BAB II

LANDASAN TEORI

2.1. Pengenalan Manajemen Persediaan

Manajemen persediaan merupakan salah satu aset yang paling mahal, mencerminkan sebanyak 40% dari total modal yang diinvestasikan oleh CV. Prima Dua Jaya. Perusahaan ini dapat mengurangi biaya dengan cara menurunkan tingkat persediaan ditangan. Di pihak lain, konsumen akan merasa tidak puas bila suatu produk stoknya habis. Oleh karena itu, perusahaan harus mencapai keseimbangan diantara keduanya.

Perusahaan mempertahankan 4 jenis persediaan : (1) Persediaan bahan mentah, (2) Persediaan barang dalam proses (*Work in proses*-WIP), Persediaan MRO (perlengkapan pemeliharaan/perbaikan/operasi), dan (4) persediaan barang jadi.

Persediaan bahan mentah telah dibeli seperti besi dan plat, namun belum di proses. Bahan mentahnya dapat digunakan dari proses produksi utuk pemasok yang berbeda-beda. Meskipun demikian, pendekatan yang lebih disukai adalah dengan menghapus variabilitas pemasok dalam hal mutu, jumlah atau pengiriman sehingga tidak diperlukan pemisahan. Persediaan barang-barang dalam proses telah mengalami beberapa perubahan, tetapi belum selesai. WIP ini ada karena untuk membuat produk memerlukan waktu (disebut waktu siklus). Pengurangan waktu siklus menyebabkan persediaan WIP pun berkurang. Sering kali hal ini tidak sulit untuk dilakukan, karena hampir disepanjang waktu “pembuatan produk” , produk itu sebenarnya menganggur.

Waktu kerja aktual atau waktu “jalan” merupakan bagian kecil dari waktu sarus bahan baku, mungkin hanya 5%. MRO merupakan persediaan yang dikhususkan untuk perlengkapan pemeliharaan/perbaikan/operasi. MRO ini ada karena waktu dan kebutuhan untuk pemeliharaan dan perbaikan dari beberapa peralatan tidak dapat diketahui. Walaupun permintaan untuk persediaan MRO ini sering kali merupakan fungsi jadwal-jadwal pemeliharaan, permintaan MRO lainnya perlu diantisipasi. Demikian pula, persediaan barang jadi selesai dan menunggu untuk dikirimkan. Barang jadi dimasukkan ke dalam persediaan karena permintaan konsumen untuk jangka waktu tertentu mungkin tidak diketahui.

2.2 Teknik Analisis ABC

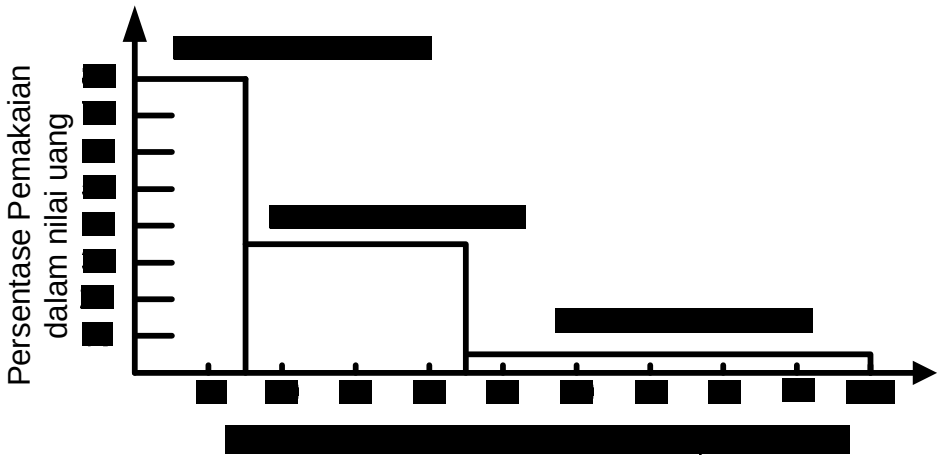
Perusahaan dapat menetapkan sistem untuk mengelola persediaan. Bagaimana mengelompokkan produk-produk persediaan(disebut analisis ABC) dan bagaimana mempertahankan keakuratan catatan persediaan yang ada.

Analisis ABC membagi persediaan ditangan kedalam tiga kelompok berdasarkan volume tahunan dalam jumlah uang. Analisis ABC merupakan persediaan dari prinsip Pareto. Prinsip Pareto menyatakan bahwa ada “beberapa yang penting dan banyak yang sepele”.Pemikiran yang mendasari ini adalah bagaiman memfokuskan sumber daya pada bagian persediaan penting yang sedikit itu dan bukan pada bagian persediaan yang banyak namun tidak penting.

Untuk menentukan nilai uang tahunan dari volume dalam analisis ABC, kita mengukur permintaan tahunan dari setiap butir persediaan dikalikan dengan biaya

perunit. Butir persediaan kelas A adalah persediaan-persediaan yang jumlah nilai uang pertahunnya tinggi. Butir-butir persediaan semacam ini mungkin mewakili sekitar lima belas persen dari butir-butir persediaan total, tetapi mewakili tujuh puluh persen sampai delapan puluh persen dari total biaya persediaan. Butir persediaan kelas B adalah butir-butir persediaan yang volume tahunannya dalam nilai uang sedang. Butir persediaan ini mungkin hanya mewakili tiga puluh persen dari keseluruhan persediaan dan lima belas sampai dua puluh lima persen dari nilainya. Butir persediaan yang volume tahunannya kecil, dinamakan kelas C, yang mewakili hanya lima persen dari keseluruhan volume tahunan tetapi sekitar lima puluh lima persen dari keseluruhan persediaan.

Secara grafik, persediaan analisis ABC di CV. Prima Dua Jaya akan terlihat sebagaimana ditunjukkan Gambar 2.1



Gambar 2.1 Penggambaran Grafik Dari Analisis ABC

Kriteria dalam nilai uang dapat menentukan klasifikasi butir persediaan. Misalnya, perubahan teknisyang diantisipasi, masalah-masalah pengiriman, masalah mutu, atau biaya per unit yang tinggi dapat membawa butir persediaan yang menaik ke dalam klasifikasi yang lebih tinggi. Keuntungan pembagian butir-butir persediaan ke dalam kelas-kelas memungkinkan ditetapkannya kebijakan dan pengendalian untuk setiap kelas yang ada.

Kebijakan yang dapat didasarkan pada analisis ABC mencakup hal-hal di bawah ini :

1. Perkembangan sumber daya pembelian yang dibayarkan kepada pemasok harus lebih tinggi untuk butir persediaan A dibandingkan butir persediaan C.
2. Butir persediaan A, berlainan dengan butir persediaan B dan C, harus dikendalikan secara lebih ketat, mungkin karena butir persediaan A ini ditempatkan di wilayah yang lebih tertutup dan mungkin karena keakuratan catatan persediaannya harus lebih sering diverifikasi.
3. Meramalkan butir persediaan A mungkin harus lebih berhati-hati dari pada meramalkan butir (kelas) persediaan yang lain.

Peramalan yang lebih baik, pengandaliaan fisik, keandalan pemasok, dan pengurangan besar stock pengaman dapat dihasilkan oleh analisis ABC, yang menyebabkan CV. Prima Dua Jaya menggunakan teknik ini.

2.2.1 Keakuratan Catatan Persediaan

Kebijakan persediaan yang baik tidak berarti manajemen tidak mengetahui persediaan apa yang ada di tangan. Keakuratan catatan mengenai persediaan ini penting dalam sistem produksi dan persediaan. Keakuratan ini memungkinkan perusahaan untuk tidak merasa yakin bahwa "beberapa dari seluruh produk" berada di persediaan dan memungkinkan perusahaan untuk tidak hanya memfokuskan pada butir-butir persediaan yang dibutuhkan. Bila perusahaan dapat secara akurat menentukan apa yang ada ditangan maka perusahaan itu dapat membuat keputusan yang tepat mengenai pemesanan, pejadwalan, dan pengangkutan.

2.2.2 Penghitungan siklus

Walaupun perusahaan mungkin telah melakukan berbagai usaha untuk mencatat persediaan secara akurat, catatan atau arsip ini harus divertifikasi melalui pemeriksaan/audit yang berkelanjutan. Audit semacam ini disebut penghitungan siklus (*cycle counting*). Dulu, banyak perusahaan mengambil persediaan fisik tahunan. Hal ini sering berarti penghentian fasilitas produksi dan menyuruh orang-orang yang tidak berpengalaman untuk menghitung komponen dan bahan baku. Arsip persediaan harus divertifikasi melalui perhitungan siklus, bukannya dengan cara diatas. Penghitungan siklus menggunakan klasifikasi persediaan yang dikembangkan lewat analisis ABC. Dengan prosedur penghitugan siklus, butir-butir persediaan dihitung, arsip divertifikasi, dan ketidak akuratan di dokumentasi secara berkala. Penyebab ketidak akuratan ini kemudian dilacak dan tindakan perbaikan yang tepat

kemudian diambil sesuai klasifikasi butir persediaannya. Butir persediaan A akan dihitung secara rutin, mungkin sekali sebulan, butir persediaan B kurang rutin mungkin sekali dalam empat bulan, butir persediaan C akan dihitung mungkin sekali dalam setahun.

2.3 Permintaan Independen

Model pengendalian persediaan mengasumsikan bahwa permintaan untuk suatu barang bersifat independen atau dependen terhadap permintaan barang lainya. Pada tugas akhir ini penulis hanya akan membahas dan menggunakan permintaan independen. Pengendalian persediaan pada CV. Prima Dua Jaya akan bahan mentahnya seperti besi dan plat bersifat independen, yang artinya bahan mentah ini secara tidak langsung tidak saling berkaitan, bahan mentah tersebut akan saling berhubungan apabila telah menjadi suatu produk.

Didalam permintaan independen terdapat beberapa biaya yang perlu diperhatikan, seperti biaya penyimpanan, pemesanan dan pemasangan. Biaya penyimpanan (*holding Cost*) adalah biaya-biaya yang berkaitan dengan penyimpanan atau "penahanan" (*carrying*) persediaan sepanjang waktu tertentu. Oleh karena itu, biaya penyimpanan juga mencakup biaya yang berkaitan dengan gudang, seperti biaya asuransi, penanganan bahan baku. Biaya pemesanan (*ordering cost*) mencakup biaya-biaya pasokan, pemrosesan pesanan, tenaga para pekerja, dan sebagainya. Pada saat produk pesanan dibuat, timbul pula biaya biaya pemesanan, tetapi biaya ini dikenal dengan nama biaya pemasangan. Biaya pemasangan adalah biaya-biaya untuk

mempersiapkan mesin atau proses untuk memproduksi pesanan. Manajer operasi dapat menurunkan biaya pesanan dengan mengurangi biaya pemasangan dan dengan menggunakan prosedur yang efisien semacam pembayaran dan pemesanan elektronik.

Pada perusahaan CV. Prima Dua Jaya, biaya pemasangan secara erat berhubungan dengan waktu pemasangan (*setup time*). Pemasangan biasanya menuntut adanya sejumlah kerja tertentu sebelum suatu operasi betul-betul dijalankan di pusat kerja. Kebanyakan persiapan yang diperlukan oleh pemasangan dapat dilakukan sebelum penghentian mesin atau proses yang ada. Waktu pemasangan dapat secara substansial dikurangi. Mesin dan proses yang biasanya memerlukan berjam-jam untuk dipasang, kini dapat dipasang dalam waktu kurang dari sepuluh menit apabila dikerja dengan imajinatif. Pengurangan waktu pemasangan merupakan cara yang sangat baik untuk mengurangi investasi persediaan dan memperbaiki produktivitas.

Pada bagian ini penulis akan memperkenalkan tiga model persediaan yang akan digunakan pada CV. Prima Dua Jaya yang mengedepankan dua pertanyaan penting, kapan pemesanan dilakukan dan berapa banyak yang akan dipesan.

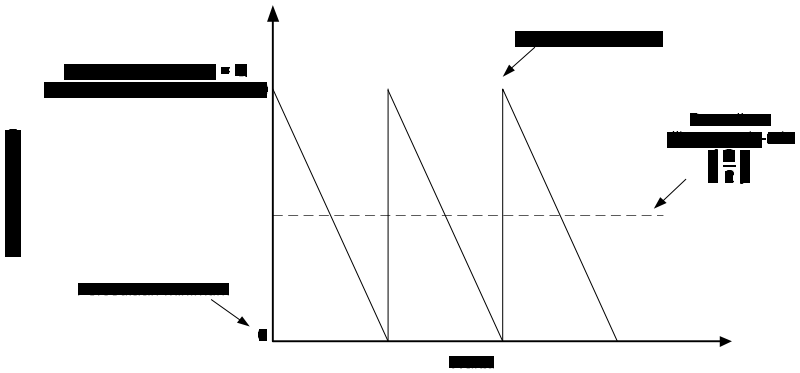
1. Model dasar *Economic order Quantity* (EOQ).
2. Model *Production Order Quantity*
3. Model *Quantity Discount*.

2.3.1 Model Dasar *Economic Order Quantity* (EOQ)

EOQ merupakan salah satu teknik pengendalian persediaan tertua dan paling terkenal. Teknik ini relatif mudah digunakan, tetapi didasarkan pada beberapa asumsi:

1. Tingkat permintaan diketahui dan bersifat konstan.
2. *Lead time*, yaitu waktu antara pemesanan dan penerimaan pesanan, diketahui, dan bersifat konstan.
3. Persediaan diterima dengan segera. Dengan kata lain, persediaan yang dipesan tiba dalam bentuk kumpulan produk, pada satu waktu.
4. Tidak mungkin diberikan diskon.
5. Biaya variabel yang muncul hanya biaya pemasangan atau pemesanan dan biaya penahanan atau penyimpanan persediaan sepanjang waktu.
6. keadaan kehabisan stok (kekurangan) dapat dihindari sama sekali bila pemesanan dilakukan pada waktu yang tepat.

Dengan asumsi-asumsi di atas, grafik penggunaan persediaan sepanjang waktu bentuknya seperti ikan hiu, seperti terlihat di gambar 2.2. pada gambar 2.2, Q mewakili jumlah yang dipesan. Bila jumlahnya 500 potong besi dan plat, keseluruhan jumlah bahan itu tiba pada satu waktu (pada saat pesanan diterima). Maka, tingkat persediaan meningkat dari nol ke lima ratus potong besi dan plat. Secara umum, tingkat persediaan meningkat dari nol ke Q unit pada saat pesanan tiba.



Gambar 2.2 Penggunaan Persediaan Sepanjang Waktu

Karena tingkat permintaan kostan sepanjang waktu, persediaan menurun dengan tingkat yang sama sepanjang waktu. (lihat garis miring digambar 2.2). ketika tingkat persediaan mencapai 0, pesanan baru dibuat dan diterima, dan tingkat persediaan meningkat lagi ke Q unit (diwakili oleh garis vertikal). Proses ini terus terjadi sepanjang waktu.

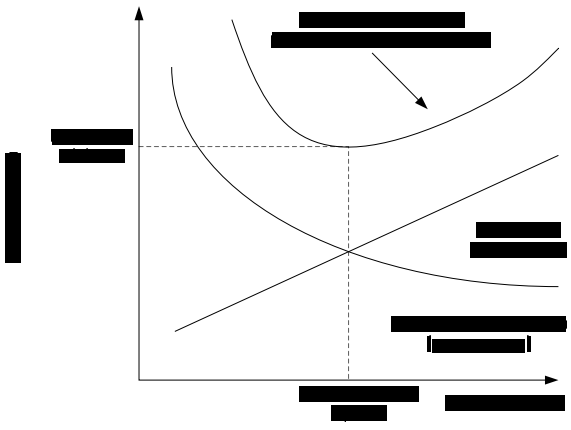
a. Minimasi Biaya

Tujuan dari kebanyakan model persediaan adalah untuk meminimasi biaya total (keseluruhan). Dengan asumsi-asumsi yang baru saja diberikan diatas, biaya yang signifikan adalah biaya pemasangan (pemesanan) dan biaya penahanan (penyimpanan). Biaya-biaya yang lainnya, seperti biaya persediaan itu sendiri, sifatnya konstan. Maka dengan meminimasi jumlah biaya pemasangan dan penahanan, kita juga meminimasi biaya total. Sebagai alat bantu visulisasi hal ini, pada Gambar 2.3 kami berikan grafik biaya total sebagai fungsi dari *Order Quantity* (jumlah yang dipesan) , Q. Ukuran pesanan optimalnya Q*, yang merupakan jumlah

pesanan yang meminimasi biaya total. Seiring dengan kenaikan jumlah yang dipesan, biaya penahan akan naik karena rata-rata persediaan yang dijaga lebih besar.

Perlu dicatat bahwa di gambar 2.3 jumlah pesanan optimalnya muncul di titik dimana kurva biaya pemesanan dan kurva penyimpanannya berpotongan. Hal ini bukannya tidak disengaja. Dengan model EOQ, jumlah pesanan optimal akan muncul di titik dimana biaya pemasangan totalnya sama dengan biaya penahanan total. Biaya total minimum muncul karena kemiringan biaya totalnya nol. Dengan menggunakan kenyataan ini, dikembangkanlah persamaan yang langsung mencari nilai Q*. Tahapan yang harus dilakukan adalah :

1. Mengembangkan persamaan untuk biaya pemasangan atau pemesanan.
2. Mengembangkan persamaan untuk biaya penahanan atau penyimpanan.
3. Menetapkan biaya pemasangan sama dengan biaya penahanan.
4. Menyelesaikan persamaan dengan hasil angka jumlah pesanan yang optimal.



Gambar 2.3 Biaya Total Sebagai Fungsi Jumlah Pesanan

Dengan menggunakan variabel-variabel dibawah ini, kita dapat menentukan biaya pemasangan dan penyimpanan, sehingga didapatkan nilai Q*:

Q = Jumlah barang setip pemesanan

Q^* = Jumlah optimal barang per pemesanan (EOQ)

D = Permintaan tahunan barang persediaan, dalam unit

S = Biaya pemasangan atau pemesanan untuk setiap pesanan

H = Biaya penahanan atau penyimpanan per unit per tahun

1. Biaya pemasangan tahunan = (jumlah pesanan yang dilakukan pertahun)(biaya pemasangan atau pemesanan setiap kali pesan).

$$= \left(\frac{\text{Permintaan tahunan}}{\text{jumlah barang setiap pemesanan}} \right) (\text{Biaya pemesanan atau pemasangan setiap pesanan})$$

$$= \left(\frac{D}{Q} \right) (S)$$

$$= \frac{D}{Q} S$$

2. Biaya penyimpanan tahunan = (tingkat persediaan rata-rata)(biaya penyimpanan perunit per tahun)

$$= \left(\frac{\text{Jumlah pesanan}}{2} \right) (\text{Biaya penyimpanan per unit per tahun})$$

$$= \left(\frac{Q}{2} \right) (H)$$

$$= \frac{Q}{2} H$$

3. Jumlah pesanan optimal ditemukan pada saat biaya pemasangan tahunan sama dengan biaya penyimpanan tahunan, yakni :

$$= \frac{D}{Q} S = \frac{Q}{2} H$$

4. Untuk mendapatkan nilai Q^*, lakukan perkalian silang dan pisahkan Q di sebelah kiri tanda sama dengan.

$$2DS = Q^2 H$$

$$Q^2 = \frac{2DS}{H}$$

$$Q^* = \sqrt{\frac{2DS}{H}} \qquad (2.1)$$

Setelah membuat persamaan untuk mencari EOQ, Q^*, dapat langsung memecahkan masalah persediaan.

Kita dapat pula menetapkan jumlah pemesanan yang ingin dibuat sepanjang tahun yang bersangkutan (N) dan waktu yang diinginkan antar-pemesanan(T), sebagai berikut:

$$\text{Jumlah pemesanan yang diinginkan} = N = \frac{\text{Permintaan}}{\text{Jumlah unit yang dipesan}} = \frac{N}{Q^*} \qquad (2.2)$$

$$\text{Jumlah waktu antar-pemesanan yang diinginkan} = \frac{\text{Jumlah hari kerja per hari}}{N} \qquad (2.3)$$

Sebagaimana telah dibahas di awal bagian ini, biaya persediaan tahunan merupakan penjumlahan biaya pemesanan dan biaya penyimpanan :

Biaya tahunan total = biaya pemesanan + biaya penyimpanan (2.4)

Dengan konteks variabel-variabel yang ada di model EOQ, kita dapat menuangkan biaya total sebagai :

$$TC = \frac{D}{Q}S + \frac{Q}{2}H \qquad (2.5)$$

b. Titik Pemesanan Ulang (*Reorder Point*)

Setelah menentukan berapa yang akan dipesan, kita akan melihat pada pertanyaan persediaan yang kedua, kapan pesanan akan dilakukan. Model persediaan sederhana mengasumsikan bahwa penerimaan suatu pesanan bersifat seketika. Dengan kata lain, model-model persediaan mengsumsikan bahwa perusahaan akan menunggu sampai tingkat persediaanya mencapai nol sebelum perusahaan memesan lagi, pada CV. Prima dua Jaya hal ini sering terjadi. Akan tetapi, waktu antara dilakukannya pemesanan, disebut *Lead Time* atau waktu pengiriman, bisa cepat beberapa jam atau lambat beberapa bulan. Maka keputusan kapan akan memesan biasanya diungkapkan dalam konteks titik pemesanan ulang, tingkat persediaan dimana harus dilakukan pemesanan, yang dapat dilihat pada gambar 2.4.

Titik pemesanan ulang (*reorder point*) dicari dengan cara :

ROP = (permintaan perhari)(*Lead Time* untuk pemesanan baru dalam hari)

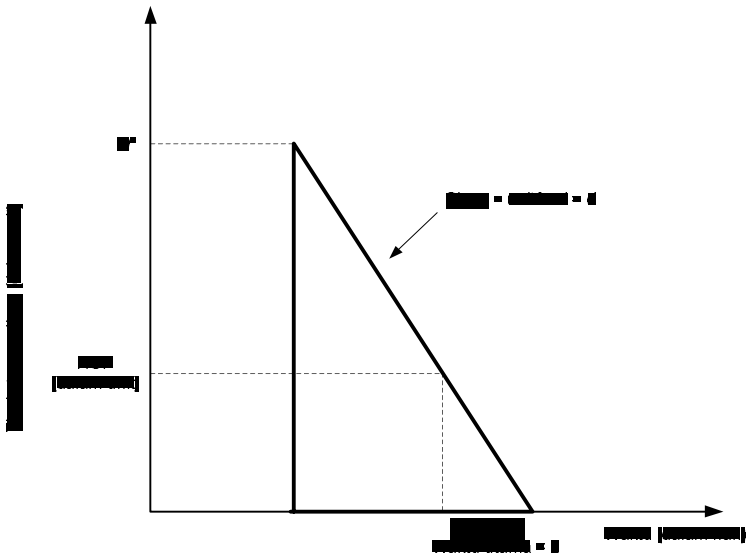
= d x L

Persamaan diatas mengasumsikan apabila di CV. Prima Dua Jaya permintaanya sama dan bersifat konstan dikarenakan sering terjadi hal demikian. Bila

tidak demikian halnya, harus ditambah stok tambahan, sering kali disebut stok pengamanan (*safety stock*).

Permintaan per hari, *d*, dicari dengan membagi permintaan tahunan, *D*, dengan jumlah hari kerja pertahun :

$$d = \frac{D}{\text{Jumlah hari kerja per tahun}}$$

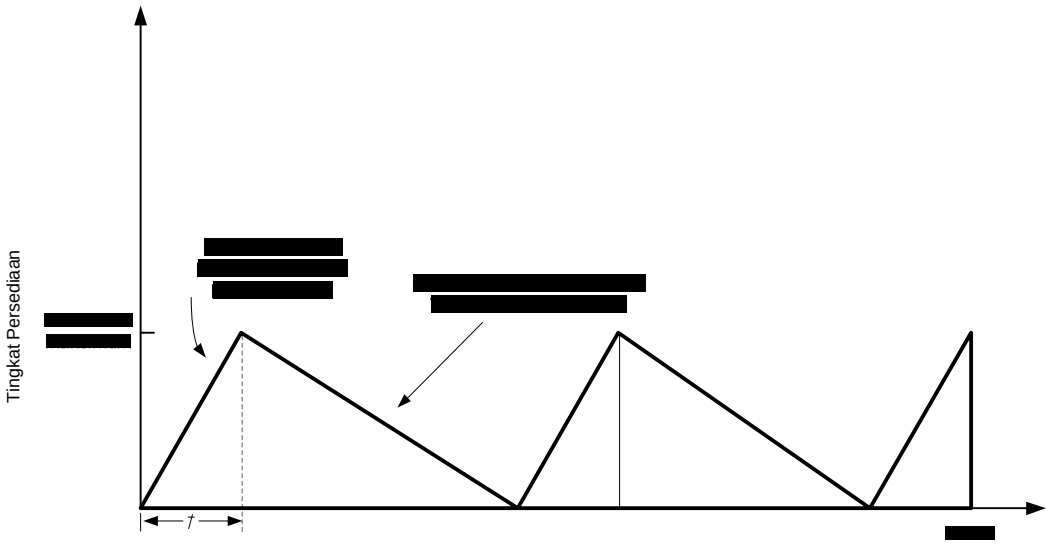


Gambar 2.4 Kurva Titik Pemesanan Ulang

2.3.2 Model *Production Order Quantity*

Pada model persediaan sebelumnya, keseluruhan pemesanan persediaan CV. Prima Dua Jaya diterima pada satu waktu. Meskipun demikian, ada saat-saat tertentu dimana CV. Prima Dua Jaya menerima persediaannya sepanjang suatu periode. Keadaan seperti ini mengharuskan pemakaian model yang berbeda, yaitu model yang

tidak memerlukan asumsi penerimaan pesanan sekita. Model ini dapat diterapkan ketika persediaan secara terus-menerus mengalir atau terbentuk sepanjang suatu periode waktu setelah dilakukan pemesanan atau ketika produk diproduksi dan dijual pada saat yang bersamaan. Dengan keadaan demikian, kita dapat memasukan catatan tingkat produksi atau arus persediaan setiap harinya, dan tingkat permintaan setiap harinya. Gambar 2.5 menunjukkan tingkat persediaan sebagai fungsi waktu.



Gambar 2.5 Perubahan Tingkat Persediaan Sepanjang Periode Waktu Tertentu.

Karena model ini cocok untuk lingkungan produksi, model ini lebih dikenal dengan sebutan model *production order quantity* atau model jumlah produksi. Model ini berguna ketika persediaan di CV. Prima Dua Jaya secara terus-menerus terbentuk sepanjang waktu dan asumsi EOQ tradisionalnya valid. Model ini dibuat dengan menetapkan biaya pemesanan atau pemasangan sama dengan biaya penyimpanan atau penahanan, sehingga didapatkan Q^*. Dengan menggunakan simbol-simbol berikut,

kita dapat menentukan persamaan untuk biaya penyimpanan persediaan tahunan untuk model pengoperasian produksi ini:

Q = Jumlah unit pmesanan

H = Biaya penyimpanan per unit per tahun

p = Tingkat produksi tahunan

d = Tingkat permintaan harian atau tingkat penggunaan

t = Lama jalannya produksi, dalam satuan hari

1. $\left(\begin{matrix}\text{Biaya penyimpanan}\\ \text{persediaan tahunan}\end{matrix}\right) = (\text{Tingkat persediaan tahunan}) \text{ x } \left(\begin{matrix}\text{Biaya penyimpanan}\\ \text{per unit per tahun}\end{matrix}\right)$

2. $\left(\begin{matrix}\text{Tingkat persediaan}\\ \text{rata - rata}\end{matrix}\right) = \left(\dfrac{\text{Tingkat persediaan maksimum}}{2}\right)$

3. $\left(\begin{matrix}\text{Tingkat persediaan}\\ \text{maksimum}\end{matrix}\right) = \left(\begin{matrix}\text{Total yang diproduksi selama}\\ \text{berjalannya operasi}\end{matrix}\right) - \left(\begin{matrix}\text{Total yang terpakai selama}\\ \text{berjalannya operasi}\end{matrix}\right)$

$= pt - dt$

Tetapi Q = total yang diproduksi = pt, dan oleh karena itu, $t = Q/p$. Dengan demikian,

$$\text{Tingkat persediaan maksimum} = p\left(\frac{Q}{p}\right) - d\left(\frac{Q}{p}\right)$$

$$= Q - \frac{d}{p}Q$$

4. Biaya penyimpanan persediaan tahunan =

$$\frac{\text{Tingkat persediaan maksimum}}{2} \text{ x } H = \frac{Q}{2}\left[1 - \left(\frac{d}{p}\right)\right]H$$

Dengan menggunakan persamaan untuk biaya diatas dan persamaan untuk biaya pemesanan yang dikembangkan pada model EOQ dasar kita akan menghitung jumlah optimal per pemesanan dengan cara menyamakan biaya pemesanan dan biaya penyimpanan :

$$\text{Biaya pemesanan} = (D/Q)S$$

$$\text{Biaya penyimpanan} = \frac{1}{2}HQ[1-(d/p)]$$

Tetapkan biaya pemesanan sama dengan penyimpanan untuk mendapatkan Q*p.

$$\frac{D}{Q}S = \frac{1}{2}HQ[1-(d/p)]$$

$$Q^2 = \frac{2DS}{H[1-(d/p)]}$$

$$Q^* = \sqrt{\frac{2DS}{H[1-(d/p)]}} \qquad (2.7)$$

Persamaan diatas, *Q*p*, dapat kita gunakan untuk menghitung jumlah pesanan atau produksi optimal bila persediaannya selagi dikonsumsi juga diproduksi.

Kita juga dapat menghitung *Q*p* apabila tersedia data tahunan. Bila yang digunakan adalah data tahunan, kita dapat menghitung *Q*p* dengan cara :

$$Q^*\,p = \sqrt{\frac{2DS}{H[1-(D/P)]}} \qquad (2.8)$$

di mana

D = tingkat permintaan tahunan

P = tingkat produksi tahunan

2.3.3 Model *Quantity Discount*

Untuk meningkatkan penjualan, CV. Prima Dua Jaya menawarkan potongan harga untuk para pelanggan mereka. *Quantity discount* ini secara sederhana merupakan pengurangan harga (*P*) untuk barang yang dibeli dengan jumlah yang lebih besar. Tidak terlalu bila daftar potongan harga dengan berbagai potongan untuk pesanan-pesanan dalam jumlah besar. Perusahaan harus memutuskan kapan dan berapa banyak pemesanan akan dilakukan. Akan tetapi, dengan potongan harga, bagaimana perusahaan tersebut membuat keputusan ini?

Bagi model persediaan lain yang sejauh ini telah dibahas, tujuan keseluruhannya adalah minimasi biaya total. Karena biaya per unit untuk pemesanan yang semakin banyak akan semakin rendah yangdisebabkan diskon yang semakin tinggi. Perusahaan mungkin akan tergoda memesan dengan jumlah yang besar agar bisa memanfaatkan biaya produksi yang lebih rendah. Memesan jumlah besar mungkin tidak akan meminimasi biaya persediaan total. Dengan meningkatnya potongan harga, biaya produksi menurun, tetapi biaya penyimpanan meningkat karena pesanannya besar. Maka, *trade-off* pada saat perusahaan mempertimbangkan potongan harga terletak antara biaya produk yang berkurang dan biaya penyimpanan yang bertambah. Bila kita memasukkan biaya produk, persamaan utuk biaya persediaan totalnya menjadi:

Biaya total = Biaya pemesanan + Biaya penyimpanan + Biaya produk

atau

$$Tc = \frac{D}{Q}S + \frac{QH}{2} + PD \tag{2.9}$$

Di mana,

Q = Jumlah unit yang dipesan

D = Permintaan tahunan dalam satuan unit

S = Biaya pemesanan per pemesanan

P = Harga per unit

H = Biaya penyimpanan per unit per tahun

Kini perusahaan harus menentukan jumlah yang akan meminimasi biaya persediaan tahunan total. Karena ada beberapa potongan harga, proses ini meliputi empat tahap:

Tahap 1. Untuk setiap potongan harga, hitunglah nilai Q*, dengan menggunakan persamaan dibawah ini:

$$Q^* = \sqrt{\frac{2DS}{IP}} \tag{2.10}$$

Harus dicatat bahwa biaya penyimpanan adalah *IP*, dan bukan *H*. Karena harga produk merupakan faktor dalam biaya penyimpanan tahunan, perusahaan tidak dapat mengasumsikan bahwa biaya penyimpanan bersifat konstan ketika harga per unit berubah untuk setiap potongan harga yang diberikan. Maka, biasanya biaya penyimpanan (*I*) diungkapkan sebagai persentase dari harga per unit per tahun yang konstan, *H*.

Tahap 2. Untuk setiap tingkat potongan harga, bila jumlah pesanannya terlalu rendah untuk mendapatkan potongan harga, sesuaikan jumlah pesanan keatas ke jumlah terendah yang memungkinkan diperolehnya potongan harga dalam daftar potongan harga. Alasan dilakukannya tahap kedua mungkin tidak terlalu jelas. Bila jumlah pesanannya berada di bawah kisaran jumlah yang akan memungkinkan perolehan potongan harga, jumlah di antara kisaran ini mungkin tetap akan menghasilkan biaya total terendah.

Tahap 3. Dengan menggunakan persamaan biaya total di atas, hitung biaya total untuk setiap Q^* yang ditetapkan pada tahap 1 dan 2. Bila perusahaan harus menyesuaikan Q^* keatas karena Q^* tadinya berada di bawah kisaran jumlah, pastikan bahwa kita menggunakan nilai Q^* yang telah disesuaikan.

Tahap 4. Pilih Q^* yang biaya totalnya paling rendah seperti telah dihitung ditahap 3. Q^* dengan biaya total terendah ini merupakan jumlah pesanan yang akan meminimkan biaya persediaan total.

2.4 Pengenalan Pemrograman Borland Delphi 7.0

Borland Delphi atau yang biasa disebut Delphi saja, merupakan sarana pemrograman aplikasi visual. Bahasa pemrograman yang dipakai adalah bahasa pemrograman Pascal, atau yang kemudian juga disebut bahasa pemrograman Delphi. Delphi merupakan generasi penerus dari Turbo Pascal. Turbo Pascal yang

diluncurkan pada tahun 1983 dirancang untuk dijalankan pada Sistem Operasi DOS (yang merupakan Sistem Operasi yang paling banyak digunakan pada saat itu). Sedangkan Delphi yang diluncurkan pertama kali pada tahun 1995 dirancang untuk beroperasi dibawah Sistem Operasi Windows.

Kebutuhan akan adanya program aplikasi yang bekerja dibawah Sistem Operasi Windows serta memiliki antarmuka visual yang *user friendly* telah memancing minat banyak orang menggunakan bahasa pemrograman yang mampu menyediakan antarmuka grafis.

Borland Delphi merupakan salah satu bahasa pemrograman yang semenjak diluncurkan pertama kali, langsung dilirik dan diminati oleh para *programmer* komputer. Hal ini disebabkan karena Delphi menyediakan fasilitas untuk pembuatan aplikasi dengan antarmuka visual secara mudah dan dapat memberikan hasil yang memuaskan.

Delphi merupakan sarana yang tangguh untuk pembuatan aplikasi, mulai dari sarana untuk pembuatan *form*, *menu*, *toolbar* hingga kemampuan untuk menangani pengelolaan basis data yang besar. Kelebihan-kelebihan yang dimiliki Delphi antara lain karena pada Delphi, *form* dan komponen-komponennya dapat dipakai ulang dan dikembangkan, tersedia *template* aplikasi dan *template form*, memiliki lingkungan pengembangan visual yang dapat diatur sesuai dengan kebutuhan.menghasilkan *file* terkompilasi yang berjalan lebih cepat serta kemampuan mengakses data dari bermacam-macam format.

Dalam membuat aplikasi menggunakan Delphi, Anda akan sering bertemu dengan istilah-istilah yang biasa dipakai dalam lingkungan pemrograman visual, yaitu *object*, *property* dan *event*. Delphi menerapkan konsep aplikasi yang digerakkan oleh *event* (*event driven*). Pemrograman model memang telah ada sebelum *Graphical User Interface* (GUI) dibuat dan diimplementasikan dalam beberapa cara. Tetapi teknik menjadi lebih berkembang dan banyak disukai para *programmer* dan pemakai aplikasi semenjak diperkenalkannya *Mouse*.

Pemrograman *event-driven* mencoba melangkapi seperti yang telah disebutkan diatas. Delphi merupakan bahasa objek Pascal di dalam lingkungan pemrograman visual. Kombinasi ini menghasilkan sebuah lingkungan pengembangan aplikasi yang berorientasi objek (*Object Oriented Programming*). Dengan konsep ini, maka pembuatan aplikasi dengan menggunakan Delphi dapat dilakukan dengan cepat dan menghasilkan aplikasi yang tangguh.

2.5 Rekayasa Perangkat Lunak

Perangkat lunak merupakan suatu program yang berisi sejumlah intruksi yang dibuat untuk melakukan sejumlah proses dengan tujuan sebagai alat pengendalian kegunaan sistem atau untuk menyelesaikan kesulitan suatu aplikasi tertentu.

Dalam membangun suatu perangkat lunak selalu timbul berbagai macam kesulitan. Kesulitan yang terdapat dalam pembangunan perangkat lunak dapat diatasi dengan cara mengenalipermasalahan yang ada. Tapi solusi ini harus dapat memberikan bantuan yang berguna bagi pembangunan perangkat lunak dan

meningkatkan kualitas perangkat lunak dan akhirnya penerapan perangkat lunak pada perangkat keras.

2.5.1 Pengertian Rekayasa Perangkat Lunak

Untuk mengatasi kesuliatan tersebut, disusunlah suatu cara atau metodologi yang terangkum dalam rekayasa perangkat lunak, yang secara umum adalah disiplin ilmu untuk membangun perangkat lunak yang mengunakan metode-metode untuk seluruh fase-fase pembangunan perangkat lunak. Penentuan alat bantu yang baik untuk mengoptimalisasikan metode-metode ini, pembuatan model perangkat lunak yang sangat berpengaruh untuk penerapan perangkat lunak, serta teknik-teknik yang baik untuk menjamin kualitas perangkat lunak.

Rakayasa perangkat lunak didefinisikan berbeda-beda oleh setip orang tetapi pada intinya bahwa rekayasa perangkat lunak dengan menggunakan prinsip-prinsip rekayasa yang ditunjukan oleh sekelompok orang untuk menghasilkan perangkat lunak yang baik.

Untuk menghasilkan rakayasa perangkat lunak yang baik adalah :

a. Perangkat yang dibuat harus mempunyai sifat yang mudah untuk diperbaiki dan di kembangkan, sehingga dapat terus disesuaikan dengan perkebangan jaman, baik dari segi teknologi maupun manajemen.
b. Perangkat lunak harus dapat diandalkan, sehingga kemampuan dari perangkat lunak yang diharapkan bekerja tanpa kesalahan.

c. Perangkat lunak harus bersifat efisien, yaitu tingkat kemampuan perangkat lunak dalam mendayagunakan sumber untuk menjalankan fungsi yang didefinisikan oleh pemakai.

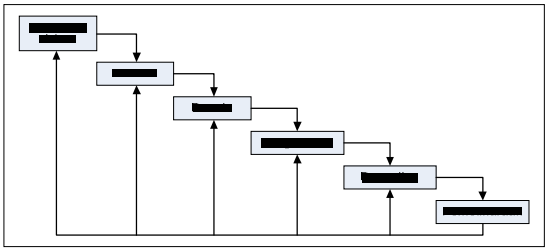
d. Perangkat lunak harus memberikan antarmuka bagi pemakai dengan tepat.

2.5.2 Pendekatan Pembangunan Perangkat Lunak

Beberapa metodologi pembangunan perangkat lunak dikembangkan orang, masing-masing metodologi memiliki kelebihan-kelebihan dan kekurangan-kekurangan tersendiri. Walaupun demikian setiap pendekatan yang ada bertujuan agar dapat menghasilkan suatu perangkat lunak yang baik dan sesuai dengan kebutuhan pengguna.

a. *Water fall* Model

Pada tugas akhir ini digunakan metoda *Water fall model* bisa juga disebut model klasik. Dengan pendekatan sekuensial untuk pengembangan perangkat lunak dimulai pada tingkat sistem dan berkembang sampai dengan pengetesan dan perbaikan kesalahan. Berikut ini adalah gambar siklus hidup pembangunan perangkat lunak dengan pendekatan *water fall model*.



Gambar 2.6 *Water fall model*

Berikut ini tahapan-tahapan pendekatan "*water fall model*" beserta pengertiannya :

1. Rekayasa sistem

Karena perangkat lunak selalu merupakan dari suatu sistem yang besar, maka harus diawali dengan menentukan kebutuhan-kebutuhan bagi seluruh elemen-elemen sistem, dan mengalokasikan beberapa subset dari kebutuhan-kebutuhan tersebut bagi perangkat lunak. Gambaran sistem merupakan hal yang paling penting pada saat perangkat lunak harus berinteraksi dengan elemen-elemen sistem lain seperti perangkat keras, manusia dan basis data. "*System engineering*" mencakup kumpulan kebutuhan-kebutuhan pada setiap tingkat sistem dengan jumlah yang kecil pada tingkat teratas perancangan dan analisis.

2. Analisis

Merupakan proses pengumpulan kebutuhan perangkat lunak. Untuk memahami program yang akan dibangun, analisis harus memahami wawasan informasi bagi perangkat lunak dan juga fungsi yang dibutuhkan performansi dan antarmuka.

Beberapa target lain yang ingin dicapai antara lain :

1. Mengetahui keinginan *user*.
2. Efisien biaya dan sumber daya pembangunan perangkat lunak.
3. Efisiensi waktu pembangunan perangkat lunak.

3. Desain

Perancangan perangkat lunak sebenarnya merupakan kumpulan proses yang difokuskan pada empat atribut yang berbeda pada program, yaitu struktur data, arsitektur perangkat lunak, rincian prosedur, dan karakteristik antarmuka. Proses perancangan, mentranslasikan kebutuhan-kebutuhan kedalam sebuah representasi perangkat lunak yang dapat dinilai kualitasnya sebelum pegkodean dimulai.

Beberapa target dari tahap perancangan adalah:

a. Spesifikasi perangkat lunak harus dapat dapat dimengerti oleh pembangun perangkat lunak.
b. Rancangan modul-modul memungkinkan untuk digunakan kembali pada aplikasi lain baik secara parsial maupun secara total.
c. Perangkat lunak yang dirancang harus dapat ditranfer dari suatu lingkungan sistem kelingkungan sistem lain.
d. Pembentukan dokumentasi perancangan perangkat lunak.

4. Pengkodean

Pada tahap ini, perancangan harus ditranslasikan kedalam bentuk yang dapat dibaca oleh bahasa mesin. Apabila perancangan dilakukan dengan rinci, pengkodean dapat dikerjakan secara mekanis. Yang menjadi target dari tahap pengkodean adalah menuliskan program secara rinci pada setiap modul.

5. Pengujian

Setelah program dohasilkan, selanjutnya dilakuka test program. Proses tes terfokus pada "*logical internal*" dari *software* yang memastikan bahwa semua pernyataan-pernnyataan telah dites, dan pada "*functional external*" yaitu tes dilakukan untuk menentukan kesalahan-kesalahan dan memastikan bahwa masukan dapat disesuaikan dengan kebutuhan.

Yang menjadi target pada tahap pengujian adalah:

1. Mendeteksi setiap kesalahanyang terjadi.
2. Memperbaiki setiap kesalahan yang terjadi.
3. Mencegah kesalahan baru selama perangkat lunak dimodifikasi.
4. Perangkat lunak yang telah sesuai diserahkan pada pengguna tepat pada waktu yang ditentukan.
5. Perangkat lunak yang dibangun harus efisien.
6. Perangkat lunak mampu beradaptasi dengan lingkungan.
7. Perangkat lunak dapat dipercaya/andal.

6. Pemeliharaan

Perangkat lunak secara tidak terasa menjalani perubahan setelah diserahkan kepada pengguna. Perubahan akan terjadi karena kesalahan-kesalahan yang ditemukan, karena perangkat lunak harus beradaptasi untuk menyesuaikan perubahan-[erubahan dalam lingkungan luarnya atau karena pengguna membutuhkan fungsi-fingsi atau peningkatan performansi. Misalnya kebutuhan

perubahan karena sistem operasi baru atau perangkat keras baru. Untuk memperbaiki perangkat lunak, maka dipergunakan kembali setiap langkah pada daur kehidupan terdahulu kepada program yang telah ada dari pada program yang baru.

b. *Spiral Model*

Model Spiral (*Spiral Model*) pada awalnya diusulkan oleh Boehm, adalah model prosesperagkat lunak yang evolusioner yang merangkai sifat iteratif dari prototipe dengan cara kontrol dan aspek sistematis dari model sekuensiallinier. Model ini berpotensi untuk pengembangan versi pertambahan perangkat lunak secara cepat. Di dalam proses spiral, perangkat lunak dikembangkan didalam sesuatu deretan pertambahan. Selama awal iterasi, rillis inkremental bisa merupakan sebuah model atau prototipe kertas. Selama iterasi berikutnya, sedikit demi sedikit dihasilkan versi sistem rekayasa yang lebi lengkap. Model spiral dibagi menjadi sejumlah aktifitas kerangka kerja, disebut juga wilayah tugas, diantara tiga sampai enam wilayah gambar, dibawah ini menggambarkan model spiral yang berisi enam wilayah tugas.

1. **Komunikasi Pelanggan (*Customer Communication*)**

   Tugas-tugas yang dibutuhkan untuk membangun komunikasi yang efektif diantara pengembang dan pelanggan.

2. **Perancangan (*Planning*)**

   Tugas-tugas yang dibutuhkan untuk mendefinisikan sumber daya, ketepatan waktu, dan proyek informasi lain yang berhubungan.

3. **Analisis Resiko (*Risk Analisys*)**

   Tugas-tugas yang dibutuhkan untuk menaksir resiko-resiko baik manejemen maupun teknis.

4. **Perekayasaan (*Engineering*)**

   Tugas-tugas yang dibutuhkan untuk membangun suatu atau lebih representasi dari aplikasi tesebut.

5. **Kontruksi dan peluncuran (*Contruction and Release*)**

   Tugas-tugas yang diutuhkan untuk mengkontruksi, menguji, memasang (*instal*) dan memberikan pelayanan kepada pemakai (contohnya pelatihan dan dokumentasi)

6. **Evaluasi pelanggan (*Customer evalution*)**

   Tugas-tugas yang dibutuhkan untuk memperoleh umpan balik dari pelanggan dengan didasarkan pada evaluasi representasi perangkat lunak, yang dibuat selama masa pemesanan.

   Model spiral menjadi sebuah pendekatan yang realistis bagi perkembangan sistem dan perangkat lunak berskala besar. Karena perangkat lunak terus bekerja selama proses bergerak, pengembang dan pemakai memahami dan bereaksi lebih baik terhadap resiko dari setiap tingkat evolusi. Model spiral menggunakan prototipe sebagai mekanisme penguranan resiko. Tetapi yang penting lagi, model spiral memungkinkan pengembang menggunakan pendekatan prototipe pada setiap keadaan didalam evolusi produk. Model spiral menjaga pendekatan langkah demi langkah secara sistematis seperti

yang diusulkan oleh siklus kehidupan klasik, tetapi memasukannya kedalam kerangka kerja iteratif yang secara realistis merefleksikan dunianyata. Model spiral membutuhkan pertimbangan langsung terhadap resiko teknis pada semua keaadaan proyek, yang jika dipakai secara benar akan mengurangi resiko sebelum menjadi sangat diplomatik. Tetapi seperti paradigma yang lain, model spiral bukanlah sebuah obat mujarab. Mungkin akan sangat sulit untuk meyakinkan konsumen (khususnya dalam situasi kontrak) bahwa pendekatan evolusioner bisa dikontrol. Model spiral memerlukan keahlian penaksiran resiko yang masuk akal, dan sangat bertumpu pada keahlian ini untuk mencapai keberhasilan. Jika sebuah resiko utama tidak ditemukan dan diatur, pasti akan terjadi masalah. Pada kenyataannya model spiral itu sendiri masih baru dan dipergunakan secara luas seperti paradigma sekuensial linier dan prototipe. Sebuah model akan membutuhkan waktu bertahun-tahun sampai kehandalan sebuah paradigma baru bisa dipertimbangkan untuk digunakan dengan pasti.

2.5.3 Elemen Model Analisis

Model analisis harus dapat mencapai tiga sasaran utama : (1) untuk menggambarkan apa yang dibutuhkan oleh pelanggan, (2) untuk membangun dasar bagi pembuatan desain perangkat lunak dan (3) untuk membatasi serangkaian persyaratan yang dapat divalidasi begitu perangkat lunak dibangun. Untuk mencapai

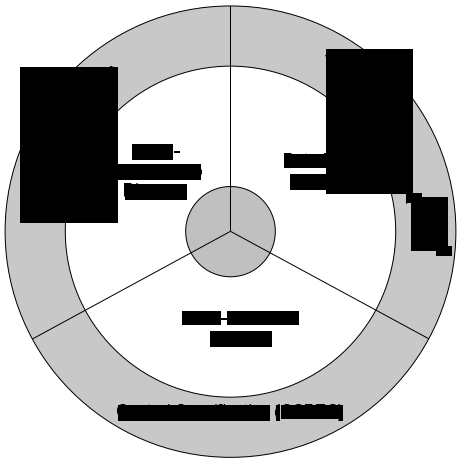
sasaran tersebut, model analisis yang ditarik selama analisis terstruktur berlangsung ditunjukkan pada gambar 2.7.

Pada inti model ada kamus data (*data dictionary*) – penyimpanan yang berisi deskripsi dari semua objek data yang dikomsumsi atau diproduksi oleh perangkat lunak. Dimana ada tiga diagram yang mengelilingi inti. *Entity-relationship diagram (ERD)* menggambarkan hubungan antara objek data. ERD adalah notasi yang digunakan untuk melakukan aktivitas pemodelan data. Atribut dari masing-masing objek data yang ditulis pada ERD dapat digambarkan dengan menggunakan dekskripsi objek data.

Data flow diagram (*DFD*) melayani dua tujuan : (1) untuk memberikan indikasi mengenai bagaimana data ditranformasi pada saat data bergerak melalui sistem dan (2) untuk menggambarkan fungsi-fungsi (dan sub-fungsi) yang mentrasformasi aliran data. DFD memberikan informasi tambahan yang digunakan selama analisis domain informasi dan berfungsi sebagai dasar bagi pemodelan fungsi. Deskripsi setiap fungsi yang disajikan pada DFD diisikan dalam sebuah spesifikasi proses (PSPEC).

State-tansition diagram (STD) menunjukkan bagaimana sistem bertingkah laku sebagai akibat dari kejadian eksternal. Untuk melakukannya, STD menunjukkan berbagai model tingkah laku (disebut *state*) sistem dan cara di mana transisi dibuat dari *state* satu ke *state* lainnya. STD berfungsi sebagai dasar bagi pemodelan tingkah laku. Informasi tambahan mengenai aspek kontrol dari perangkat lunak diisikan dalam spesifikasi kontrol atau *control specification (CSPEC)*.

Model analisis meliputi masing-masing diagram, spesifikasi, dan deskripsi, serta kamus yang ditulis pada Gambar 2.7. Pada penulisan tugas akhir ini, penulis menggunakan *data flow diagram* dalam pembuatan perangkat lunak persediaan.



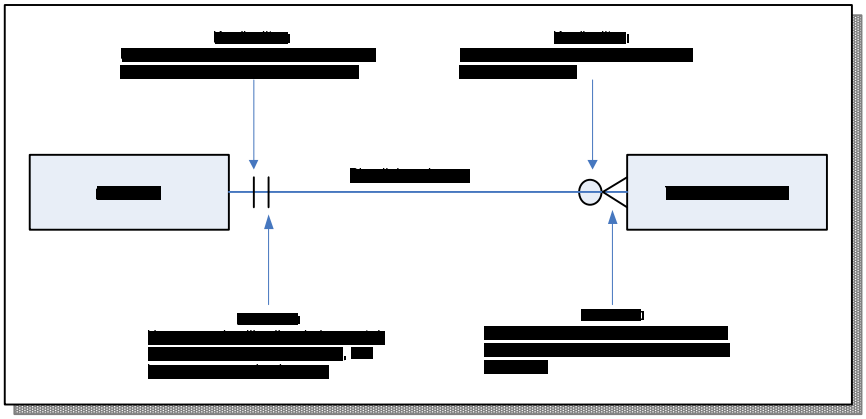
Gambar 2.7 Struktur Model Analisis

a. Objek Data

Objek data adalah representasi dari hampir semua informasi gabungan yang harus dipahami oleh perangkat lunak. Dengan informasi gabungan yang harus dipahami oleh perangkat lunak. Dengan informasi gabungan yang berarti memiliki sejumlah sifat atau atribut yang berbeda. Dengan demikian, “luas” (sebuah harga tunggal) tidak akan menjadi objek data yang valid, tetapi dimensi (gabungan antara ketinggian, luas dan kedalaman) dapat ditentukan sebuah objek.Objek data dapat berupa entitas - entitas eksternal (misalnya semua yang menghasilkan atau mengkomsumsi informasi).

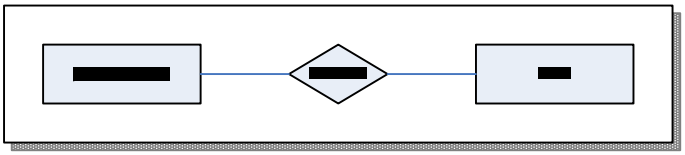
b. ***Entity Relationship Diagram***

ERD pada awalnya diusulkan oleh Peter Chen untuk desain sistem basis data relasional dan telah dikembangkan oleh yang lainnya. Serangkaian komponen utama diidentifikasikan untuk ERD: objek data, atribut, hubungan, dan berbagai tipe indikator. Tujuan utama dari ERD adalah untuk mewakili objek data dan hubungan mereka.



Gambar 2.8 Kardinalitas dan Modalitas

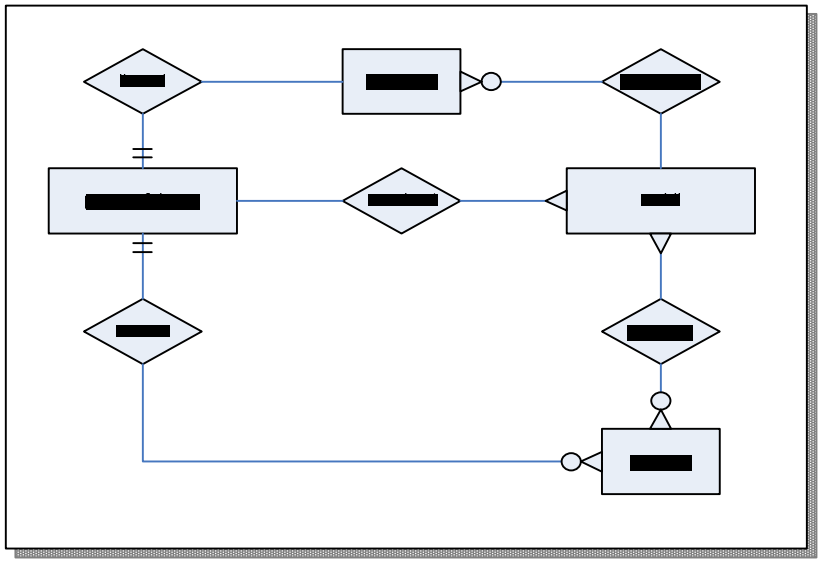
Objek data diwakili oleh persegi panjang yang diberi label. Hubungan ditunjukan dengan garis yang diberi labelyang menghubungkan objek. Dalam beberapa variasi ERD, garis yang menghubungkan berisi sebuah permata yang di beri label dengan hubungan tersebut. Sambungan antara data dan objek data hubungan dibangun dengan berbagai simbol khusus yang menunjukan kardinalitas dan modalitas.



Gambar 2.9 ERD Sederhana

Hubungan antara mobil dan pabrik akan direpresentasikan seperti diperlihatkan didalam gambar 2.9. Satu pabrik membangun satu atau banyak mobil. Diberikan konteks yang diimplikasikan oleh ERD, spesifikasi dari objek data mobil akan menjadi sangat berbeda dengan spesifikasi sebelumnya. Dengan mengamati simbol pada akhir garis sambungan antara objek dapat dilihat bahwa modalitas dari kedua peristiwa merupakan keharusan (garis vertikal).

Dengan memperluas model, kita mewakili ERD yang sangat disederhanakan dari elemen distribusi bisnis automobil. Objek data yang baru diperkenankan. Sebagai tambahan, hubungan baru transportasi, kontrak, ijin, dan stock meliputi bagaimana objek data diperlihatkan di dalam gambar berhubungan satu dengan yang lainnya. Tabel untuk masing-masing objek data yang diisikan di dalam ERD harus dikembangkan sesuai dengan aturan yang diperkenalkan.



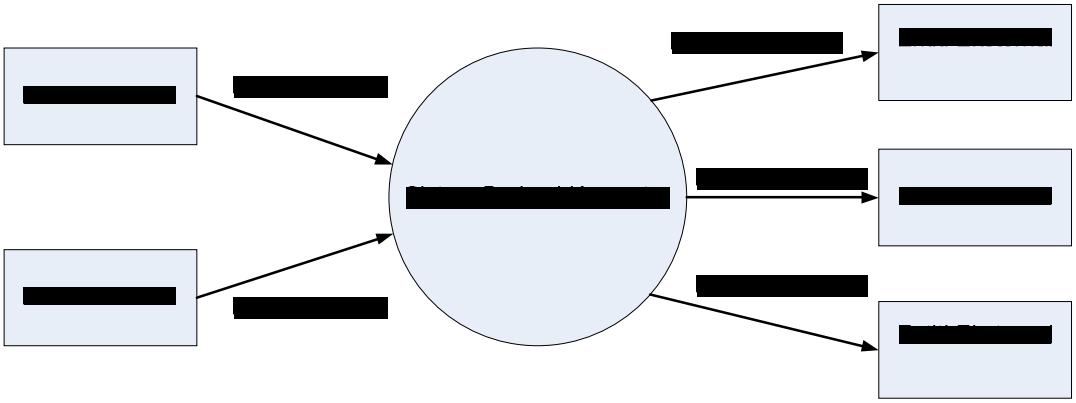
Gambar 2.10 ERD Diperluas

Sebagai tambahan untuk notasi ERD dasar. Analisis dapat direpresentasikan hirarki tipe objek data. Dalam banyak contoh, sebuah objek data secara aktual merepresentasikan sebuah kelas atau kategori informasi. Misalnya, objek data mobil dapat dikategorikan sebagai domestik, Eropa, atau Asia.

c. Diagram Aliran Data (DFD)

Pada saat informasi megalir melalui perangkat lunak, informasi dimodifikasi oleh suatu deretan transformasi. Diagram aliran data atau *data flow diagram* (DFD) adalah sebuah teknis grafis yang menggambarkan aliran informasi dan tranformasi yang diaplikasi pada saat data bergerak dari *input* menjadi *output*. Bentuk dasar dari suatu diagram aliran data diilustrasikan di dalam Gambar 2.8. DFD juga dikenali sebagai grafik aliran data atau *bubble chart.*

DFD dapat digunakan untuk menyajikan sebuah sistem atau perangkat lunak pada setiap tingkat abtraksi. Kenyataanya, DFD dapat dipartisipasi ke dalam tingkat-tingkat yang merepresentasikan aliran informasi yang bertambah dan fungsi ideal. Demikianlah, DFD memberikan suatu mekanisme agi pemodelan fungsional dan pemodelan aliran informasi. Untuk melakukannya, dilakukan prinsip analisis operasional yang kedua (seperti menciptakan sebuah model fungsional).



Gambar 2.11 Model Aliran Informasi

DFD tingkat 0, yang disebut juga dengan model sistem fundamentasi atau model konteks, mempresentasikan seluruh elemen sistem sebagai sebuah *bubble* tunggal dengan data *input* dan *output* yang ditunjukkan oleh anak panah yang masuk dan keluar secara berurutan. Proses tambahan (*bubble*) dan jalur aliran informasi direpresentasikan pada saat DFD tingkat 0 dipartisi untuk mengungkap detail yang lebih. Contohnya, sebuah DFD tingkat 1 dapat berisi lima atau enam *bubble* dengan anak panah yang saling menghubungkan. Setiap proses yang direpresentasikan pada tingkat 1 merupakan subfungsi dari seluruh sistem yang digambarkan di dalam model konteks.

Notasi dasar yang digunakan untuk menciptakan suatu DFD diilustrasikan di dalam Gambar 2.9 sebuah persegi panjang digunakan untuk merepresentasikan sebuah entitas *eksternal*, yaitu sebuah elemen sistem (misalnya perangkat keras, seseorang, program yang lain) atau sistem yang lain yang menghasilkan informasi bagi transformasi oleh perangkat lunak atau menerima informasi yang dihasilkan oleh

perangkat lunak.lingkaran mempresentasikan sebuah proses atau tranformasi yang diaplikasikan kedata (atau kontrol) dan mengubahnya dengan berbagai cara. Anak panah melambangkan satu atau lebih data. Garis dobel merepresentasikan sebuah penyimpanan data, informasi tersimpan yang digunakan oleh perangkat lunak. Penyingkatan notasi DFD merupakan satu alasan mengapa teknik analisis terstruktur paling banyak digunakan.

Penting untuk diketahui bahwa tidak ada indikasi eksplisit dari urutan pemrosesan yang didukung oleh diagram tersebut. Prosedur atau urutan dapat menjadi implisit di dalam diagram, tetapi representasi prosedural biasanya ditunda sampai desain perangkat lunak.

2.6 Interaksi Manusia Dengan Komputer

Prinsip kerja dalam sebuah sistem komputer adalah memproses (*process*) masukan (*input*) menjadi keluaran (*output*). Ketika pengguna bekerja dengan sebuah komputer, maka pengguna tersebut akan melakukan interaksi dengan komputer dengan menggunakan cara-cara tertentu. Cara yang umum digunakan adalah pengguna memberikan suatu perintah kepada komputer dan komputer menanggapinya dengan mencetak atau menuliskan tanggapan itu pada layar tampilan.

Salah satu kriteria yang harus dimiliki oleh sebuah perangkat lunak untuk mendapatkan predikat “ramah dengan pengguna atau *user friendly*” adalah perangkat lunak itu mempunyai antarmuka yang bagus, mudah dioperasikan, mudah dipelajari dan pengguna selalu senang untuk menggunakan perangkat lunak tersebut.

Antarmuka suatu perangkat lunak menggambarkan bagaimana cara sebuah program akan dilihat dan digunakan oleh pengguna. Perancangan sebuah antarmuka termasuk didalamnya mendefinisikan screens, keluaran printer, dan struktur menu.

Konsistensi tampilan dilayar harus dipertahankan, tombol (*keyboard*) yang harus dipergunakan oleh pengguna terjangkau dan mudah pengoperaiannya, artinya setiap pengguna perangkat lunak tidak perlu melakukan pelatihan khusus dalam pengoperasiannya.