MODUL

PRAKTIKUM SISTEM INFORMASI MANAJEMEN

Disusun Oleh :
Shoffin Nahwa Utama, M.T

Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Darussalam Gontor
2016

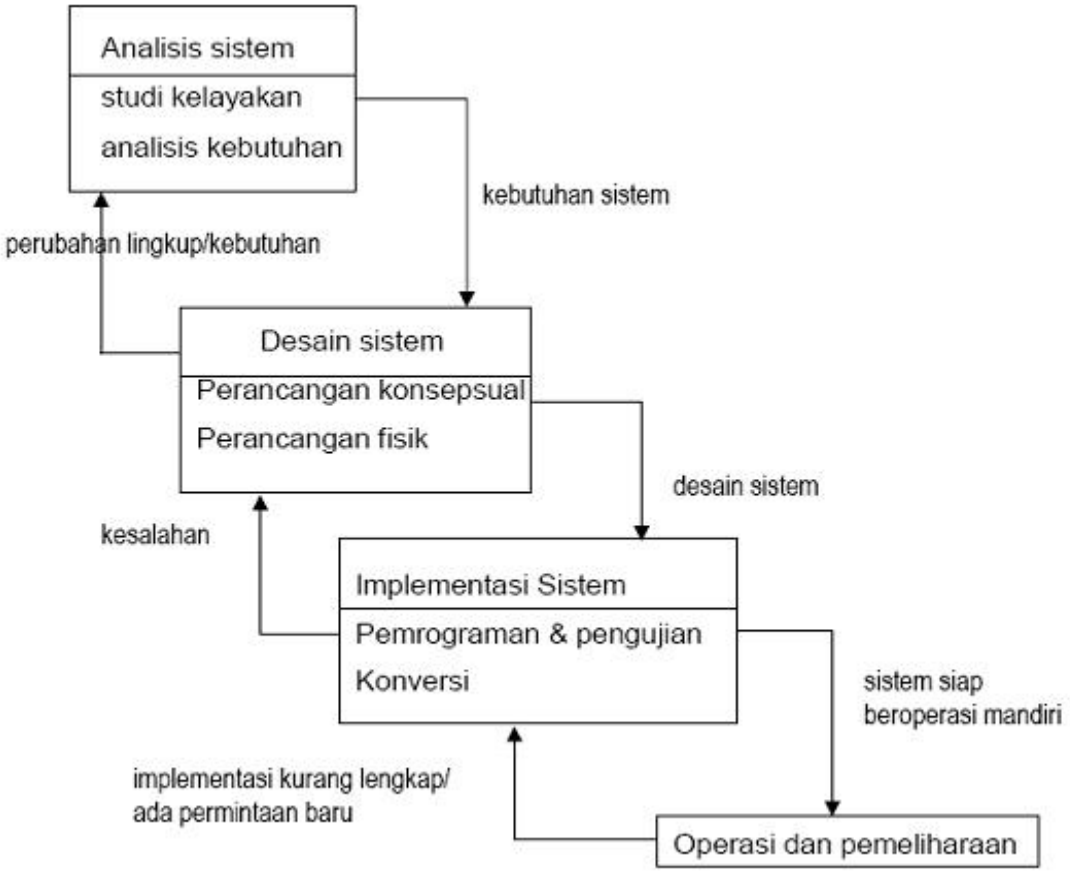
DISUSUN OLEH

SHOFFIN NAHWA UTAMA, M.T

DAFTAR ISI

PENDAHULUAN

Pada praktikum sistem informasi manajemen kali ini, kita akan mencoba merancang sebuah sistem informasi berdasarkan studi kasus yang berada dilingkungan pondok Gontor dan Unida Gontor. Metodologi pengembangan sistem pada praktikum kali ini akan menggunakan daur hidup pengembangan sistem (SDLC = *System Development Life Cycle*)



Gambar 1. Tahapan Dalam SDLC

Diharapkan hasil akhir dari praktikum sistem informasi manajemen dapat menghasilkan sebuah produk SIM yang dibangun berdasarkan alur pengembangan sebuah sistem.

Modul 1
Pengembangan Sistem Informasi

1.1 Kompetensi Dasar

Mahasiswa dapat :

1. Memahami alur pengembangan system (SDLC)
2. Membuat analisis Sistem meliputi studi kelayakan dan analisis masalah
3. Mencari studi kasus untuk analisa sistem informasi

1.2 Analisa Sistem

Analisa sistem dapat didefinisikan sebagai penguraian dari suatu system informasi yang utuh ke dalam bagian-bagian komponennya dengan maksud untuk mengidentifikasikan dan mengevaluasi permasalahan-permasalahan, kesempatan-kesempatan, hambatan-hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan-perbaikannya

1.3 Langkah Analisa sistem

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

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

1.4 MENGIDENTIFIKASI MASALAH

Mengidentifikasi (mengenal) masalah merupakan langkah pertama yang dilakukan dalam tahap analisis sistem. Masalah dapat didefinisikan sebagai suatu pertanyaan yang diinginkan untuk dipecahkan. Masalah inilah yang menyebabkan sasaran dari system tidak dapat dicapai. Oleh karena itu langkah pertama yang harus dilakukan oleh analis sistem adalah mengidentifikasi terlebih dahulu masalah-masalah yang terjadi.

Tugas analis system Dalam Mengidentifikasi Masalah adalah :

1. **Mengidentifikasi Penyebab Masalah**

   Analis sistem harus mempunyai pengetahuan yang cukup tentang aplikasi yang sedang dianalisisnya. Untuk aplikasi bisnis, analis sistem perlu mempunyai pengetahuan tentang sistem bisnis yang diterapkan di organisasi, sehingga dapat mengidentifikasi penyebab terjadinya masalah ini.

   Tugas mengidentifikasi penyebab masalah dimulai dengan mengkaji ulang terlebih dahulu subyek permasalahan yang telah diutarakan oleh manajemen atau yang telah ditemukan oleh analis sistem ditahap perencanaan sistem.

2. Mengidentifikasi Titik Keputusan

Setelah penyebab terjadinya masalah dapat diidentifikasi, selanjutnya juga harus mengidentifikasikan titik keputusan penyebab masalah tersebut. Titik keputusan menunjukkan suatu kondisi yang menyebabkan sesuatu terjadi.

Analis sistem bila telah dapat mengidentifikasi terlebih dahulu titik-titik keputusan penyebab masalah, maka dapat memulai penelitiannya dititik-titik keputusan tersebut. Sebagai dasar identifikasi titik-titik keputusan ini, dapat digunakan dokumen paperwork flow atau form flowchart bila dokumentasi ini dimiliki oleh perusahaan.

3. Mengidentifikasi Personil-personil Kunci

Setelah titik-titik keputusan penyebab masalah dapat diidentifikasi beserta lokasi terjadinya, maka selanjutnya yang perlu diidentifikasi adalah personil-personil kunci baik yang langsung maupun yang tidak langsung dapat menyebabkan terjadinya masalah tersebut. Identifikasi personil-personil kunci ini dapat dilakukan dengan mengacu pada bagan alir dokumen perusahaan serta dokumen deskripsi kerja (job description).

1.5 Tugas Praktikum !

1. Dari tahapan diatas buatlah identifikasi masalah dari studi kasus pada unit usaha di Gontor.
2. Susunlah dalam bentuk laporan yang sudah diketik rapi

Modul 2
Analisa Kebutuhan Sistem

2.1 Kompetensi Dasar

Mahasiswa mampu :

1. Memahami alur pengembangan system
2. Membuat analisa kebutuhan Sistem berdasarkan studi kasus yang didapatkan

2.2 Requirement gathering

Tahap pertama adalah Requirement gathering, yaitu mengumpulkan spesifikasi kebutuhan sebuah perangkat lunak dari semua stakeholder yang terlibat. Stakeholder adalah semua orang/organisasi yang terlibat dalam pengembangan sistem mulai dari top manager sampai end user. Proses ini bisa jadi sangat melelahkan tergantung seberapa besar dari sistem yang akan dibangun.

Requirement bisa diperoleh dari wawancara atau investigasi terhadap business processyang berjalan. Jika perusahaan sudah memiliki SOP Standar Operation Procedure) yang jelas, investigasi bisa dilakukan secara bottom up dengan cara menganalisis laporan yang ingin dihasilkan.

Jika tidak ada SOP yang baku mau tidak mau kita lakukan pendekatan top-down, dengan cara wawancara semua stakeholder. Setelah requirement terkumpul barulah dimulai tahap analisis sistem. Pada tahap ini menghasilkan dokumen spesifikasi kebutuhan perangkat lunak atau disebut juga SRS (Software Requirement Spesiifcation). Proses analisis dilakukan oleh sorang atau lebih system analyst.

Analisis kebutuhan dimaksudkan untuk menghasilkan spesifikasi kebutuhan, yaitu spesifikasi rinci tentang hal-hal yang akan dilakukan oleh sistem ketika diimplementasikan. Analisis kebutuhan diperlukan, karena dengan adanya analisis kebutuhan diharapkan dapat untuk menentukan :

1. Masukan yang diperlukan sistem
2. Keluaran yang dibutuhkan
3. Lingkup proses
4. Volume data yang ditangani sistem
5. Kategori pemakai sistem
6. Kontrol sistem

2.3 Kebutuhan fungsional

Contoh: Pengisian Formulir Pendaftaran

Calon Santri	Panitia	Sistem Informasi
Minta formulir	Memberi Formulir	Menampilkan Formulir
Mengisi Formulir	-	Entry Data
Melampirkan berkas-berkas	-	Upload berkas
Mengembalikan formulit	Menerima formulir dan berkas	Submit
-	Mengecek berkas	Mengecek kelengkapan berkas
-	Membuat kartu seleksi	Menampilkan kartu seleksi
Menerima kartu seleksi	Memberikan kartu seleksi	Cetak kartu seleksi

Lakukan analisis terhadap scenario di atas tentang:

- Apa
- Siapa
- Dimana
- Kapan
- Bagaimana

2.4 Kebutuhan nonfungsional

Lakukan analisis terhadap 5 komponen Sistem Informasi:

1. Hardware
2. Software
3. Brainware
4. Data, Informasi, Pengetahuan
5. Network

Modul 3
IDENTIFIKASI & DESAIN OUTPUT

3.1 Kompetensi Dasar

Mahasiswa mampu :

1. Memahami proses identifikasi dan desain output
2. Membuat desain output

3.2 Identifikasi output

Contoh tabel identifikasi :

Nama laporan	**Bentuk laporan**	**Periode laporan**	**Alat untuk menampilkan laporan**	**Pembuat laporan**	**Penerima laporan**	**Data/informasi yang ditampilkan**	**Deskripsi laporan**
Laporan kesehatan santri	Tabel, grafik	bulanan	Monitor,printer	Staf kesehatan	Kepala staf kesehatan, kepala asrama	Jumlah santri yang sakit dalam 1 bulan, prosentase santri yang sakit dan sehat dalam satu bulan	Laporan ini memberikan informasi tentang keadaan kesehatan santri secara umum pada setiap bulan
Dst…							

3.3 Desain Output

Menggambarkan bentuk atau layout dari laporan, baik tabel, grafik, diagram, formulir, dan lain sebagainya.

Jadwal Perkuliahan Semester Genap Tahun Akademik 2015/2016 Prodi Teknik Informatika Unida Gontor						
no	Hari	Jam/waktu	Matakuliah	Dosen pengampu	SKS	Ruang

Tugas!

Buatlah tabel identifikasi output dan desain output seperti contoh diatas sesuai dengan projek yang anda kerjakan!

Modul 4
Identifikasi dan Desain Input

4.1 Kompetensi Dasar

Mahasiswa mampu :

1. Memahami proses identifikasi dan desain input
2. Membuat desain input

Langkah-langkahnya sebagai berikut

4.2 Identifikasi Input

Nama input	Alat untuk entry data	Bentuk input	Yang menyediakan data	Yang mengentry data	Periode input	Deskripsi input

4.3 Desain Input

Dalam tahap ini, desainer harus membuat layout interface yang akan digunakan untuk menginputkan data.

4.4 Source Document

Source document adalah dokumen-dokumen sumber yang digunakan dalam input data. Contoh source document berupa formulir isian untuk mendapatkan data alumni:

NIM	:
Nama	:
Tempat tanggal lahir	:
Alamat sekarang	:
Angkatan	:
Tanggal lulus	:
Tanggal ijazah	:
No. Ijazah	:
Nama Ayah	:
Nama Ibu	:

Riwayat pekerjaan

No.	Nama instansi	Tanggal masuk	Tanggal keluar	Jabatan

TUGAS:

1. Lakukan identifikasi dan desain input terhadap proyek yang akan Saudara kembangkan!
2. Dikumpulkan dalam bentuk laporan pr

MODUL 5
PENGENALAN APLIKASI POWER DESIGNER

Pada praktikum Sistem informasi Manajemen kali ini kita akan memakai aplikasi power designer untuk mendesain DFD mulai dari Context diagram. *Power Designer* adalah perangkat lunak buatan *Sybase* yang dibuat untuk membantu dalam perancangan sistem informasi. Namun untuk keperluan yang paling sering digunakan adalah PDPA (*Power Designer Process Analyst*) dan PDDA (*Power Designer Data Architect*). Perangkat lunak yang digunakan adalah *Power Designer* versi 6 meskipun sampai tulisan ini ditulis *Sybase* telah mengeluarkan *Power Designer* versi 12.5.

Power Designer Process Analyst (PDPA) digunakan untuk membantu dalam proses penggambaran *data flow diagram* mulai dari *context diagram*. Kelebihan dari perangkat lunak ini adalah dapat membantu untuk memeriksa apakah model yang dibuat sudah valid atau belum dan dapat langsung di-*generate* menjadi bentuk *Entity Relationship Diagram*.

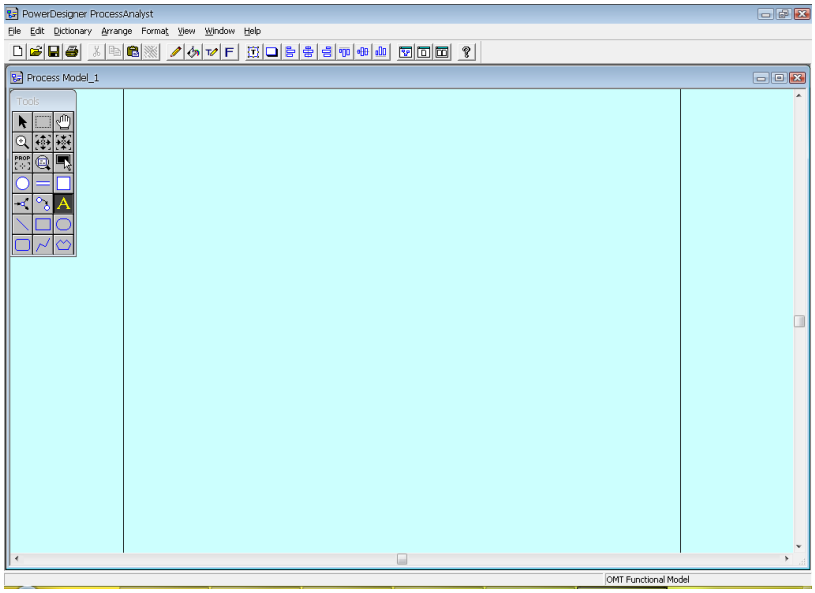
Power Designer Data Architect (PDDA) digunakan untuk membantu dalam penggambaran *entity relationship diagram*. PDDA ini meng-*import* data dari *data flow diagram* yang telah dibuat dengan PDPA. PDDA ini akan meng-*import* semua *datastore* yang telah dibuat di *data flow diagram.*

Langkah-langkah penggunaan PDPA dan PDDA akan dijelaskan di masing-masing subbab di bawah ini. Sebelum memulai, pastikan dahulu bahwa kedua program tersebut telah diinstalasi di komputer. Bagi yang belum memiliki, program dapat diunduh di Laboratorium Internet.

1.1. POWER DESIGNER PROCESS ANALYST

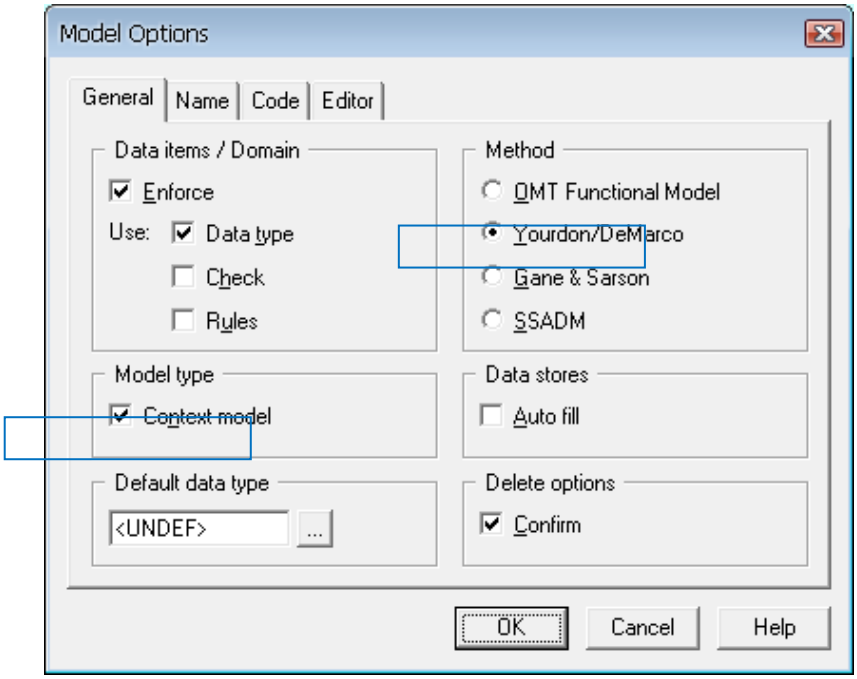
Sebelum mempelajari cara penggunaan PDPA, ada baiknya jika terlebih dahulu mengenal *interface* dari PDPA ini. *Interface* untuk program PDPA dapat dilihat pada gambar 1. Penjelasan untuk bagian *interface* ini hanya pada bagian–bagian penting yang biasa digunakan untuk membuat *data flow diagram*.

Ada hal yang perlu diketahui pada Power Designer bahwa setiap objek memiliki *code* dan *name*. *Name* adalah nama atau label yang akan ditampilkan pada objek, sedangkan *object* adalah identitas objek itu sendiri sehingga harus unik (tidak boleh ada yang sama dalam satu proyek). Pada umumnya, *user* memberikan nama pada objek, kemudian *code* disamakan dengan nama. Caranya menekan tombol (=) pada baris *code*. Contohnya, lihat gambar 3.



Gambar 1 *Interface Power Designer Process Analyst*

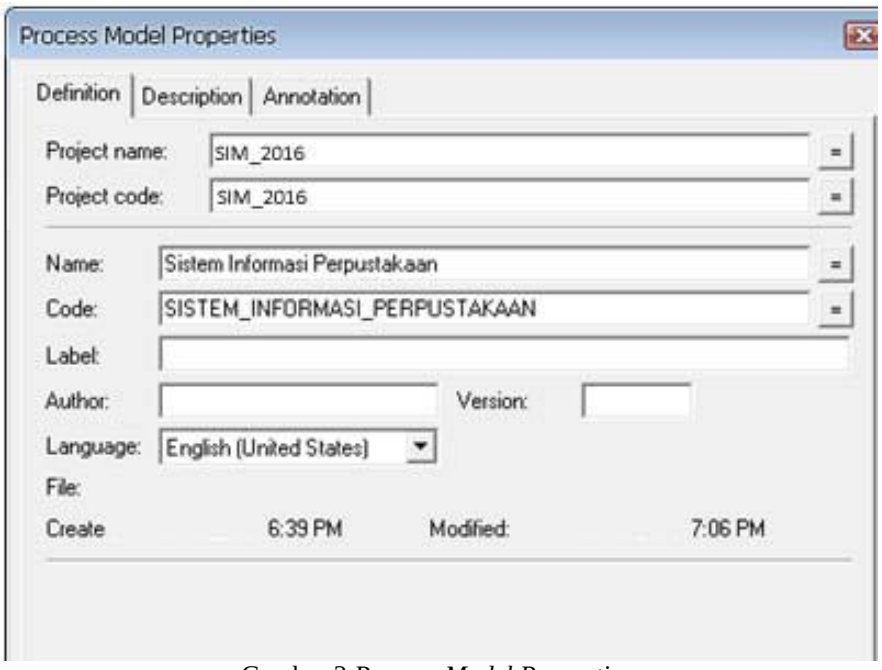
Sebelum mulai menggambar, hal yang lebih dahulu perlu dilakukan adalah menentukan terlebih dahulu jenis model yang akan dibuat. Oleh karena itu, bukalah **"Model Options"** dengan cara membuka menu **File → Model Options**. Jendela yang muncul dapat dilihat pada gambar 2. Yang diubah hanyalah tipe model apakah akan membuat *context diagram* atau tidak; dan metode penggambaran yang digunakan. Yang umum digunakan adalah **Yourdon/DeMarco dan Gane&Sarson**. Pelajari kembali perbedaan kedua metode ini.



Gambar 2 *Model Option*

Setelah menentukan properti dari model tersebut, langkah selanjutnya adalah memberikan status untuk proyek yang akan dibuat, yaitu mengisikan nama proyek, nama model, dan pembuat model tersebut. Caranya buka menu **Dictionary → Model Properties**

dan jendela yang muncul dapat dilihat pada gambar 3. Coba perhatikan, *code* selalu tidak ada spasi dan tidak boleh ada spasi.



Gambar 3 *Process Model Properties*

Setelah properti model dan opsi model diset sesuai dengan keinginan penggambar, baru penggambaran *context diagram* dan *data flow diagram* dapat dilakukan. Penggambaran *context diagram* dan dekomposisi proses akan dijelaskan pada sub bab tersendiri berikut ini.

Context Diagram (CD)

Contex Diagram merupakan pendekatan terstruktur yang mencoba untuk menggambarkan sistem pertama kali secara garis besar (disebut dengan *top level*) dan memecah-mecahnya menjadi bagian yang lebih terinci. *Contex diagram* ini menggambarkan hubungan *input/output* antara sistem dengan kesatuan luar.

Context Diagram adalah bagian dari *Data Flow Diagram* (DFD) yang berfungsi memetakan model lingkungan, yang dipresentasikan dengan lingkaran tunggal yang mewakili keseluruhan sistem. CD menyoroti sejumlah karakteristik penting sistem, yaitu:

1. Kelompok pemakai, organisasi atau sistem lain di mana sistem melakukan komunikasi (sebagai terminator).
2. Data masuk, yaitu data yang diterima sistem dari lingkungan dan harus diproses dengan cara tertentu.
3. Data keluar, yaitu data yang dihasilkan sistem dan diberikan ke dunia luar.
4. Penyimpanan data *(storage)*, yaitu digunakan secara bersama antara sistem dengan terminator. Data ini dapat dibuat oleh sistem dan digunakan oleh lingkungan atau sebaliknya dibuat oleh lingkungan dan digunakan oleh sistem. Hal ini berarti pembuatan simbol data storage dalam CD dibenarkan, dengan syarat simbol tersebut merupakan bagian dari dunia diluar sistem.
5. Batasan, antara sistem dan lingkungan.

Aliran dalam CD memodelkan masukkan ke sistem dan keluaran dari sistem, seperti halnya sinyal kontrol yang diterima atau dibuat sistem. Aliran data hanya digambarkan jika diperlukan untuk mendeteksi kejadian dalam lingkungan dimana sistem harus memberikan respon atau membutuhkan data untuk menghasilkan respon. Selain itu aliran data dibutuhkan untuk menggambarkan transportasi antara sistem dan terminator. Dengan kata lain aliran data digambarkan jika data tersebut diperlukan untuk menghasilkan respon pada kejadian tertentu.

Dalam hal ini seharusnya menggambar dengan asumsi bahwa masukan disebabkan dan diinisialisasi oleh terminator, sedangkan keluaran disebabkan dan diinisialisasi oleh sistem. Hal itu dilakukan dengan mencegah interaksi yang tidak perlu (extraneous prompts) yang berorientasi pada implementasi masukan-keluaran, dan mengkonsentrasikan pemodelan pada aliran data yang esensial saja.

CD dimulai dengan penggambaran terminator, aliran data, aliran kontrol, penyimpanan, dan proses tunggal yang mempresentasikan keseluruhan sistem. Bagian termudah adalah menetapkan proses yang hanya terdiri dari satu lingkaran dan diberi nama yang mewakili sistem. Nama harus dapat menjelaskan proses.

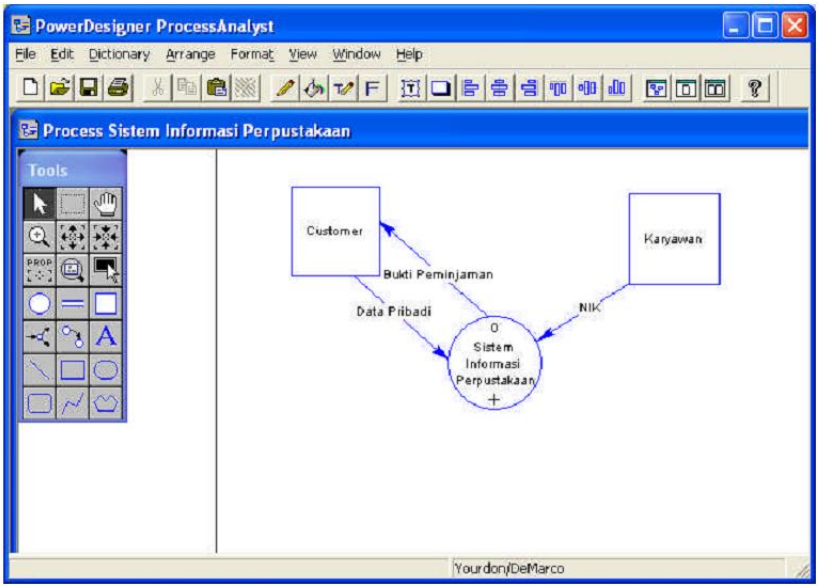
Langkah yang dapat membantu dalam menggambarkan CD:

1. Identifikasikan seluruh informasi yang dibutuhkan.
2. Identifikasikan seluruh data yang dibutuhkan proses/informasi.
3. Identifikasikan seluruh tujuan setiap informasi bagi penggunanya.
4. Identifikasikan seluruh sumber data yang dibutuhkan proses/informasi

Cara menggambar *context diagram* dilakukan dengan meng-klik salah satu tool yang ada kemudian menempatkannya di lembar kerja PDPA. Langkah pertama, tempatkan dahulu satu buah proses yang merupakan *context diagram* dan semua entitas eksternal yang ada.

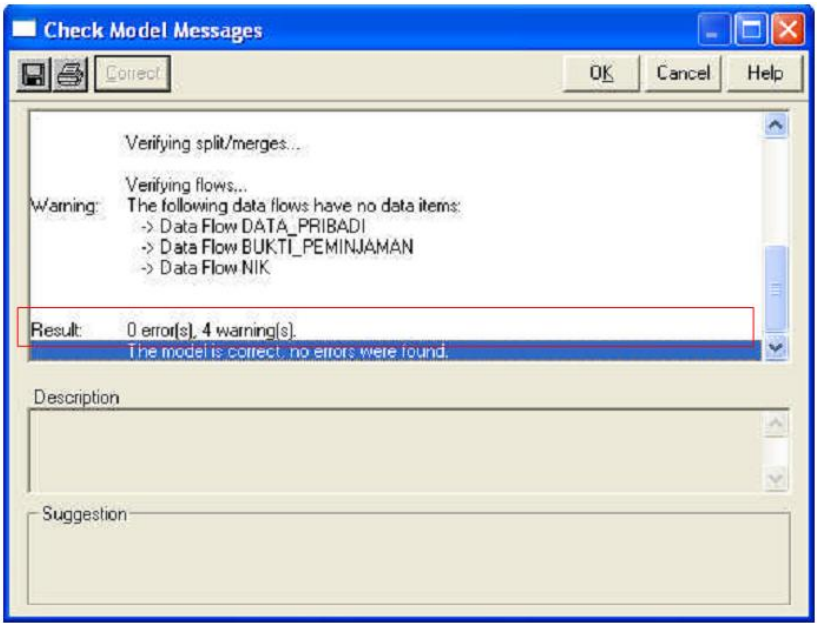
Langkah kedua adalah mengisikan properti untuk masing-masing entitas eksternal dan proses. Caranya adalah dengan meng-*double-click* objek yang akan diubah. Berikan nama dan *code* yang sesuai dengan keinginan namun merepresentasikan objek tersebut.

Langkah ketiga adalah menghubungkan entitas eksternal dan proses dengan menggunakan aliran (*flow tool*). **Ingat**, untuk menghubungkannya, tariklah garis dari tengah objek ke tengah objek yang lainnya. Jangan menarik garis dari luar objek karena program tidak akan mengenali sumber atau tujuan aliran data tersebut. Contoh penggambaran *context diagram* dapat dilihat pada gambar 4.



Gambar 4 Contoh *Context Diagram*

Langkah keempat adalah mengevaluasi model tersebut apakah penggambaran model tersebut telah sesuai dengan ketentuan yang berlaku dalam penggambaran objek. Caranya adalah dengan membuka menu **Dictionary → Check Model** atau menekan tombol **F4**. Akan muncul sebuah jendela yang menunjukkan berapa banyak jumlah error yang ditemukan dan berapa banyak warning yang diberikan pada bagian *RESULT* di paling akhir (Gambar 5). Keterangan error dan warning dapatdilihat pada bagian atasnya mengapa error dan apa yang harus diperhatikan, namun perhatikan hanya bagian error saja.



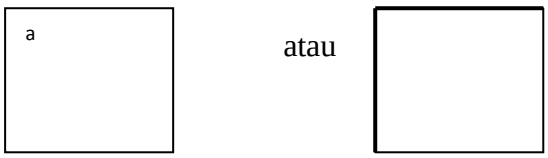
Gambar 5 *Check Model Massage*

Langkah-langkah yang telah diuraikan adalah langkah– langkah untuk membuat *context diagram*. Pada context diagram hanya dapat dibuat satu buah proses saja, sedangkan pada breakdown proses dapat dibuat lebih dari satu buah proses.

Data Flow Diagram (DFD)

DFD sering digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika tanpa mempertimbangkah lingkungan fisik dimana data tersebut mengalir atau lingkungan fisik dimana data tersebut akan disimpan. DFD merupakan alat yang digunakan pada metodologi pengembangan sistem yang terstruktur. Simbol-simbol yang digunakan dalam menggambarkan DFD adalah sebagai berikut.

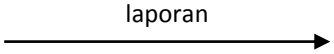
1. *External entity*, merupakan kesatuan di lingkungan luar sistem yang dapat berupa orang, organisasi, atau sistem lainnya yang berada di lingkungan luarnya yang akan memberikan *input* atau menerima *output* dari sistem. Kesatuan luar dapat disimbolkan dengan suatu notasi kotak atau suatu kotak dengan sisi kiri dan atasnya berbentuk garis tebal dan juga dapat diberi identifikasi dengan huruf kecil di ujung kiri atas yaitu sebagai berikut:



Gambar 6 Simbol *External Entity* DFD

2. Arus data, yaitu diberi simbol suatu anak panah. Arus data ini mengalir diantara proses, simpanan data, dan kesatuan luar serta menunjukkan arus dari data yang dapat berupa masukan untuk sistem atau hasil dari proses sistem dan dapat berbentuk sebagai berikut :
    a. Formulir atau dokumen yang digunakan di perusahaan.
    b. Laporan tercetak yang dihasilkan oleh sistem.
    c. Tampilan atau *output* di layar komputer yang dihasilkan oleh sistem.
    d. Masukan untuk komputer.
    e. Komunikasi ucapan.
    f. Surat-surat atau memo.
    g. Data yang dibaca atau direkamkan ke suatu file.
    h. Suatu isian yang dicatat pada buku agenda.
    i. Transmisi data dari suatu komputer ke komputer lain.

    Arus data sebaiknya diberi nama yang jelas dan mempunyai arti. Nama dari arus data dituliskan di samping garis panahnya, yaitu :



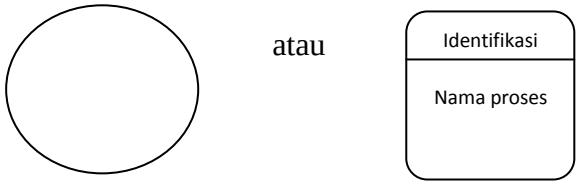
Gambar 7 Simbol Arus Data DFD

3. Proses, adalah kegiatan atau kerja yang dilakukan oleh orang, mesin atau komputer dari hasil suatu arus data yang masuk ke dalam proses untuk dihasilkan arus data yang akan keluar dari proses. Untuk *physical data flow diagram* (PDFD), proses dapat dilakukan oleh orang, mesin, atau komputer, sedangkan untuk *logical data flow diagram* (LDFD), suatu proses hanya menunjukkan proses dari komputer.

   Proses dipresentasikan dalam bentuk lingkaran *(circle)* atau bujursangkar dengan sudut melengkung *(a rounded rectangle)*. Setiap proses ditandai dengan nomor. Nomor berfungsi menjelaskan tingkatan proses dari *hierarchy chart*. Setiap proses dinamai dengan sepasang kata kerja yang simpel, contoh: *screen customer-order, record customer-order*.

   Dalam physical DFD, lokasi atau program yang memproses seringkali dituliskan dibagian bawah simbol, tetapi dalam *logical* DFD tidak perlu disebutkan. Untuk menunjukkan transformasi dari masukan menjadi keluaran, dalam hal ini sejumlah masukan dapat menjadi hanya satu keluaran ataupun sebaliknya. Proses umumnya didefinisikan dengan kata tunggal, atau kalimat sederhana. Fokus simbol ini adalah apa yang dikerjakan atau tindakan yang dilakukan (proses), bukan orang atau melakukan kegiatan apa.

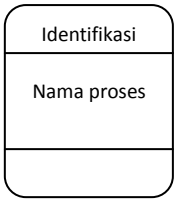
   Suatu proses dapat ditunjukkan dengan symbol sebagai berikut:



Gambar 8 Simbol Proses DFD

Setiap proses harus diberi penjelasan yang lengkap meliputi berikut ini:

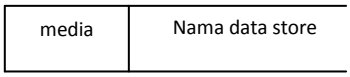
a. Identifikasi proses, umumnya berupa suatu angka yang menunjukkan nomor acuan dari proses dan ditulis pada bagian atas di simbol proses.
b. Nama proses, menunjukkan apa yang dikerjakan oleh proses tersebut.
c. Pemroses, untuk PDFD pemroses harus ditunjukkan siapa atau dimana suatu proses harus dilakukan. Untuk LDFD pemroses dapat tidak disebutkan. Keterangan pemroses ini dapat dituliskan di bawah nama proses sebagai berikut :



Gambar 9 Simbol Proses PDFD

4. Simpanan data (*data store*), merupakan simpanan dari data yang dapat berupa sebagai berikut :
   a. Suatu file atau data base dalam komputer.
   b. Suatu arsip atau catatan manual.
   c. Suatu kotak tempat data di meja seseorang.
   d. Suatu tabel acuan manual.
   e. Suatu agenda atau buku.

   Simpanan data di *DFD* dapat disimbolkan sebagai berikut :



Gambar 10 Simbol Simpanan Data DFD

Beberapa petunjuk untuk membuat DFD yang jelas, dan mudah dibaca:

1. Pilih nama yang jelas maksudnya (bagi proses, aliran, penyimpanan, dan terminator
   a. Untuk proses sebaiknya menggunakan nama yang mengacu pada fungsi, yaitu gabungan antara kata kerja yang spesifik dan obyek, misalnya memproses laporan inventori, validasi nomer telepon dan lain-lain. Untuk terminator lebih mengacu pada orang atau kelompok orang, sedangkan untuk aliran dan penyimpanan mengacu pada paket data atau informasi yang terkandung didalamnya.
   b. Jangan menggunakan nama yang terlalu umum, misalnya proses data, tangani masukan dan lain-lain.
   c. Gunakan nama yang familiar bagi pemakai.
2. Menomori proses untuk memperjelas sistematika.
   a. Tidak jadi masalah bagaimana urutan ditempatkan.
   b. Nomor tidak menunjukkan urutan.
   c. Penomoran dimaksudkan sebagai identifikasi proses dan memudahkan penurunan ke level yang lebih rendah atau ke proses berikutnya.
3. Menggambar kembali DFD hingga beberapa kali, sehingga cukup estetik.

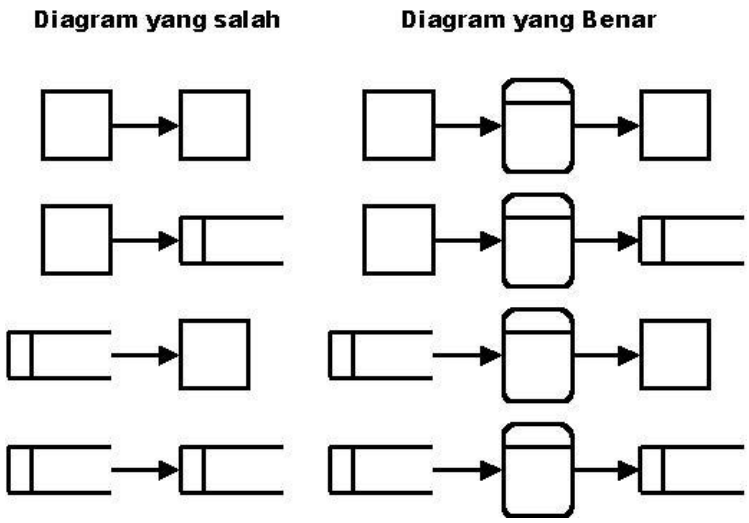
   Penggambaran kembali yang kadang-kadang bahkan lebih dari sepuluh kali digunakan untuk menjaga secara teknis gambar tersebut sudah benar, dapat diterima oleh pemakai, misalnya tingkat operasional yang akan mengoperasikan dan dapat diterima oleh pimpinan yang dalam hal ini menentukan strategi organisasi. Ketika penggambaran dilakukan ada beberapa hal yang perlu diperhatikan, yaitu:

a. Ukuran dan bentuk lingkaran sebaiknya tetap sama, karena jika tidak dapat menimbulkan kesan lingkaran yang lebih besar memproses sesuatu yang juga lebih besar.
b. Panah yang melengkung dan lurus tidak jadi masalah tetapi sebaiknya tidak menggunakan kedua cara tersebut pada gambar yang sama.
c. Menggambar dengan tangan atau menggambar dengan perangkat tertentu, tidak merupakan masalah. Menggambar dengan tangan seringkali memberikan sesuatu yang lain katakanlah sentuhan seni, tetapi menggambar dengan mesin, seringkali lebih memudahkan modifikasi.

4. Mencegah DFD yang terlalu kompleks dan tidak perlu. Kegunaan DFD bukan hanya menggambarkan fungsi dan interaksinya dalam sistem secara akurat tetapi juga untuk dibaca dan dimengerti oleh bukan hanya penganalisis sistem, tetapi juga pemakai yang berpengalaman dalam sistem yang dimodelkan. Hal ini berarti: jangan membuat DFD dengan terlalu banyak proses, aliran, penyimpanan dan terminator.
5. Menjamin konsistensi DFD tersebut secara intern ataupun yang berkualitas dengan DFD. Konsistensi dalam hal ini juga menyangkut konsistensi dengan model lain (misalnya: *entity relationship diagram, state transition diagram*, *data dictionary* dan *process specification*).

Hal-hal penting yang harus diperhatikan / diingat adalah:

a. Mencegah proses yang mempunyai masukan tetapi tidak mempunyai keluaran yang dikenal dengan lubang hitam.
b. Mencegah proses yang mempunyai keluaran tetapi tidak punya masukan, misalnya penghasil bilangan acak.
c. Hati-hati dengan aliran dan proses yang tidak dinamakan karena dapat mengakibatkan elemen data yang saling tidak berhubungan menjadi satu.
d. Hati-hati dengan penyimpanan yang punya status hanya dapat dibaca atau hanya dapat ditulis dan berkaitan dengan proses yang hanya memproses masukan atau hanya memproses keluaran.

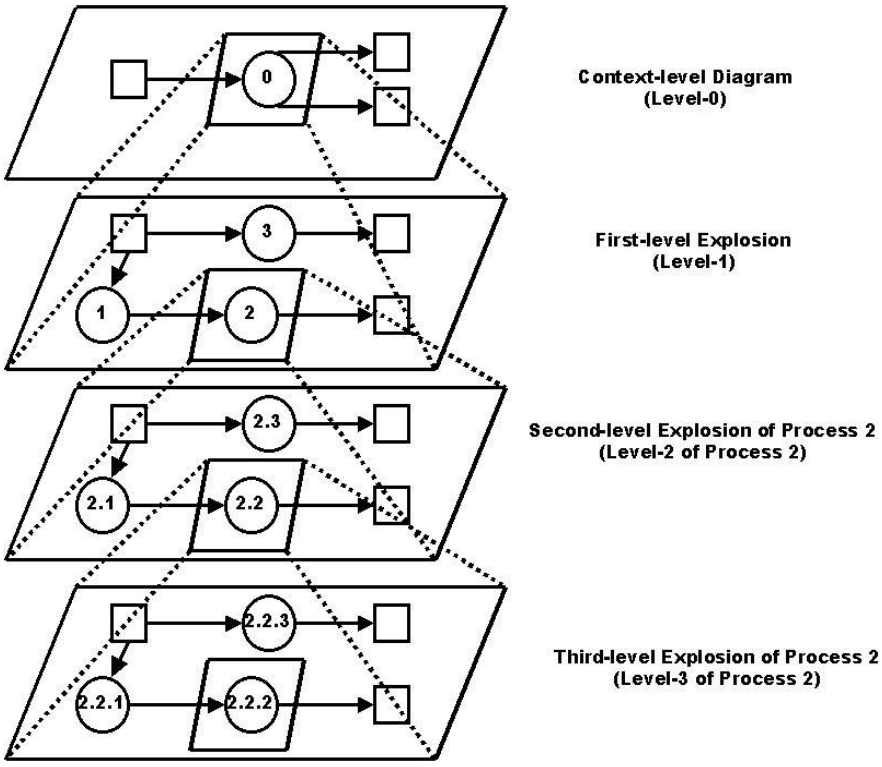


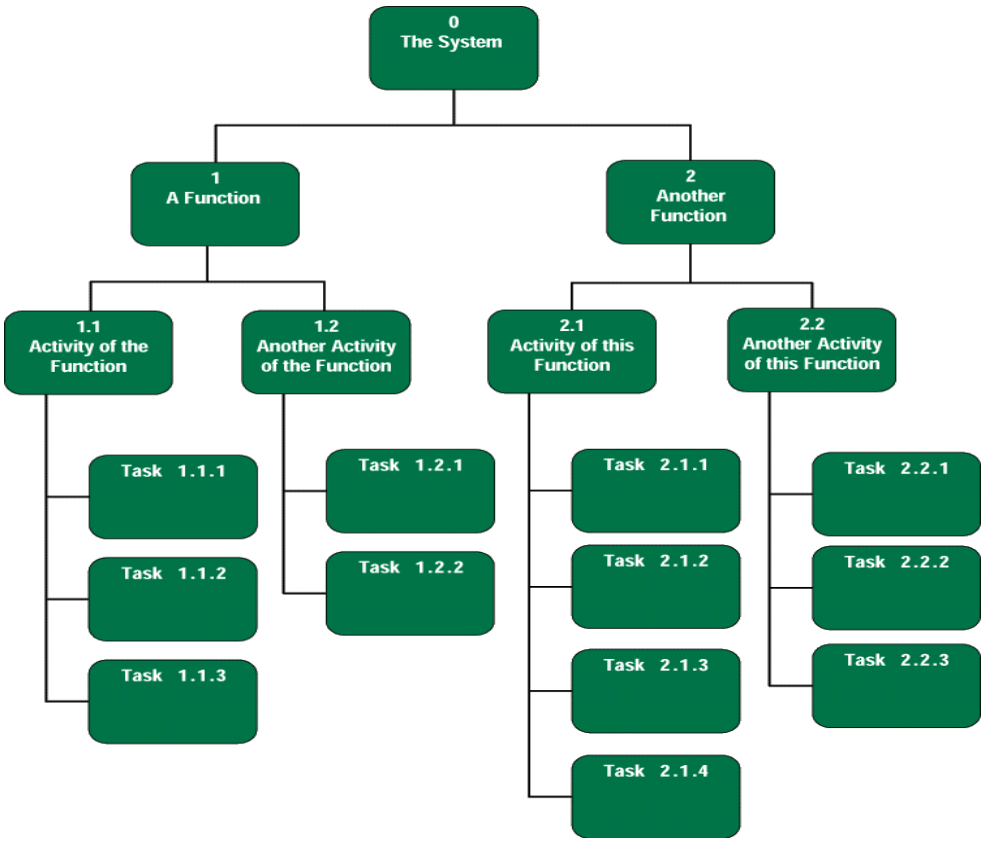
Gambar 11 Bentuk Penggambaran Diagram

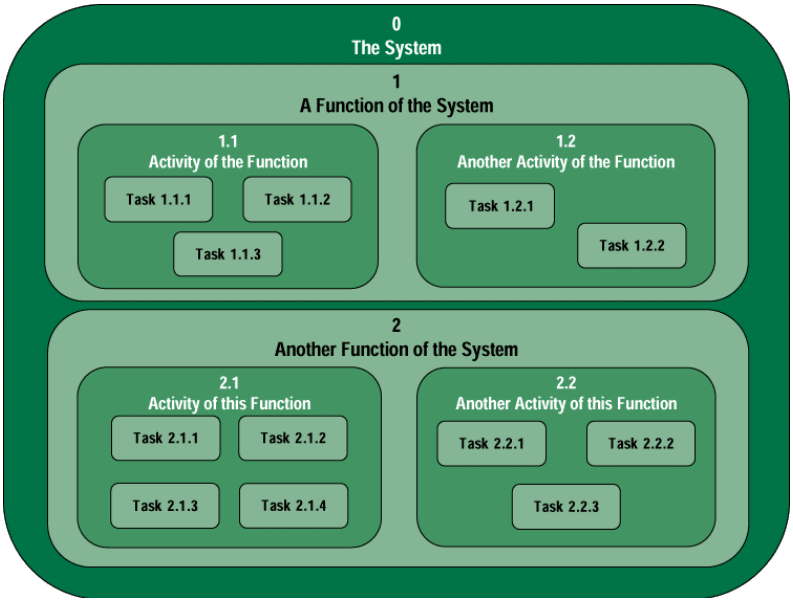
Sebagaimana analogi dalam bahasan sebelumnya yang membahas tentang peta dan penurunnya dalam peta yang lebih detail dan detil lagi, maka demikian juga dengan DFD. Level paling tinggi dalam suatu DFD hanya punya sebuah lingkaran (proses) yang memodelkan seluruh sistem, sedangkan aliran memodelkan hubungan antara sistem dengan terminal diluar sistem *(external terminator).* Level ini disebut *Context Diagram*. Dalam hal ini berperan pemberian nomor pada setiap lingkaran pada DFD yang berguna untuk memudahkan penurunan DFD ke level yang lebih rendah.

Penurunan ini mengacu pada status tertentu, yaitu:

a. Setiap penurunan ke level yang lebih rendah harus mampu mempresentasikan proses tersebut dalam spesifikasi proses yang jelas, sehingga seandainya belum cukup jelas maka seharusnya diturunkan ke level yang lebih rendah.
b. Setiap penurunan harus dilakukan hanya jika perlu.
c. Tidak semua bagian dari sistem harus diturunkan dengan jumlah level yang sama karena yang kompleks bisa saja diturunkan, dan yang sederhana mungkin tidak perlu diturunkan. Selain itu, karena tidak semua proses dalam level yang sama punya derajat kompleksitas yang sama juga.
d. Konfirmasikan DFD yang telah dibuat pada pemakai dengan cara top-down.
e. Aliran data yang masuk dan keluar pada suatu proses di level harus berhubungan dengan aliran data yang masuk dan keluar pada level x+1. Di mana level x+1 tersebut mendefinisikan sub proses pada level x tersebut.
f. Ketika mulai menurunkan DFD dari level tertinggi, cobalah untuk mengindentifikasi *external events* dimana sistem harus memberikan respon. *External events* dalam hal ini berarti suatu kejadian yang berkaitan dengan pengolahan data di luar sistem, dan menyebabkan sistem kita memberikan respon.


0
3
1
2
2.3
2.1
2.2
2.2.3
2.2.1
2.2.2
Context-level Diagram
(Level-0)
First-level Explosion
(Level-1)
Second-level Explosion of Process 2
(Level-2 of Process 2)
Third-level Explosion of Process 2
(Level-3 of Process 2)


0
The System
1
A Function
2
Another
Function
1.1
Activity of the
Function
1.2
Another Activity
of the Function
2.1
Activity of this
Function
2.2
Another Activity
of this Function
Task 1.1.1
Task 1.1.2
Task 1.1.3
Task 1.2.1
Task 1.2.2
Task 2.1.1
Task 2.1.2
Task 2.1.3
Task 2.1.4
Task 2.2.1
Task 2.2.2
Task 2.2.3



Gambar 12 Penurunan *Context Diagram*

DFD sebagai perangkat pemodelan bisa dikatakan sederhana, tetapi sangat berguna untuk memodelkan fungsi dalam sistem. Jika yang akan dimodelkan punya ketergantungan dengan waktu (misalnya *real time system*), maka harus digunakan tambahan model lain atau kadang-kadang tambahan notasi lain, artinya tidak bisa hanay dengan DFD. Tetapi ironisnya, banyak penganalisis sistem mengira hanya dengan DFD, mereka sudah tahu segala-galanya tentang analisis terstruktur. Pada kenyatannya telah dibuktikan bahwa tanpa perangkat pemodelan tambahan, DFD masih sesuatu yang serba kekurangan.

Perbedaan antara *Physical* DFD dan *Logical* DFD:

a. *Physical* DFD

Diagram phisik memiliki atribut yang menjelaskan tentang, lokasi di mana sebuah proses dilakukan, siapa yang melakukan proses tersebut, perangkat apa yang digunakan untuk melakukan proses tersebut. Atribut tersebut biasanya dituliskan pada bagian bawah simbol proses (bawah garis horisontal)

b. *Logical* DFD

Diagram logik hanya berisi data yang diarahkan menuju/dari sistem, yaitu data yang benar-benar dibutuhkan proses.

Dekomposisi Proses

Setelah membuat *context diagram*, langkah selanjutnya yang harus dilakukan adalah memecah proses tersebut menjadi proses yang lebih detil. Langkah ini yang disebut dengan dekomposisi proses.

Dekomposisi proses pada *Power Designer* dilakukan dengan memilih proses yang akan didekomposisi, kemudian tekan klik kanan *(right click)* proses tersebut kemudian pilih **Decompose**. Setelah pilihan tersebut dipilih, akan muncul jendela model proses yang baru seperti pada *context diagram*.

Pada saat sebuah proses didekomposisi, entitas–entitas dan data yang masuk ke proses dan keluar dari proses akan secara otomatis ditampilkan pada jendela yang baru tersebut. Jika sebuah entitas memberikan atau menerima data lebih dari satu buah, maka pada saat dekomposisi akan muncul entitas tersebut beberapa kali namun berbeda dengan entitas yang hanya ada satu kali. Perbedaannya adalah adanya gambar segitiga kecil di sudut kanan bawah entitas tersebut.

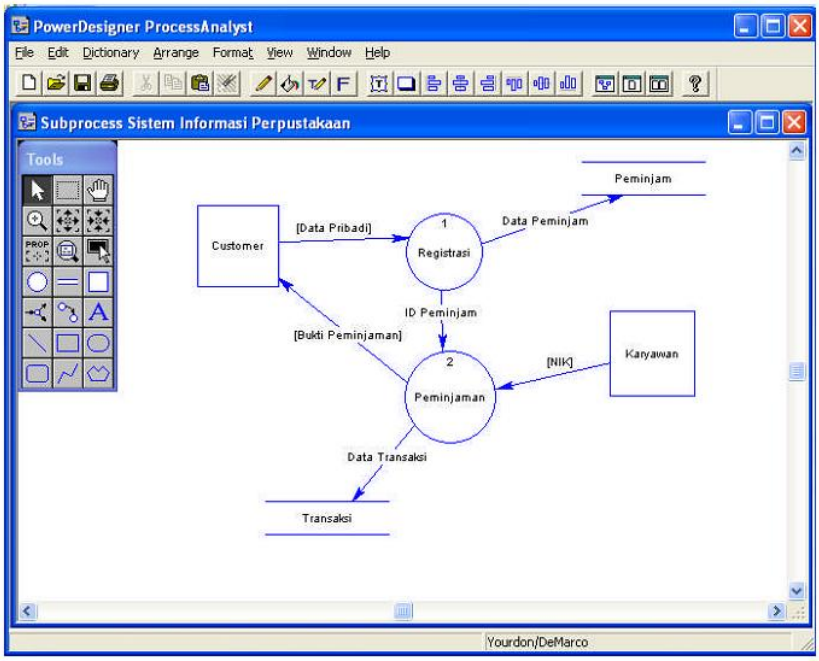
Hal yang perlu diperhatikan adalah jangan menghapus entitas maupun aliran data yang secara otomatis dibuat oleh *Power Designer* karena semua objek itu saling berhubungan dari awal sampai akhir. Untuk menjaga konsistensi pembuatan DFD ini, ikutilah beberapa langkah yang dijelaskan berikut ini.

Langkah pertama pembuatan dekomposisi proses ini adalah menambahkan proses-proses yang diperlukan di subproses ini. Langkah ini sama seperti saat membuat proses pada *context diagram*. Setelah ditempatkan, semua property dari objek diubah sesuai dengan nama proses yang telah direncanakan beserta code dari masing-masing proses tersebut.

Langkah kedua adalah menghubungkan entitas dan aliran data yang telah ditampilkan oleh program secara otomatis. Ingat baik-baik bahwa menghubungkan garis tersebut harus ke tengah objek untuk memastikan tidak ada aliran yang terputus. Setelah semua entitas dan aliran hasil dekomposisi dari proses sebelumnya dihubungkan, baru lanjutkan ke langkah berikutnya.

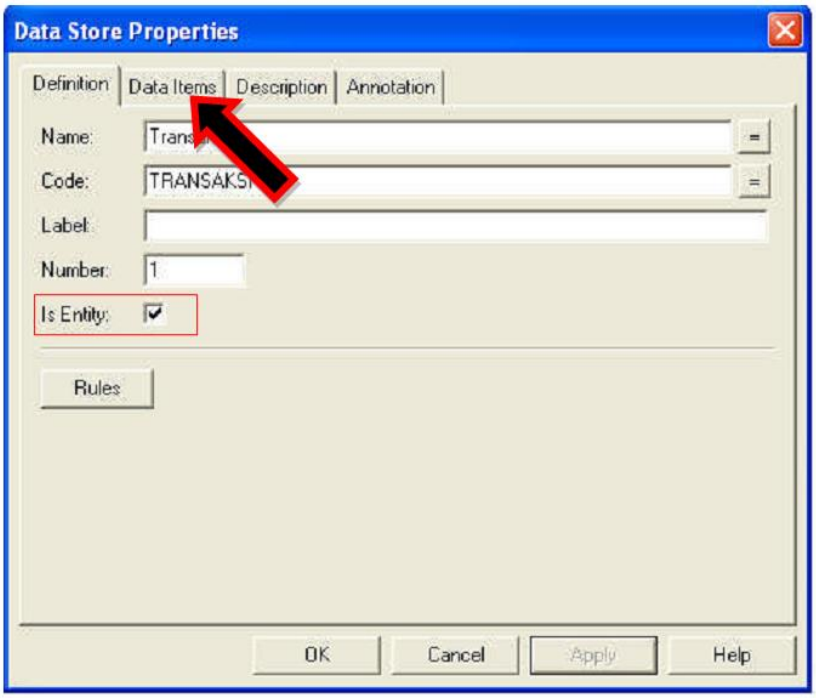
Langkah ketiga adalah membuat aliran antarproses yang belum berhubungan antara proses yang satu dengan yang lain. Kemudian isikan pula nama data yang mengalir dan code dari aliran tersebut. Jika ada entitas yang lebih dari satu kali muncul ingin dimunculkan hanya satu saja, atau dengan kata lain, ingin menampilkan entitasnya satu kali dengan aliran data yang banyak, hapuslah salah satu entitas tersebut, bukan menghapus aliran datanya. Untuk lebih jelasnya, lihat gambar 13 untuk entitas *customer*.

Langkah keempat adalah menambahkan data store jika memang diperlukan. Pada saat mengubah properti dari data store ini ada sedikit perbedaan dengan objek lain.



Gambar 13 Dekomposisi Proses

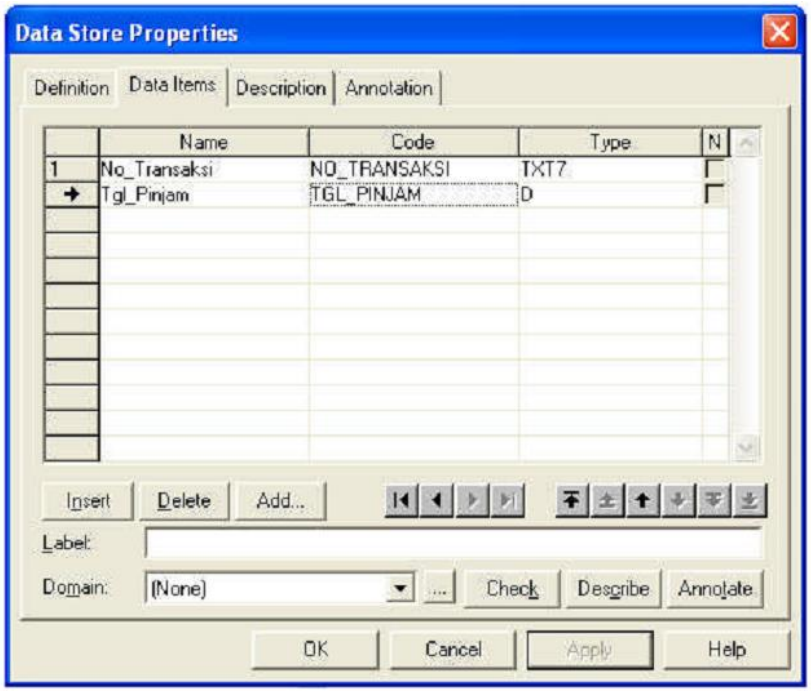
Cara memberikan properti untuk *data store* dilakukan dengan tiga tahap. Tahap pertama seperti cara memberikan properti pada objek lain, yaitu memberikan nama dan *code* untuk data store tersebut. Tahap yang kedua adalah memberikan tanda check (√) pada bagian **"Is Entity"** jika *data store* tersebut merupakan data yang akan disimpan ke dalam basis data *(database)*. Lihat contoh pada gambar 14 untuk penjelasan yang lebih baik.



Gambar 14 *Property Data Store*

Tahap ketiga adalah memilih tab **"Data Items"** untuk menentukan *field* apa saja yang diperlukan untuk entitas tersebut. Penentuan *field* apa saja pun sudah harus ditentukan pula

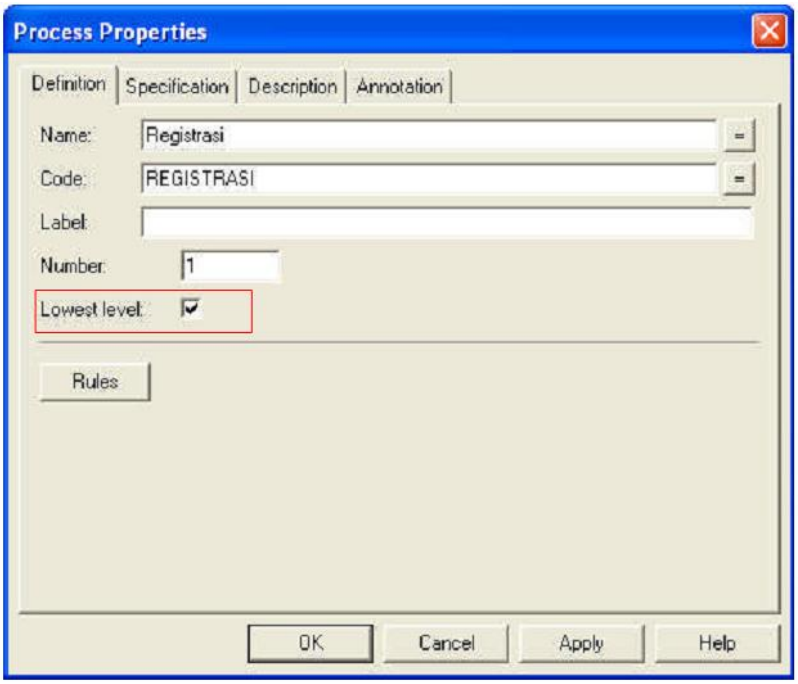
tipe dari *field* tersebut, apakah bertipe *text* atau *string* dengan lebar tertentu, bertipe *integer*, *date*, *time*, dsb.



Gambar 15 *Properti Data Store – Data Items*

Contoh yang dibuat pada gambar 15 adalah untuk *data store* Transaksi dengan keperluan *field* No_Transaksi dengan tipe *String*[7] dan *field* Tgl_Pinjam bertipe *Date*. Untuk keperluan pembuatan ERD selanjutnya, dibuat pula contoh untuk *data store* Peminjam dengan keperluan *field* ID_Konsumen bertipe *String*[7], Nama bertipe *String*[30], Alamat bertipe *String*[50], No_HP bertipe *String*[13], dan No_Telp bertipe *String*[13].

Setelah selesai proses untuk dekomposisi satu proses, lakukan dekomposisi untuk proses yang lainnya. Proses yang terdapat pada dekomposisi proses pun dapat didekomposisi lagi menjadi lebih detil. Jangan lupa, apabila sebuah proses sudah tidak memiliki dekomposisi lagi, beri tanda check (√) pada bagian **"Lowest Level"** seperti ditunjukkan pada gambar 16.



Gambar 16 Properti Proses

Setelah selesai pada sebuah subproses, lakukan hal yang sama untuk membuat subproses atau dekomposisi proses yang lainnya. Patuhi tahapan-tahapan yang telah diberikan untuk menjaga konsistensi pembuatannya, namun tidak menutup kemungkinan untuk mengubah apa yang telah dibuat. Jangan pula lupa lakukan pengecekan terhadap model tersebut dengan menekan tombol **F4** atau memilih menu **Dictionary → Check Model**.

Apabila seluruh proses telah dibuat dan didekomposisi, berarti pembuatan DFD telah selesai dilakukan. Langkah selanjutnya yang perlu dilakukan adalah dengan membuat *Entity Relationship Diagram* (ERD). Pastikan pula file DFD telah disimpan dengan baik di media penyimpanan tetap *(hard disk).*

Modul 6
IDENTIFIAKSI DAN DESAIN PROSES

Nama Proyek	:	
Nama Manajer Proyek	:	(ketua kelompok)
Dibuat oleh	:	(anggota yang membuat)
Tanggal dibuat	:	
Versi	:	

Tugas !

Buatlah identifikasi proses dan desain proses mulai arsitektur sampai dataflow diagram

1. Identifikasi proses

Nama proses	Deskripsi proses	Input proses	Output proses	Alur proses (jika perlu lampirkan flowchart programnya)
dr kebutuhan fungsional				

2. Arsitektur aplikasi
3. Data flow diagram level 1
4. Data flow diagram level 2
5. Data flow diagram level …

MODUL 7
IDENTIFIKASI DAN DESAIN DATABASE

Nama Proyek :
Nama Manajer Proyek : (ketua kelompok)
Dibuat oleh : (anggota yang membuat)
Tanggal dibuat :
Versi :

Tugas !

1. Identifikasi tabel database

Nama tabel	Nama field	Tipe data	Lebar	Key

2. Entity Relationship Diagram Conceptual
3. Entity Relationship Diagram Physical

MODUL 8
IDENTIFIKASI DAN DESAIN INTERFACE

Nama Proyek :
Nama Manajer Proyek : (ketua kelompok)
Dibuat oleh : (anggota yang membuat)
Tanggal dibuat :
Versi :

Tugas !

1. Identifikasi interface

Nama interface	**Jenis interface**	**Bentuk interface**	**Deskripsi interface**
	form,massage,dialog,pop up menu,tool bar,error screen,dll	Tampilan, suara,voice	

2. Desain interface

FINAL REPORT

Nama Proyek	:	
Nama Manajer Proyek	:	(ketua kelompok)
Anggota	:	(anggota yang membuat)
Tanggal dibuat	:	

HALAMAN PENILAIAN

Tanggal presentasi :

No.	Item penilaian	Nilai 1	Nilai 2
1.	Sistematika laporan		
2.	Penggunaan bahasa		
3.	Presentasi		
4.	Slide presentasi		
5.	Demo program		
Total Nilai			
Nilai akhir			

Penilai

Shoffin Nahwa Utama, M.T

CEK KELENGKAPAN

Laporan	ada / tidak ada
Slide Presentasi	ada / tidak ada
CD Program yang berisi :	
- Program aplikasi	ada / tidak ada
- Laporan	ada / tidak ada
- Rancangan DFD Power designer	ada / tidak ada
- Rancangan ERD Power designer	ada / tidak ada
- Slide presentasi	ada / tidak ada