BAB II

LANDASAN TEORI

Dalam merancang dan mengimplementasikan sistem ini serta pembuatan tugas akhir ini digunakan beberapa landasan teori, yaitu :

2.1 Biometrik

Biometrik merupakan suatu karakteristik-karakteristik sifat dan tingkah laku fisik yang unik dari *user* serta merupakan sebuah pilihan dasar untuk bukti identifikasi. Biometrik digunakan sebagai bukti yang bagus untuk identifikasi sebab tidak seperti kunci-kunci / tombol- tombol atau *password-password*, biometrik tidak dapat hilang, dicuri, atau terdengar, serta sangat aman dan mudah untuk menentukan identifikasi *user-user*. Karakterisitik-karakteristik fisiologis, seperti cetakan-cetakan jari, merupakan calon untuk bukti (verifikasi) sebab hal tersebut merupakan hal unik pada sebuah bagian populasi yang besar.

Biometrik sangat diperlukan untuk seluruh sistem keamanan, dimana diperkenalkan sebuah kehidupan *user* dan mencakup karakteristik-karakteristik keduanya yaitu fisiologis dan tingkah laku. Karakteristik-karakteristik fisiologis seperti cetakan jari yang relatif stabil secara fisik dimana tidak dapat diubah tanpa penyebab yang berat pada seorang individu. Sifat tingkah laku pada tangan lain, mempunyai beberapa dasar fisiologis tersendiri, tetapi juga menggambarkan pembuatan fisiologis penggunanya.

Teknologi biometrik merupakan definisi yang sama seperti “metode automasi dari pengelompokan atau penggolongan identitas dari sebuah kehidupan manusia yang berdasarkan pada sebuah karakteristik fisiologis atau karakteristik tingkah laku”.

Teknologi-teknologi biometrik menyediakan sebuah level keamanan tambahan. Penyediaan perhitungan terhadap pengukuran untuk masalah bukti identifikasi dapat dikategorikan kedalam tiga grup utama; yaitu (a) sesuatu yang *user* pahami (contoh : *password*), (b) sesuatu yang *user* miliki (contoh : ID card / Kartu ID), atau (c) karakteristik-karakteristik dari *user* itu sendiri.

Jaminan keamanan untuk mengidentifikasi dengan teknik biometrik yang meliputi tiga golongan dari identifikasi (biometrik) merupakan sebuah solusi untuk metode yang lebih aman pada identifikasi.

2.2 Dinamika Keystroke

Dinamika *keystroke* merupakan suatu proses dari penganalisaan dari tipe-tipe *user*, berdasarkan pada pola-pola ritme kebiasaan pengetikannya yang dipantau pada banyaknya inputan dari *keyboard* pada setiap detiknya. Hal tersebut menunjukkan bahwa ritme *keystroke* merupakan sebuah bukti yang bagus dalam pengidentifikasian. Selain itu, tidak seperti sistem-sistem biometrik lainnya yang mungkin sangat mahal untuk diterapkan, dinamika *keystroke* hampir cuma-cuma hanya membutuhkan perangkat *keyboard*.

Teknik-teknik verifikasi *keystroke* dapat digolongkan secara statis atau berkelanjutan (bersambung). Pendekatan verifikasi statis pada analisa pengelompokan karakteristik-karakteristik *keystroke* hanya dilakukan pada saat-saat tertentu, sebagai contoh, pada waktu melakukan serangkaian *login*. Pendekatan statis menyediakan verifikasi *user* yang tepat dibandingkan dengan contoh *password*, tetapi tidak menyediakan keamanan yang terus-menerus, *keystroke* tidak dapat mendeteksi sebuah pergantian / perubahan dari *user* setelah verifikasi inisialnya. Verifikasi yang

berkelanjutan (bersambung), sebaliknya, monitor-monitor berinteraksi dengan *user* dengan mempelajari kebiasaannya mengetik.

Pada awal tahun 1980-an penelitian mempelajari pemakaian pola-pola pada kebiasaan mengetik user untuk suatu identifikasi. Gaines adalah peneliti yang untuk pertama kalinya menyelidiki kemungkinan menggunakan pewaktuan keystroke untuk pengecekan. Percobaan dilakukan dengan sebuah populasi yang sedikit yaitu dengan melihat tingkah laku 7 sekretaris. Sebuah tes statistis yang independen dari profil-profil mereka mempengaruhi penggunaan T-Tes (uji-T) dibawah rata-rata hipotesis dimaksudkan dua kali pewaktuan pada kedua sesi adalah sama, tetapi mempunyai variasi yang berbeda. Pembuatan laporan untuk mengidentifikasi sebuah verifikasi dimana hasil-hasilnya valid pada sebuah sistem verifikasi identifikasi dengan tanda kecepatan kesalahan yaitu 5.5% dan kecepatan proses manipulasi pada kurang lebih 5.0%.

2.3 Neo Fuzzy Neuron

Usaha ini dibuat untuk mengubah model *neuron* konvensional oleh konsep *fuzzy* dalam rangka untuk memperoleh ide adaptasi model *neuron* yang lebih baik untuk mempelajari perilaku sistem yang didefinisikan tidak tepat oleh sifat baik dari kompleksitas derajat yang tinggi. S. C. Lee dan E. T. Lee mengubah aktifitas semua atau tidak (“*all-or-none”*) dari model MacCulloch-Pitts untuk memperoleh *output* sama dengan nol dalam keadaan tidak menembak (*non-firing*) dan beberapa nomer positif μ_i dalam menembak(*firing*), dimana $0 < \mu_i \leq 1$. Dan mereka menghadirkan prosedur sintese umum untuk mencapai otomasi *fuzzy* apapun menggunakan *fuzzy neurons*. Takeshi Yamakawa menghadirkan jenis yang berbeda dari *fuzzy neuron*, dimana hubungan *synaptic* linear diganti dengan karakteristik nonlinear oleh fungsi keanggotaan bernama

seperti *"tightly connected","loosely connected"*, dan sebagainya. Dan hubungan *excitatory* dan hubungan *inhibitory* digambarkan oleh titik pertemuan *fuzzy logic* dan melengkapi *fuzzy logic* diikuti oleh titik pertemuan *fuzzy logic*, berturut-turut (Yamakawa dan Tomoda, 1989). *Fuzzy neuron* ini digunakan untuk membangun sistem perangkat keras dimana telah mencapai pengenalan karakter tulisan tangan dalam 1 *microsecond* (Yamakawa dan Furukawa, 1992). Model algoritma dari *fuzzy neuron* untuk pengenalan pola yang dibuat oleh pembelajaran contoh dasar. Hal ini menggambarkan tipe baru dari model *fuzzy neuron*, masing-masing *nonlinear synapse* dimana digolongkan oleh kumpulan aturan pengertian *fuzzy* dengan bobot yang tunggal dalam akibatnya.

2.3.1 Struktur dan Algoritma Pembelajaran *Neo Fuzzy Neuron*

Struktur dari *neo fuzzy neuron* ditampilkan pada gambar 2.1(a), dimana karakteristik dari tiap *synapse* digambarkan oleh fungsi nonlinear f_i dan *soma* tidak memamerkan fungsi *sigmoidal* sama sekali. Kesatuan dari *signal synaptic* diambil dari aljabar penjumlahan. *Output* dari *neo fuzzy neuron* ini bisa digambarkan dengan persamaan berikut :

$$\hat{Y} = f_1(x_1) + f_2(x_2) + \cdots + f_m(x_m)$$

$$= \sum_{i=1}^{m} f_i(x_i) \qquad (1)$$

Struktur dari nonlinear *synapse* ditunjukkan dengan gambar 2.1(b). Jarak input x_i dipisah menjadi beberapa *fuzzy segments* dimana dikarakteristikkan oleh fungsi keanggotaan μ_{i1}, μ_{i2},…, μ_{ij},…, μ_{in} dalam jangkauan antara x_{min} dan x_{max} seperti

ditunjukkan pada gambar 2.1(c). 1,2,…,j,…,n adalah nomer yang ditempatkan pada nama *fuzzy segments*. Fungsi keanggotaan diikuti oleh variabel bobot $w_{i1},w_{12},\ldots,w_{ij},\ldots,w_{in}$.

Pemetaan dari x_i ke $f_i(x_i)$ ditentukan oleh kesimpulan *fuzzy* dan defuzzifikasi seperti digambarkan dalam gambar 2.2. Kesimpulan *fuzzy* yang dipakai disini adalah sebagai akibat tunggal, karena itu, setiap bobot w_{ij} menentukan nilai seperti 0,3. Hal ini harus menekankan bahwa setiap fungsi keanggotaan dalam kata adalah bentuk segitiga dan diberikan untuk menjadi saling melengkapi (jadi disebut oleh pengarang) dengan satu yang berdekatan. Dengan kata lain, *signal input* x_i mengaktifkan hanya dua fungsi keanggotaan terus menerus dan penjumlahan nilai dari dua fungsi keanggotaan yang berdekatan ini dinamakan k dan k+1 selalu sama dengan 1, karena itu, $\mu_{i,k}(x_i)+\mu_{i,k+1}(x_i)$ sehingga defuzzifikasi mengambil pusat dari gaya berat tidak perlu pembagian dan *output* dari *neo fuzzy neuron* bisa digambarkan dengan persamaan sederhana berikut:

$$f_i(x_i)=\frac{\sum_{j=1}^{n}\mu_{ij}(x_i)-w_{ij}}{\sum_{j=1}^{n}\mu_{ij}(x_{ij})}=\frac{\mu_{ik}(x_i)-w_{ik}+\mu_{i,k+1}x_i-w_{i,k+1}}{\mu_{ik}(x_i)+\mu_{i,k+1}(x_i)}$$

$$=\mu_{ik}(x_i)-w_{ik}+\mu_{i,k+1}(x_i)-w_{i,k+1} \qquad (2)$$

Persamaan ini bisa dicapai dengan arsitektur ditunjukkan dalam gambar 2.1(b).

Bobot w_{ij} ditempatkan dengan belajar, aturan yang digambarkan dengan aturan n *if-then* seperti ditunjukkan sebagai berikut :

[Rule no.j]

Jika x_i salah dalam I*fuzzy segments* μ_{ij}, maka bobot yang sesuai w_{ij} harus menambah secara langsung sebanding dengan kesalahan *output* $(y - \hat{y})$, karena kesalahan disebabkan oleh bobot.

Soal ini bisa diwakili dengan persamaan berikut :

$$w_{ij}(t + 1) = w_{ij}(t) + \alpha_{ij}\gamma_{ij}(x_i)(y - \hat{y}). \quad (3)$$

dimana α_i adalah koefisien pembelajaran untuk nonlinear *synapse* i-th. Ini adalah algoritma pembelajaran dari *neo fuzzy neuron*. Setelah pembelajaran, *neo fuzzy neuron* ini memfasilitasi pemolaan dari data.

2.3.2 Identifikasi dari sistem dinamik nonlinear oleh Neo Fuzzy Neuron

M-input neo fuzzy neuron bisa dikombinasikan dengan elemen rangkaian penundaan untuk mendapatkan *one-input* dan signal proses *one-input* seperti ditunjukkan pada gambar 2.2. Sistem ini bisa mencapai identifikasi dari sistem dinamik walaupun mereka kacau.

Dalam rangka untuk menguji kemungkinan dari sistem identifikasi dari *neo fuzzy neuron* ini dengan elemen penundaan, signal penguji dibangkitkan dimana sangat kacau dan juga sangat sulit untuk diprediksi perilakunya. Signal penguji dibangkitkan dengan sistem yang dinamik ditetapkan sebagai berikut :

$$\begin{aligned} y_{n+1} &= f_1(x_1) + f_2(x_2) + f_3(x_3) \\ &= f_1(y_n) + f_2(y_{n-1}) + f_3(y_{n-2}) \\ &= \frac{5y_n}{1+y_n^2} - 0.5y_n - 0.5y_{n-1} + 0.5y_{n-2} \end{aligned} \quad (4)$$

dengan nilai awal dari $y_0 = 0.733$, $y_1 = 0.234$ dan $y_2 = 0.973$. Contoh ditunjukkan pada gambar 2.3, dimana sangat kacau dan sangat sulit di prediksi perilaku waktu sequensialnya.

Sistem dinamik diidentifikasikan dengan kecenderungan *neo fuzzy neuron* dengan elemen penundaan. Walaupun sistem sistem dinamik tiga dimensi seperti ditunjukkan pada persamaan diatas, *1-synapse, 2-synapse, 3-synapse, 4-synapse, dan 5-synapse neo fuzzy neuron* diperiksa untuk mempelajari seberapa tepat mereka bisa mengidentifikasi perhatian dari sistem.

Nomer dari *fuzzy segment* di setiap ruang input x_i adalah 12 dan fungsi keanggotaan dua belas itu ditempatkan disetiap *synapse*. Setiap bobot yang cocok dengan fungsi keanggotaan berubah sesuai dengan persamaan berikut :

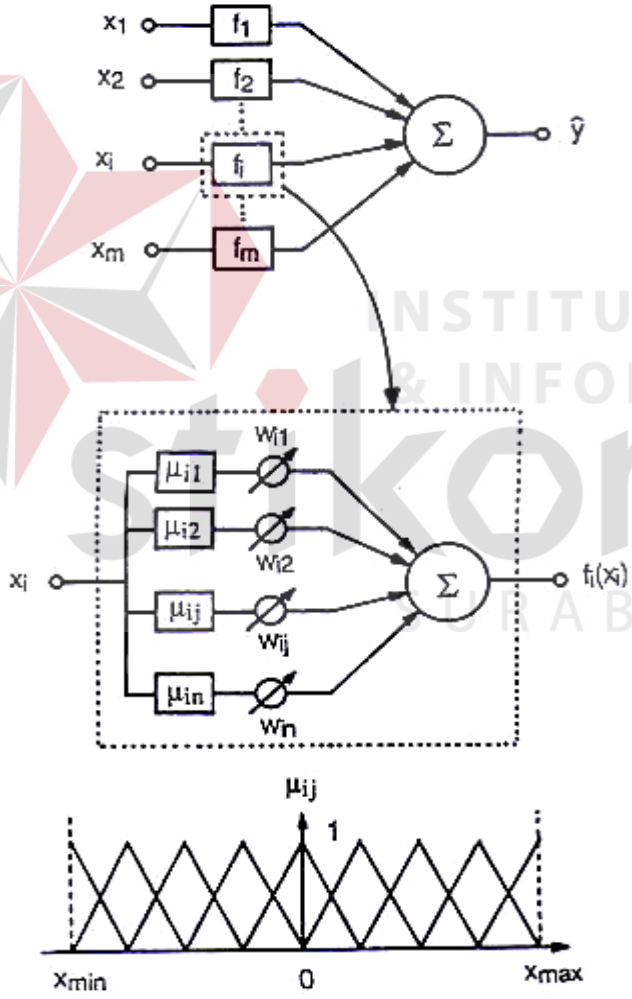
$$w_{ij}(t+1) = w_{ij}(t) + \alpha_i \mu_{ij}(x_i)(y_{n+1} - \overline{y_{n+1}} \quad (5)$$

Pembelajaran dicapai sebagai berikut. Semua nilai awal dari bobot adalah nol. Ke 50 rangkaian y_0, y_1, y_2 y_3,…, y_{49} (gambar 2.3) dipakai secara berurutan pada terminal input y_{n+1} (gambar 2.2) dan data diturunkan satu persatu melalui rangkaian penundaan. Dalam rangkaian ini, semua bobot diubah oleh persamaan diatas, dimana koefisien pembelajaran α_i adalah 0.2. Setelah lingkaran pembelajaran selesai oleh 45 set dari lima data dipakai dari elemen penundaan ke input *synapse* x_1, x_2, x_3, x_4, x_5, pada keadaan *5-synapse neo fuzzy neuron*. Setelah 100 kali pembelajaran, nilai akar rata-rata kuadrat (*root-mean-square*) dari kesalahan bisa diukur untuk setiap nomor dari *synapse*.yang dapat dilihat pada tabel 2.1.

Tabel 2.1 Tabel akar rata-rata pangkat

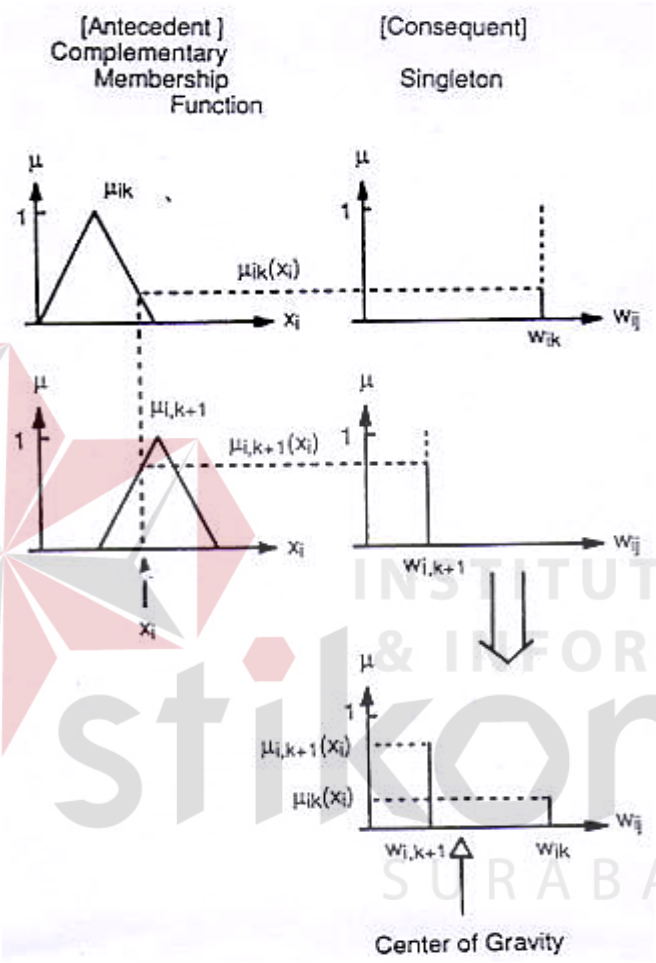
number of synapsises (dimensions)	1	2	3	4	5
$\sqrt{\text{error}^2}$	8.59×10^{-1}	5.24×10^{-1}	3.86×10^{-2}	3.23×10^{-2}	1.87×10^{-2}

.Pada gambar 2.1 menunjukkan struktur dari *neo fuzzy neuron*. Gambar (a) menunjukkan struktur dari *neo fuzzy neuron*. Setiap karakteristik *synaptic* ditunjukkan oleh fungsi nonlinear f_i. Gambar (b) menunjukkan struktur sari nonlinear



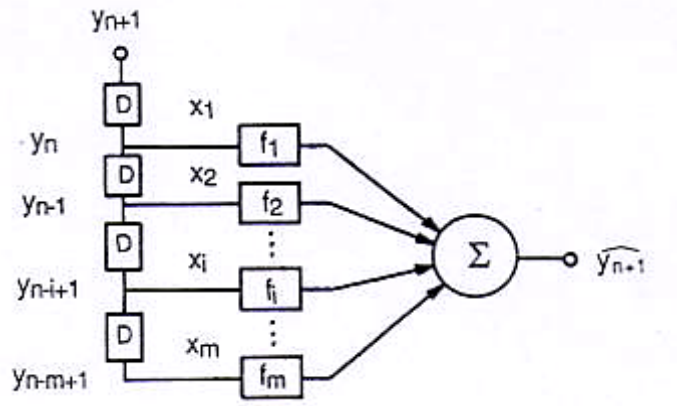
Gambar 2.1 Struktur *neo fuzzy neuron*

synapse dimana didefinisikan dengan kumpulan *if-then ruleincluding* yang tunggal sebagai akibat. Bentuk segitiga dan fungsi keanggotaan komplemen yang ditunjukkan pada gambar (c) ditentukan untuk *fuzzy segments* dalam ruang input.



Gambar 2.2 *Synapse* nonlinear

Gambar 2.2 menunjukkan *synapse* nonlinear yang dikarakteristikkan dengan kesimpulan *fuzzy* yang sangat sederhana. Aktifitas signal input hanya ada dua aturan, karena *antecedents* dari semua aturan telah dijelaskan dengan fungsi keanggotaan komplemen. Lebih lanjut, semua akibat dari semua aturan telah dijelaskan dengan tunggal, jadi defuzzyfikasi adalah mudah sukses tanpa pembagian aritmetika.



Gambar 2.3 Kombinasi dari *neo fuzzy neuron*

Gambar 2.3 menunjukkan kombinasi dari *neo fuzzy neuron* dengan rangkaian elemen penundaan D dimana memfasilitasi identifikasi sistem dan prediksi perilakunya.

2.4 Penggunaan DFD (*Data Flow Diagram*)

Meskipun suatu analisa yang disebut dengan DFD mempunyai struktur tersendiri, namun sistem analisa dapat meletakkan secara bersamaan sebuah gambar yang merepresentasikan seluruh proses-proses data dalam sebuah organisasi. Pendekatan data *flow* menitik beratkan pada logika yang tersirat dari suatu sistem.

Dengan menggunakan kombinasi simbol, sistem analisa dapat membuat sebuah gambaran dari suatu proses yang sebenarnya dengan menggunakan dokumen sistem.

2.4.1 Keuntungan pembuatan data flow

Data *flow* mempunyai lima keuntungan utama dari penjelasan-penjelasan jalannya data dalam sistem, yaitu :

1. Kebebasan yang berasal dari kepercayaan untuk mengimplementasikan secara benar teknik sistem dari suatu sistem yang baru.
2. Memberikan pengertian dari hubungan sistem-sistem dan subsistem yang ada.

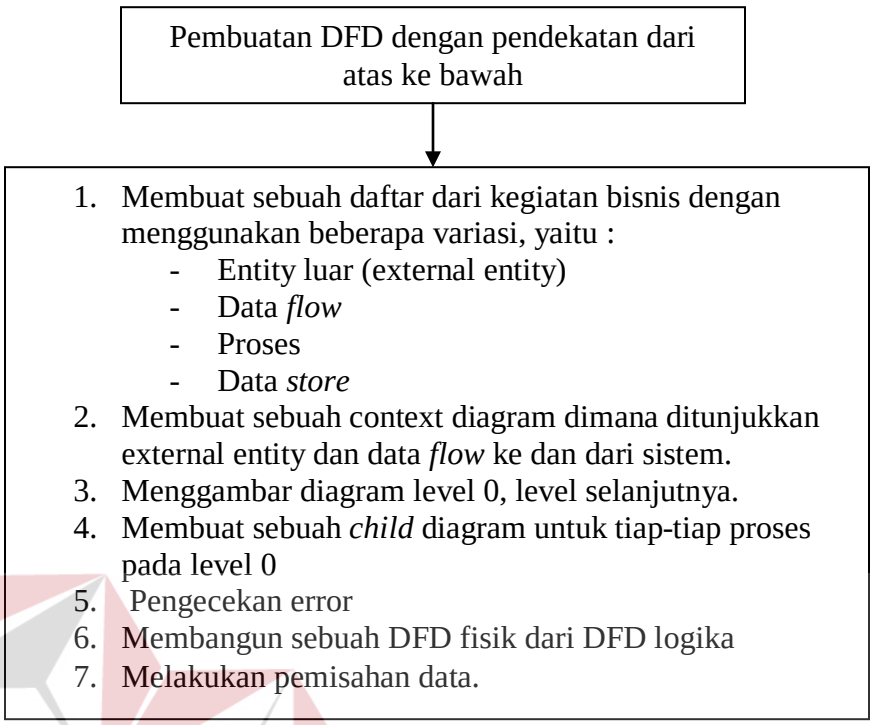
3. Komunikasi mengenai pengetahuan sistem bagi *user* melalui DFD
4. Analisa dari sebuah usulan sistem untuk menentukan jika data dan proses-proses yang ada dapat didefinisikan secara mudah.
5. Penggunaan data *flow* merupakan keuntungan tambahan yang dapat digunakan sebagai latihan bagi sistem analis, kesempatan sistem analis menjadi lebih baik untuk mengerti tentang hubungan sistem dan subsistem yang ada didalamnya.

Keuntungan dari kelima penggunaan data *flow* tersebut dapat digunakan sebagai *tools* yang interaktif dengan *user*. Hal yang menarik dalam penggunaan DFD adalah ditunjukannya kepada *user* gambaran-gambaran secara lengkap dari sistem. *User* dapat menanyakan guna memberikan komentar pada konsep, sistem analis dapat merubah sistem berdasarkan keinginan *user*. Keuntungan terakhir dari penggunaan DFD adalah dapat mengikuti sistem analis untuk mendeskripsikan komponen-komponen yang digunakan dalam suatu diagram. Analisa dapat ditampilkan untuk menjamin bahwa semua *output* mempunyai isi atau memperoleh data inputan dari prosesnya.

2.4.2 Pembuatan DFD (Data Flow Diagram)

DFD dapat dan harus digambarkan secara sistematis . Pertama, dibutuhkan sistem analis untuk mengkonsep data *flow*, dari atas ke bawah seperti ditunjukkan pada gambar 2.7.

Untuk memulai sebuah DFD dari suatu sistem biasanya dituangkan dalam sebuah daftar dengan empat kategori yaitu entity luar, arus data, proses, dan penyimpanan data. Daftar ini akan membantu menentukan batasan-batasan dari suatu sistem yang akan digambarkan.



Gambar 2.4 Pembuatan DFD (*Data Flow Diagram*)

Pada dasarnya daftar itu berisi elemen-elemen data yang dikarang. Elemen-elemen tersebut terdiri dari :

a. Pembuatan context diagram

Context diagram adalah level yang tertinggi dalam sebuah DFD dan hanya berisi satu proses serta merupakan representasi dari sebuah sistem. Proses dimulai dengan penomeran ke-0 dan untuk seluruh entity luar akan ditunjukkan dalam context diagram yang sama seperti data awal yang dikirim dari entity luar. Context diagram tidak berisi penyimpanan data.

b. Pembuatan diagram level 0 serta level berikutnya

Diagram level 0 dihasilkan oleh context diagram dan berisi proses-proses. Pengisian proses-proses yang berlebihan pada level ini akan menghasilkan sebuah

diagram yang salah, sehingga sulit umtuk dimengerti . Masing-masing proses diberikan penomoran dengan sebuah bentuk integer. Umumnya dimulai dari kiri atas dan penyelesaiannya di kanan bawah dalam sebuah bentuk diagram.

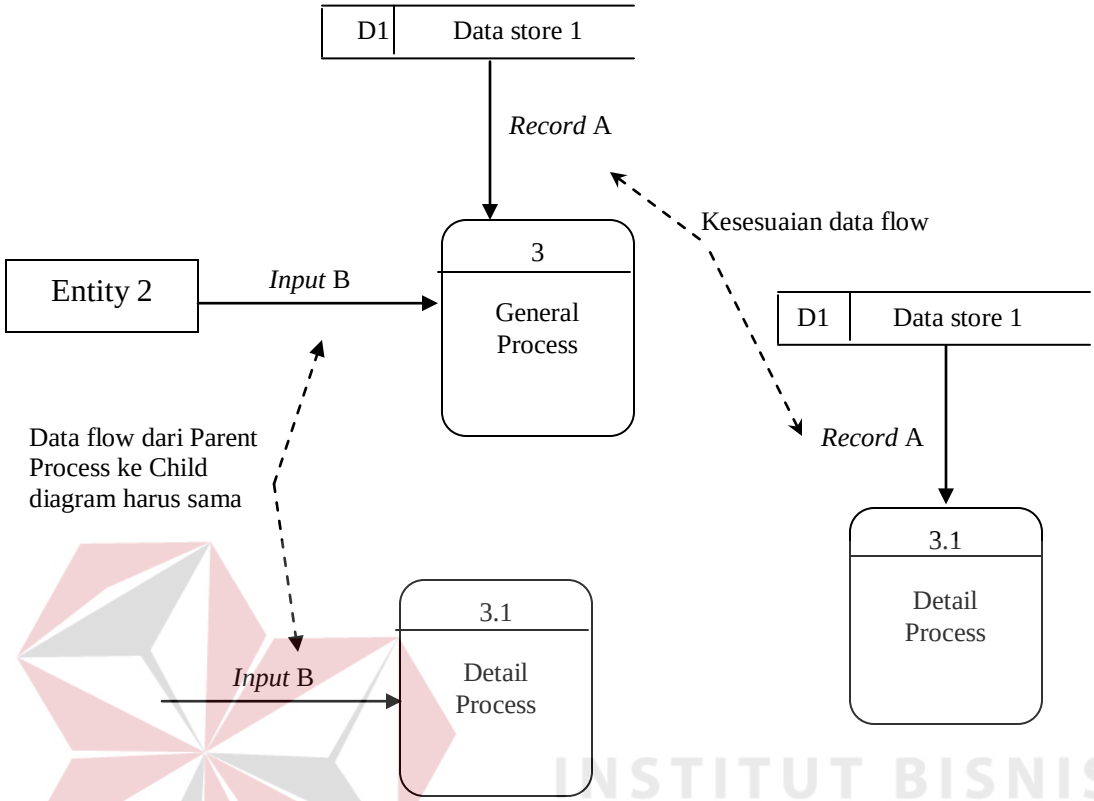
c. Pembuatan *child* diagram

Child diagram diberikan nomor yang sama seperti proses diatasnya (parent proses) dalam diagram level 0. Contohnya, proses 3 harus diturunkan ke diagram 3, proses pada *child* diagram menggunakan penomoran unik untuk masing-masing proses dengan mengikuti penomoran proses diatasnya . Contohnya, dalam diagram 3 proses-proses diberikan nomor 3.1, 3.2, 3.3 dan seterusnya. Konversi ini diikuti oleh analis sistem untuk menelusuri seri-seri dari proses-proses yang dikeluarkan oleh beberapa level, jika pada proses diagram level 0 digambarkan sebagai 1, 2, , dan 3 maka *child* diagram-diagramnya adalah 1, 2, dan 3 pada level yang sama. Ilustri level detil dengan sebuah *child* DFD dapat ditunjukkan pada gambar 2.8.

d. Pengecekan kesalahan

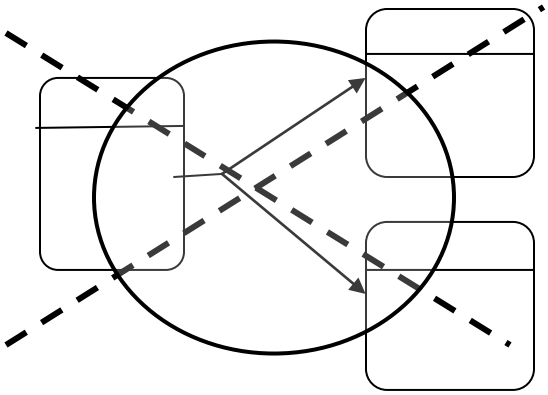
Pengecekan kesalahan-kesalahan pada diagram digunakan untuk melihat kesalahan-kesalahan yang terdapat pada sebuah DFD. Beberapa kesalahan-kesalahan yang umum terjadi ketika penggambaran / pembuatan DFD, ditunjukkan pada gambar 2.6.Beberapa kesalahan yang terjadi diantaraya :

1. Lupa untuk meng*input*kan sebuah arus data atau arah panah langsung. Sebagai contoh adalah penggambaran proses yang menunjukkan sebuah data *flow* seperti *input* atau seperti *output*. Tiap-tiap proses pengubahan data harus menerima *input* dan *output*. Tipe kesalahan ini terjadi ketika sistem analis lupa memasukkan sebuah data *flow* atau meletakkan sebuah arah panah ditempat yang salah.

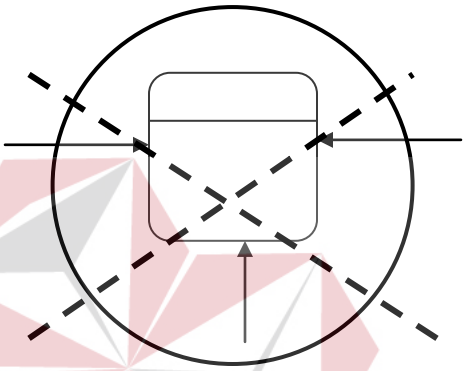


Gambar 2.5 Pembuatan *Child Diagram*

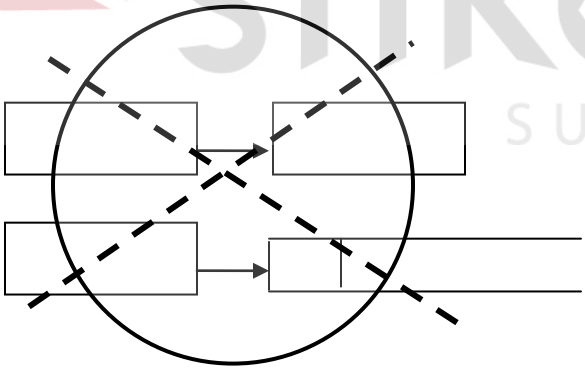
2. Hubungan penyimpanan data dan entity luar secara langsung satu sama lain. Data *store* dan entity tidak mungkin dikoneksikan satu sama lain ; data *store* dan entity luar harus dikoneksikan melalui sebuah proses.
3. Kesalahan penamaan (label) pada proses-proses atau data *flow*. Pengcekan DFD untuk memastikan bahwa tiap-tiap objek atau data *flow* telah diberikan label. Sebuah proses haruslah di indikasikan seperti nama dari sistem atau menggunakan format kata kerja-kata benda. Tiap data *flow* haruslah dideskripsikan dengan sebuah kata benda.



Sebuah data flow tidak diperbolehkan mempunyai percabangan / memisahkan diri (flow) ke dalam dua atau lebih data flow yang berbeda



Sebuah proses harus mempunyai minimal satu *input*an data flow dan satu *output* data flow



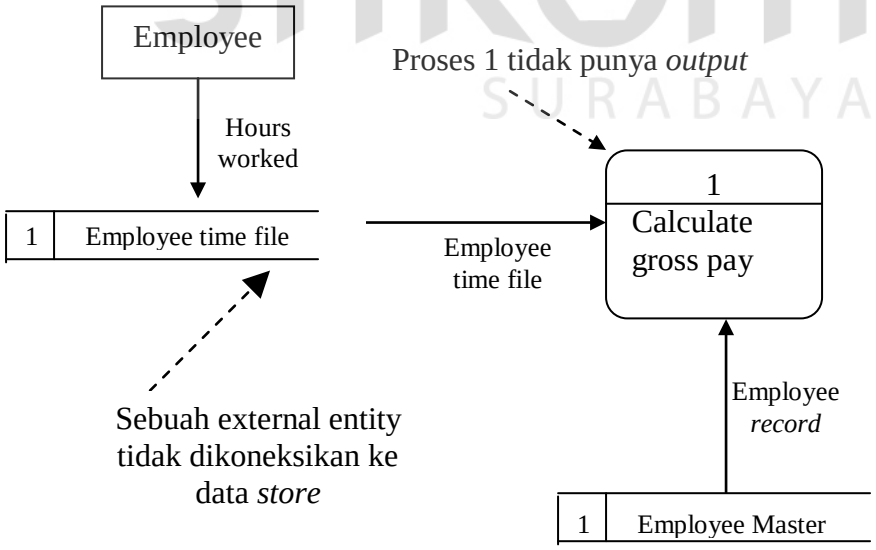
Semua data flow salah satunya harus berasal atau berakhir pada sebuah proses

Gambar 2.6 Pengecekan kesalahan pada DFD (Data Flow Diagram)

4. Memasukkan lebih dari sembilan proses dalam sebuah DFD. Memiliki banyak proses akan mengakibatkan kekacauan pada diagram sehingga dapat menyebabkan

kebingungan dalam pembacaan sebuah proses dan akan menghalangi tingkat komunikasi. Jika lebih dari sembilan proses dalam sebuah sistem, maka beberapa grup dalam proses dilakukan bersama-sama ke dalam sebuah sub sistem dan meletakkannya dalam sebuah *child* diagram.

5. Menghilangkan suatu arus data. Pengujian dari suatu diagram yang menunjukkan garis / arah (*flow*), dimana untuk setiap proses data *flow* hanya mempunyai *input* data, *output* kecuali dalam kasus dari detil (*child*). Setiap *child* data dari DFD, arah arus data seringkali digambarkan untuk mengidentifikasikan bahwa diagram tersebut kehilangan data *flow*. Seperti di tunjukkan pada gambar 2.7
6. Membuat ketidaksesuiaan komposisi dalam *child* diagram , dimana tiap *child* diagram harus mempunyai *input* dan *output* arus data yang sama seperti proses dilevel atasnya (parent proses). Pengecualian untuk rule ini adalah kurangnya *output*, seperti kesalahan garis yang ada didalam *child* diagram.



Gambar 2.7 Arus data

2.4.3 Perbedaan DFD (Logika dan Fisik)

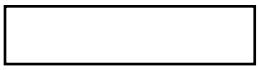
Tabel 2.2 Tabel perbedaan DFD (Logika dan Fisik)

Disain	Logika	Fisik
Gambaran model	Operasi-operasi bisnis	Bagaimana sistem akan diimplementasikan (atau bagaimana sistem dijalankan)
Apa yang ditampilkan oleh proses	Aktivitas bisnis	Program-program, modul program, dan prosedur-prosedur manual
Apa yang ditampilkan oleh data *store*	Koleksi-koleksi dari data yang dikesampingkan dari bagaimana data tersebut di simpan	File-file fisik dan database-database dari file-file manual
Kontrol sistem	Menunjukkan kontrol-kontrol bisnis	Menunjukkan kontrol-kontrol untuk validasi *input* data, untuk memperoleh sebuah *record* , untuk memastikan kesuksesan proses dan untuk keamanan sistem

2.5 Entity Relationship Diagram

Struktur logika secara keseluruhan dari sebuah basis data (database) dapat dinyatakan secara grafis melalui sebuah ER Diagram yang terdiri atas komponen-komponen sebagai berikut :

1. Persegi panjang yang melambangkan himpunan entiti.



Gambar 2.8. Himpunan Entitas

2. Elips, yang melambangkan atribut

Gambar 2.9. Atribut

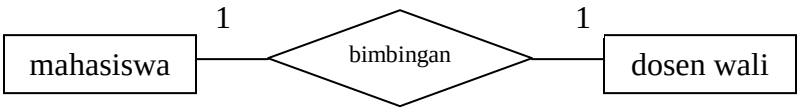
3. Belah Ketupat, yang menghubungkan atribut pada himpunan entiti dan himpunan entiti pada himpunan hubungan.

Gambar 2.10. Himpunan Relasi

4. Garis Lurus, yang menghubungkan atribut-atribut pada himpunan entiti dan himpunan entiti pada himpunan hubungan.

Gambar 2.11. E-R Link

5. Kardinalitas Relasi dapat dinyatakan dengan banyaknya garis cabang atau dengan pemakaian angka.



Gambar 2.12. Relasi One-to-One

Relasi one-to-one berarti dalam satu atribut hanya memiliki satu relasi dengan atribut yang lainnya.

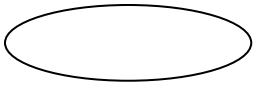
Gambar 2.13. Relasi One-to-Many

Relasi one-to-many berarti satu atribut memiliki lebih dari satu relasi dengan atribut yang lain.

Gambar 2.14. Relasi Many-to-Many

Relasi many-to-many berarti suatu atribut memiliki banyak relasi dengan atribut lainnya.

Tabel 2.3. Simbol pada ERD

Nama	Symbol	Keterangan
Entity		Entity digambarkan dengan lambang segi empat yang menggambarkan kejadian yang dicatat dalam database
Atribut		Atribut digambarkan dengan lambang elips dimana fungsinya adalah untuk menjelaskan entitas.
Hubungan		Lambang ini menggambarkan hubungan yang terjadi dari setiap entity yang dilambangkan dengan gambar belah ketupat.
Penghubung		Merupakan garis penghubung yang menghubungkan antara entity termasuk menjelaskan jenis hubungan apa yang terjadi antara kedua entity tersebut.