi dengan sistem yang dikembangkan, seperti yang terlihat pada gambar 3.1.



Gambar 3.1 Simbol Eksternal Entity

b) Proses

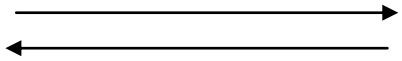
Proses sering dikenal dengan nama *Bubble*, fungsi atau informasi. Komponen proses menggambarkan bagian dari sistem, yang mentransformasikan *input* ke *output*, atau dapat dikatakan bahwa komponen proses menggambarkan transformasi satu *input* atau lebih menjadi *output*. Dilambangkan dengan lingkaran atau empat persegi panjang tegak dengan sudut tumpul, seperti yang terlihat pada gambar 3.2



Gambar 3.2 Simbol Proses

c) Data *Flow*

Data flow menunjukkan arus dari data yang berupa masukan untuk sistem atau hasil dari proses sistem dan dapat berbentuk seperti gambar 3. berikut ini.



Gambar 3.3 Simbol Data flow

d) Data *Store* (Penyimpanan Data)

Data *store* digunakan sebagai saran untuk pengumpulan data. Data *store* disimbolkan dengan dua garis horizontal yang pararel dimana tertutup pada salah satu ujungnya. Suatu nama perlu diberikan pada data *store* menunjukan nama dari *file*nya. Data *store* ini biasanya berkaitan dengan penyimpanan seperti : *file* atau database yang berkaitan dengan penyimpanan secara komputerisasi. Data *store* juga berkaitan dengan penyimpanan data.

Gambar 3.4 Simbol Data source

3.4 Entity Relationship Diagram

Entiti relasi adalah suatu alat untuk mempresentasikan model data yang ada pada suatu sistem dimana terdapat *entity* dan *relationship*. *Entity* merupakan objek yang ada dan terdefinisikan di dalam suatu organisasi dapat abstrak atau nyata, misal dapat berupa orang, objek atau waktu kejadian. Setiap *entity* mepunyai atribut atau karakteristik *entity* tersebut. Adapun elemen – elemen dari ERD ini adalah :

1. Entitas adalah sesuatu yang dapat diidentifikasi di dalam lingkup pemakai, sesuatu yang penting bagi pemakai dari sistem yang akan dikembangkan.
2. Atribut, entitas memiliki atribut yang berfungsi untuk menjelaskan karakteristik dari entitas.

3. Pengidentifikasian, data – data entitas memiliki nama yang berfungsi untuk mengidentifikasi mereka. Sebuah identifikasi dapat bersifat unik atau tidak unik.
4. Hubungan atau relasi berfungsi menunjukan hubungan satu entitas daengan entitas yang lain. Hubungan ini boleh memiliki atribut. Banyaknya entitas dalam suatu relasi menunjukan tingkat dari relasi yang bersangkutan, namun yang banyak digunakan dalam aplikasi – aplikasi adalah model yang menggunakan relasi tingkat dua atau yang disebut dengan hubungan *biner*. Hubungan *biner* ini memiliki tiga tipe yaitu hubungan *biner* dengan satu, *biner* satu ke banyak dan hubungan *biner* banyak ke banyak.

Sedangkan *relationship* adalah hubungan yang mewujudkan pemetaan antar *entity*. Fungsi untuk hubungan yang mewujudkan pemetaan antar *entity*. Jenis *relationship* diagram dapat berbentuk :

a. One to One

Yaitu relasi satu lawan satu yang terjadi bila satu *record* yang ada didalam satu *entity* atau *table* hanya punya satu relasi pada *file* lain. Misalnya suatu departemen hanya mengerjakan satu jenis pekerjaan saja dan satu pekerjaan hanya dikerjakan oleh satu departemen saja.

b. One to Many

Yaitu relasi satu lawan banyak yang terjadi bila *record* dengan kunci tertentu pada satu *file* mempunyai relasi banyak *record* pada *file* lain. Misalnya suatu pekerjaan hanya dikerjakan oleh satu departemen saja, namun suatu departemen dapat mengerjakan beberapa macam pekerjaan sekaligus.

c. Many to Many

Yaitu relasi banyak lawan banyak yang terjadi bila kedua *file* saling mempunyai relasi banyak *record* pada *file* lain. Misalnya satu departemen mampu mengerjakan banyak pekerjaan, juga satu pekerjaan dapat ditangani oleh banyak departemen.

3.5 Siklus Hidup Pengembang sistem

Siklus hidup pengembang sistem atau *software Development System Life Cycle* (SDLC) adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan mengguanakan model-model dan metodologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya (berdasarkan *best practice* atau cara-cara yang sudah teruji baik).(Paul Bocij, 2008 : 286)

3.6 Tahapan SDLC

3.6.1 Elisitasi Kebutuhan

Elisitasi atau pengumpulan kebutuhan merupakan aktivitas awal dalam proses rekayasa perangkat kebutuhan. Sebelum kebutuhan dapat dianalisis, dimodelkan, atau ditetapkan, kebutuhan harus dikumpulkan melalui proses elisitasi. Elisitasi kebutuhan adalah sekumpulan aktivitas yang ditujukan untuk menemukan kebutuhan suatu sistem melalui komunikasi dengan pelanggan, pengguna sistem dan pihak lain yang memiliki kepentingan dalam pengembangan sistem.

Sejalan dengan proses rekayasa kebutuhan secara keseluruhan, elisitasi kebutuhan bertujuan untuk :

a. Mengetahui masalah apa saja yang perlu dipecahkan dan mengenali batasan-batasan sistem. Proses-proses dalam pengembangan perangkat lunak sangat ditentukan oleh seberapa dalam dan luas pengetahuan developer tentang permasalahan.

b. Mengenali siapa saja para *stakeholder*, yaitu setiap pihak yang memiliki kepentingan terhadap sesuatu, dimana dalam konteks perangkat lunak adalah proyek pengembangan perangkat lunak itu sendiri, beberapa yang dapat dikatakan sebagai *stakeholder* antara lain adalah konsumen atau klien yang membayar sistem, pengembang yang merancang, membangun, dan merawat sistem, dan pengguna yang berinteraksi dengan sistem untuk mendapatkan hasil kerja mereka.

c. Mengenali tujuan dari sistem yaitu sasaran-sasaran yang harus dicapai. Tujuan merupakan sasaran sistem yang harus dipenuhi, penggalian *high level goals* di awal proses pengembangan sangatlah penting karena brtujuan lebih terfokus pada ranah masalah dan kebutuhan *stakeholder* dari pada solusi yang dimungkinkan untuk masalah tersebut.

3.6.2 Analisis Sistem

Analisis system adalah sebuah teknik pemecahan masalah yang menguraikan sebuah system menjadi bagian-bagian komponen dengan tujuan mempelajari seberapa bagus bagian-bagian komponen tersebut bekerja dan berinteraksi untuk meraih tujuan mereka.(Jeffery, 2004 : 176)

3.6.3 Langkah-langkah analisis sistem

Menurut Jogianto (2001: 178) langkah-langkah di dalam tahap analisis sistem hampir sama dengan langkah-langkah yang dilakukan dalam mendefinisikan proyek-proyek sistem yang akan dikembangkan di tahap perencanaan sistem.

Di dalam tahap analisis sistem terdapat langkah-langkah dasar yang harus dilakukan oleh analis sistem sebagai berikut ini:

1. *Identify*, yaitu mengidentifikasi masalah
2. *Understand*, yaitu memahami kerja dari sistem yang ada
3. *Analyze*, yaitu menganalisis sistem
4. *Report*, yaitu membuat laporan hasil analisis

3.6.4 Mengidentifikasi masalah dan analisis

Merupakan langkah pertama yang dilakukan dalam tahap analisis sistem. Masalah dapat didefinisikan sebagai suatu pertanyaan yang diinginkan untuk dipecahkan. Tugas-tugas yang harus dilakukannya adalah sebagai berikut ini.

a) Mengidentifikasi penyebab masalah.
b) Mengidentifikasi titik keputusan.
c) Mengidentifikasi personil-personil kunci

3.6.5 Memahami kerja dari sistem yang ada

Langkah ke dua dari tahap analisis sistem adalah memahami kerja dari sistem yang ada. Langkah ini dapat di lakukan dengan mempelajari secara terinci bagaimana sistem yang ada beroperasi. Untuk mempelajari operasi dari sistem ini diperlukan data yang dapat diperoleh dengan cara melakukan penelitian.

3.6.6 Menganalisis Hasil Penelitian

Langkah ini dilakukan berdasarkan data yang telah diperoleh dari hasil penelitian yang telah dilakukan. Menganalisis hasil penelitian sering sulit dilakukan oleh analis sistem yang masih baru. Pengalaman menunjukkan bahwa banyak analis sistem yang masih baru mencoba untuk memecahkan masalah tanpa menganalisisnya.

3.6.7 Membuat Laporan Hasil Analisis

Setelah proses analisis sistem ini selesai dilakukan, tugas berikutnya dari analis sistem dan teamnya adalah membuat laporan hasil analisis. Laporan ini diserahkan kepada *steering commitee* yang nantinya akan diteruskan ke manajemen. Tujuan utama dari penyerahan laporan ini kepada manajemen adalah:

a) Pelaporan bahwa analisis telah selesai dilakukan;

b) Meluruskan kesalah-pengertian mengenai apa yang telah ditemukan dan dianalisis oleh analis sistem tetapi tidak sesuai menurut manajemen;

c) Meminta pendapat-pendapat dan saran-saran dari pihak manajemen;

Meminta persetujuan kepada pihak manajemen untuk melakukan tindakan selanjutnya (dapat berupa meneruskan ke tahap desain sistem atau mengehntikan proyek bila dipandang tidak layak lagi) (Jogiyanto, 2005 : 130- 149).