BAB II
LANDASAN TEORI

2.1 Konsep Dasar Sistem Pendukung Keputusan (SPK)

2.1.1 Pengertian Sistem Pendukung Keputusan (SPK)

Pengertian sistem pendukung keputusan yang dikemukakan oleh Michael S Scott Morton dan Peter G W Keen, dalam buku Sistem Informasi Manajemen (McLeod, 1998) menyatakan bahwa sistem pendukung keputusan merupakan sistem penghasil informasi yang ditujukan pada suatu masalah yang harus dibuat oleh manajer.

Menurut Raymond McLeod, Jr (1998) mendefinisikan sistem pendukung keputusan merupakan suatu sistem informasi yang ditujukan untuk membantu manajemen dalam memecahkan masalah yang dihadapinya. Definisi selengkapnya adalah sistem penghasil informasi spesifik yang ditujukan untuk memecahkan suatu masalah tertentu yang harus dipecahkan oleh manajer pada berbagai tingkatan.

Definisi menurut Litlle mengemukakan bahwa sistem pendukung keputusan adalah suatu sistem informasi berbasis komputer yang menghasilkan berbagai alternatif keputusan untuk membantu manajemen dalam menangani berbagai permasalahan yang terstruktur ataupun tidak terstruktur dengan menggunakan data atau model.

2.1.2 Komponen Sistem Pendukung Keputusan (SPK)

Secara garis besar *Decision Support System* (DSS) atau Sistem Pendukung Pengambilan Keputusan dibangun oleh tiga komponen besar (Sri, 2006), yaitu :

a. Database

Sistem *database* berisi kumpulan dari semua data bisnis yang dimiliki perusahaan atau lembaga, baik yang berasal dari transaksi sehari-hari,

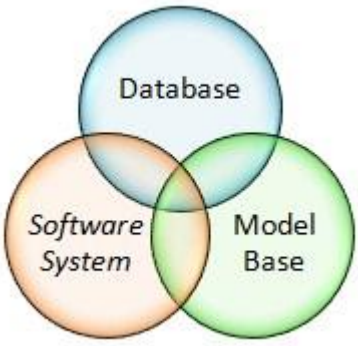
maupun data dasar (*master file*). Untuk keperluan DSS, diperlukan data yang relevan dengan permasalahan yang hendak dipecahkan melalui simulasi.

b. Model *Base*

Model *Base* atau suatu model yang merepresentasikan permasalahan ke dalam format kuantitatif (model matematika sebagai contohnya) sebagai dasar simulasi atau pengambilan keputusan, termasuk di dalamnya tujuan dari permasalahan (obyektif), komponen-komponen terkait, batasan-batasan yang ada (*constraints*), dan hal-hal terkait lainnya.

c. Software System

Kedua komponen tersebut untuk selanjutnya disatukan dalam komponen ketiga yaitu *software system*, setelah sebelumnya direpresentasikan dalam bentuk model yang "dimengerti" komputer. Contohnya adalah penggunaan teknik RDBMS (*Relational Database Management System*), OODBMS (*Object Oriented Database Management System*) untuk memodelkan struktur data. Sedangkan MBMS (*Model Base Management System*) dipergunakan untuk mere-presentasikan masalah yang ingin dicari pemecahannya. Entiti lain yang terdapat pada produk DSS baru adalah DGMS (*Dialog Generation and Management System*), yang merupakan suatu sistem untuk memungkinkan terjadinya "dialog" interaktif antara komputer dan manusia sebagai pengambil keputusan.



Gambar 2.2 Komponen Sistem Pendukung Keputusan (SPK)

2.1.3 Tahap Pengambilan Keputusan

Alur atau proses pemilihan alternatif tindakan/keputusan terdiri dari langkah-langkah berikut (Sri, 2006) :

a. Tahap Penelusuran (*Intelligence Phase*)

   Suatu tahap proses seseorang dalam rangka pengambil keputusan untuk permasalahan yang dihadapi, terdiri dari aktivitas penelusuran, pendeteksian serta proses pengenalan masalah. Data masukan diperoleh, diuji dalam rangka mengidentifikasi masalah.

b. Tahap Peracangan (*Design Phase*)

   Tahap proses pengambil keputusan setelah tahap *intellegence* meliputi proses untuk mengerti masalah, menurunkan solusi dan menguji kelayakan solusi. Aktivitas yang biasanya dilakukan seperti menemukan, mengembangkan dan menganalisa alternatif tindakan yang dapat dilakukan.

c. Tahap Pilihan (*Choice Phase*)

   Pada tahap ini dilakukan proses pemilihan diantara berbagai alternatif tindakan yang mungkin dijalankan. Hasil pemilihan tersebut kemudian diimplementasikan dalam proses pengambilan keputusan.

d. Tahap Implementasi (*Implementation Phase*)

   Pada tahap ini merupakan tahap pelaksanaan dari keputusan yang telah diambil. Pada tahap ini perlu disusun serangkaian tindakan yang terencana, sehingga hasil keputusan dapat dipantau dan disesuaikan apabila diperlukan perbaikan-perbaikan.

2.2 Metode *Simple Additive Weighting* (SAW)

Metode *Simple Additive Weighting* (SAW) sering juga dikenal istilah metode penjumlahan terbobot. Konsep dasar metode SAW adalah mencari penjumlahan terbobot dari rating kinerja pada setiap alternatif pada semua atribut. Metode SAW membutuhkan proses normalisasi matriks keputusan (X) ke suatu skala yang dapat diperbandingkan dengan semua rating alternatif yang ada. Metode ini merupakan metode yang paling terkenal dan paling banyak digunakan dalam

menghadapi situasi *Multiple Attribute Decision Making* (MADM). MADM itu sendiri merupakan suatu metode yang digunakan untuk mencari alternatif optimal dari sejumlah alternatif dengan kriteria tertentu (Sri, 2006).

Metode SAW ini mengharuskan pembuat keputusan menentukan bobot bagi setiap atribut. Skor total untuk alternatif diperoleh dengan menjumlahkan seluruh hasil perkalian antara rating (yang dapat dibandingkan lintas atribut) dan bobot tiap atribut. Rating tiap atribut haruslah bebas dimensi dalam arti telah melewati proses normalisasi matriks sebelumnya. Adapun langkah-langkah penyelesaian SAW adalah sebagai berikut :

a. Menentukan kriteria-kriteria yang akan dijadikan acuan dalam pengambilan keputusan, yaitu Ci.
b. Menentukan rating kecocokan setiap alternatif pada setiap kriteria.
c. Membuat matriks keputusan berdasarkan kriteria (Ci), kemudian melakukan normalisasi matriks berdasarkan persamaan yang disesuaikan dengan jenis atribut (atribut keuntungan ataupun atribut biaya) sehingga diperoleh matriks ternormalisasi R.
d. Hasil akhir diperoleh dari proses perankingan yaitu penjumlahan dari perkalian matriks ternormalisasi R dengan vektor bobot sehingga diperoleh nilai terbesar yang dipilih sebagai alternatif terbaik (Ai) sebagai solusi.

Formula untuk melakukan normalisasi tersebut adalah :

$$r_{ij} = \begin{cases} \dfrac{x_{ij}}{max_i x_{ij}} & jika\ j\ adalah\ atribut\ keuntungan\ (benefit) \\ \dfrac{min_i x_{ij}}{x_{ij}} & jika\ j\ adalah\ atribut\ biaya\ (cost) \end{cases}$$

Dimana :

r_{ij} = rating kinerja ternormalisasi

max_i = nilai maximum dari setiap baris dan kolom

min_i = nilai minimum dari setiap baris dan kolom

x_{ij} = baris dan kolom dari matriks

Dengan r_{ij} adalah rating kinerja ternormalisasi dari alternatif A_i pada atribut C_j ; i =1,2,…m dan j = 1,2,…,n. Nilai preferensi untuk setiap alternatif (Vi) diberikan sebagai :

$$v_i = \sum_{j=1}^{n} w_j r_{ij}$$

Dimana :

v_i = nilai akhir dari alternatif (Nilai v_i yang lebih besar mengindikasikan bahwa alternatif Ai lebih terpilih)

w_j = bobot yang telah ditentukan

r_{ij} = normalisasi matriks

2.3 Metode Pengembangan Sistem

Pada awal pengembangan perangkat lunak, para pembuat program (*programmer*) langsung melakukan pengodean perangkat lunak tanpa menggunakan prosedur atau tahapan pengembangan perangkat lunak. Dan ditemuilah kendala-kendala seiring dengan pengembangan skala sistem-sistem perangkat yang semakin besar (Rosa, 2011).

SDLC dimulai dari tahun 1960-an, untuk mengembangkan sistem skala usaha besar secara fungsional untuk para konglomerat pada zaman itu. Sistem-sistem yang di bangun mengelola informasi kegiatan dan rutinitas dari perusahaan-perusahaan yang berpotensi memiliki data yang besar dalam perkembangannya.

SDLC atau *Software Development Life Cycle* atau sering disebut juga *System Development Life Cycle* adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model-model dan metodologi yang

digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya (berdasarkan *best practice* atau cara-cara yang sudah teruji baik).

Tahapan-tahapan yang ada pada SDLC secara global adalah sebagai berikut :

a. Inisiasi (*Initiation*)

   Tahap ini biasanya ditandai dengan pembuatan proposal proyek perangkat lunak.

b. Pengembangan Konsep Sistem (*System Concept Development*)

   Mendefinisikan linkup konsep termasuk dokumen lingkup sistem, analisis manfaat biaya, manajemen rencana, dan pembelajaran kemudaha sistem.

c. Perencanaan (*Planning*)

   Mengembangkan rencana manajemen proyek dan dokumen perencaan lainnya.

d. Analisis Kebutuhan (*Requirements Analysis*)

   Menganalisis kebutuhan pemakai sistem perangkat lunak (*user*) dan mengembangkan kebutuhan user dan membuat dokumen kebutuhan fungsional.

e. Desain (*Design*)

   Mentransformasikan kebutuhan detail menjadi kebutuhan yang sudah lengkap, dokumen desain sistem fokus pada bagaimana dapat memenuhi fumgsi-fungsi yang dibutuhkan.

f. Pengembangan (*Development*)

   Mengonversi desain ke sisitem informasi yang lengkap termasuk bagaimana memperoleh dan melakukan instalasi lingkungan sistem yang dibutuhkan, mempersiapkan berkas atau file pengujian, pengodean, pengompilasian, memperbaiki dan membersihkan program.

g. Integrasi dan Pengujian (*Integration and Test*)

   Mendemonstrasikan sistem perangkat lunak bahwa telah memenuhi kebutuhan yang dispesifikasikan pada dokumen kebutuhan fungsional dengan diarahkan oleh staf penjamin kualitas dan user sehingga menghasilkan laporan analisis pengujian.

h. Implementasi (*Implementation*)

   Implementasi perangkat lunak pada lingkungan produksi (lingkungan pada user) dan menjalankan resolusi dari permasalahan yang teridentifikasi dari fase integrasi dan pengujian.

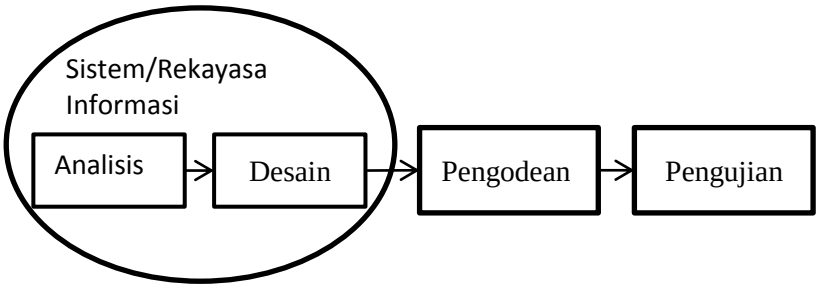
i. Operasi dan Pemeliharaan (*Operations and Maintenance*)

   Mendeskripsikan pekerjaan untuk mengoperasikan dan memelihara sistem informasi pada lingkungan produksi, termasuk implementasi akhir dan masuk pada proses peninjauan.

j. Disposisi (*Disposition*)

   Mendeskripsikan aktifitas akhir dari pengembangan sistem dan membangun data yang sebenarnya sesuai dengan aktifitas *user*.

SDLC memiliki beberapa model dalam penerapan tahapan prosesnya, diantaranya adalah model *waterfall.* Model air terjun (*waterfall*) sering juga disebut model sekuensial linier (*sequential linear*) atau alur hidup klasik (*classic life cycle*). Model air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis, desain, pengodean, pengujian dan tahap pendukung (Rosa, 2011) seperti pada Gambar 2.3.



Gambar 2.3 Ilustrasi Model *Waterfall*

a. Analisis Kebutuhan Perangkat Lunak

   Proses pengumpulan kebutuhan dilakukan secara intensif untuk mespesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat

lunak seperti apa yang dibutuhkan *user*. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk didokumentasikan.

b. Desain

   Desain perangkat lunak adalah proses multilangkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka, dan prosedur pengodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desainperangkat lunak yang dihasilkan pada tahap ini juga perlu didokumentasikan.

c. Pembuatan Kode Program

   Desain harus ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain.

d. Pengujian

   Pengujian fokus pada perangkat lunak secara dari segi logik dan fungsional dan memastikan bahwa semua bagian telah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (*error*) dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan.

e. Pendukung

   Tidak menutup kemungkinan sebuah perangkat lunak mengalami perunahan ketika sudah dikirim ke user. Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru. Tahap pendukung atau pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perbuahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru.

Dari kenyataan yang terjadi sangat jarang model air terjun dapat dilakukan sesuai alurnya karena sebab berikut :

a. Perubahan spesifikasi perangkat lunak terjadi ditengah alur pengembangan.

b. Sangat sulit bagi pelanggan untuk mendefinisikan semua spesifikasi di awal alur. Pelanggan seringkali butuh contoh (*prototype*) untuk menjabarkan spesifikasi kebutuhan sistem lebih lanjut.
c. Pelanggan tidak mungkin bersabar mengakomodasi perubahan yang diperlukan di akhir alur pengembangan.

Dengan berbagai kelemahan yang dimiliki model air terjun tapi model ini telah menjadi dasar dari model-model yang lain dalam melakukan perbaikan model pengembangan perangkat lunak. Model air terjun sangat cocok digunakan kebutuhan pelanggan sudah sangat dipahami dan kemungkinan terjadinya perubahan kebutuhan selama pengembangan pernagkat lunak kecil. Hal positif dari model air terjun adalah struktur pengembangan sistem jelas, dokumentasi dihasilkan disetiap tahap dan sebuah tahap dijalankan setelah tahap sebelumnya selesai dijalankan (tidak ada tumpang tindih pelaksanaan tahap).

2.4 Alat Bantu Pengembangan Sistem

2.4.1 Flowchart

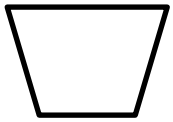
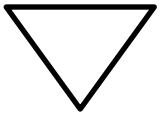
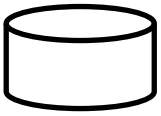
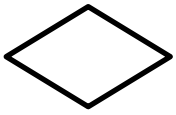
Flowchart adalah bagan alir yang menggambarkan suatu tahap penyelesaian masalah dengan menggunakan simbol-simbol yang standar efektif dan tepat. Ada dua macam *flowchart*, yaitu :

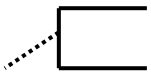
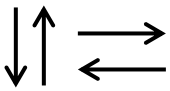
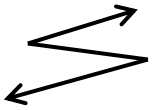
a. *Flowchart* Dokumen

   Flowchart dokumen merupakan suatu bagan yang menunjukkan arus dari laporan dan formulir termasuk tembusan-tembusannya.

Tabel 2.1 Simbol *Flowchart* Dokumen (Susanta, 2004)

No.	Simbol	Keterangan
1.		Simbol dokumen menunjukkan dokumen input dan output baik proses manual, mekanik atau manual.

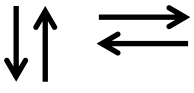
2.		Simbol manual menunjukkan kegiatan yang dilakukan secara manual.
3.		*File* bukan komputer yang diarsipkan
4.		Simbol proses menunjukkan kegiatan proses dari program komputer.
5.		Simbol *harddisk* menunjukkan input atau output menggunakan *harddisk*.
6.		Simbol *keyboard* menunjukkan *input* menggunakan *online keybord*.
7.		Simbol *diskette* menunjukkan *input* atau *output* menggunakan *disc*.
8.		Simbol penghubung menunjukkan penghubung ke halaman yang sama atau ke lain halaman.
9.		Simbol *display* menunjukkan *output* yang ditampilkan di monitor.
10.		Simbol terminal yang menunjukkan awal proses dan akhir proses.
11.		Simbol kondisi menunjukkan keadaan/kondisi.
12.		Simbol pita magnetik menunjukkan *input* atau *output* menggunakan pita magnetik.

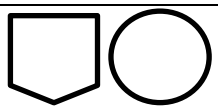
13.		Simbol pita kertas berlubang menunjukkan *input* atau *output* menggunakan kertas berlubang.
14.		Simbol drum magnetik menunjukkan *input* atau *output* menggunakan drum magnetik.
15.		Simbol penjelasan menunjukkan penjelasan dari suatu proses.
16.		Simbol garis alir menunjukkan arus dari proses.
17.		Simbol penghubung komunikasi menunjukkan proses transmisi data melalui *channel* komunikasi.

b.Flowchart Program

Flowchart program yang menggambarkan secara detail atau secara rinci tentang proses atau urutan logika yang terjadi dalam sebuah program.

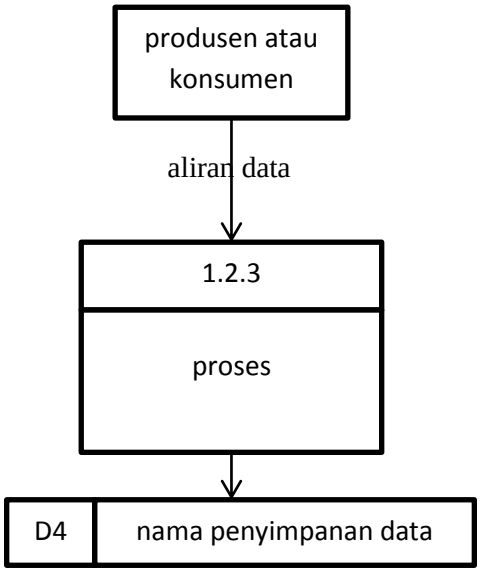
Gambar 2.2 Simbol *Flowchart* Program (Susanta, 2004)

No.	Simbol	Keterangan
1.		Simbol titik terminal digunakan untuk menunjukkan awal dan akhir dari suatu proses.
2.		Simbol *input/output* digunakan untuk mewakili data *input* dan *output.*
3.		Simbol keputusan digunakan untuk mewakili suatu proses pengolahan data.
4.		Simbol arah data digunakan sebagai arah arus logika program.

5.		Simbol proses digunakan untuk penyelesaian kondisi didalam program.
6.		Simbol proses terdefinisi digunakan untuk menunjukkan suatu operasi yang rinciannya ditunjukkan ditempat lain.
7.		Simbol persiapan pemberian nilai awal dari suatu variabel.
8.		Simbol penghubung menunjukkan penghubung ke halaman yang sama atau ke halaman berbeda.

2.4.2 *Data Flow Diagram* (DFD)

Data Flow Diagram (DFD) awalnya dikembangkan oleh Chris Gane dan Trish Sarson pada tahun 1979 yang termasuk dalam *Structure System Analysis and Design Methodology* (SSADM) yang ditulis oleh Chris Gane dan Trish Sarson. Sistem yang dikembangkan ini berbasis pada dekomposisi fungsional dari sebuah sistem. Berikut adalah contoh DFD yang dikembangkan oleh Chris Gane dan Trish Sarson pada Gambar 2.4.



Gambar 2.4 DFD dikembangkan oleh Crish Gane & Trish Sarson

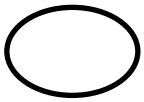
Edward Yourdon dan Tom DeMarco memperkenalkan metode yang lain pada tahun 1980-an dimana mengubah persegi dengan sudut lengkung (pada DFD Crish Gane dan Trish Sarson) dengan lingkaran untuk menotasikan. DFD Edward Yourdon dan Tom DeMarco populer digunakan sebagai model analisis sistem perangkat lunak untuk sistem perangkat lunak untuk sistem perangkat lunak yang akan diimplementasikan dengan pemograman terstruktur. Informasi yang ada di dalam perangkat lunak dimodifikasi dengan beberapa transformasi yang dibutuhkan. *Data Flow Diagram* (DFD)atau dalam bahasa Indonesia menjadi Diagram Alir Data (DAD) adalah representasi grafik yang menggambarkan aliran informasi dan transformasi informasi yang diaplikasikan sebagai data yang mengalir dari masukan (*input*) dan keluaran (*output*).

DFD dapat digunakan untuk mempresentasikan sebuah sistem atau perangkat lunak pada beberapa level abstraksi. DFD dapat dibagi menjadi beberapa *level* yang lebih detail untuk mempresentasikan aliran informasi atau fungsi yang lebih detail. DFD lebih sesuai digunakan untuk memodelkan fungsi-fungsi perangkat lunak yang akan diimplementasikan menggunakan pemograman

terstruktur, karena pemograman terstruktur membagi-bagi bagiannya dengan fungsi-fungsi dan prosedur-prosedur (Rosa, 2011).

DFD tidak sesuai untuk memodelkan sistem perangkat lunak yang akan dibangun menggunakan pemograman berorientasi objek. Paradigma pemograman terstruktur dan pemograman berorientasi objek merupakan hal yang berbeda. Simbol-simbol pada DFD Edward Yourdon dan Tom DeMarco adalah seperti pada Tabel 2.3.

Tabel 2.3 Simbol *Data Flow Diagram* (DFD)

No.	Simbol	Keterangan
1.		Proses atau fungsi atau prosedur; pada pemodelan perangkat lunak yang akan diimplementasikan dengan pemograman terstruktur, maka pemodelan notasi inilah yang harusnya menjadi fungsi atau prosedur di dalam kode program. Nama yang diberikan pada sebuah proses biasanya berupa kata kerja.
2.		*File* atau basis data atau penyimpanan (*storage*); pada pemodelan perangkat lunak yang akan diimplementasikan dengan pemograman terstruktur, maka pemodelan notasi inilah yang harusnya dibuat menjadi tabel-tabel basis data yang dibutuhkan, tabel-tabel ini juga harus sesuai dengan perancangan tabel-tabel pada basis data (*Entity Relationship Diagram*, *Conseptual Data Model*, *Physical Data Model*). Nama yang diberikan pada sebuah penyimpanan biasanya kata benda.

3.		Entitas luar (*external entity*)atau masukan (*input*) atau keluaran (*output*) atau orang yang memakai/berinteraksi dengan perangkat lunak yang dimodelkan atau sistem lain yang terkait dengan aliran data dari sistem yang dimodelkan. Nama yang digunakan pada masukan (*input*) atau keluaran (*output*) biasanya berupa kata benda.
4.		Aliran data; merupakan data yang dikirim antar proses, dari penyimpanan ke proses atau dari proses ke masukan (*input*) atau keluaran (*output*). Nama yang digunakan pada aliran data biasanya berupa kata benda, dapat diawali dengan kata data misalnya "data siswa" atau tanpa kata data seperti "siswa".

Tahapan-tahapan perancangan dengan menggunakan DFD adalah sebagai berikut:

a. Membuat DFD Level 0 (Diagram Konteks)

   DFD level 0 menggambarkan sistem yang akan dibuat sebagai suatu entitas tunggal yang berinteraksi dengan orang maupun sistem lain. DFD level 0 digunakan untuk menggambarkan interaksi antara sistem yang akan dikembangkan dengan entitas luar.

b. Membuat DFD Level 1

   DFD level 1 digunakan untuk menggambarkan modul-modul yang ada dalam sistem yang akan dikembangkan.

c. Membuat DFD Level 2

   DFD level 2 merupakan turunan dari DFD level 1. DFD level 2 lebih detail tergantung pada tingkat kedetailan modul tersebut.

d. Membuat DFD Level 3 dan seterusnya.
   DFD level 3, 4, 5 dan seterusnya merupakan breakdown dari modul pada DFD level di atasnya. Breakdown pada level 3, 4, 5 dan seterusnya aturannya sama persisdengan DFD level 1 atau level 2.

2.4.3 Basis Data

Basis data (*database*) adalah suatu pengorganisasian sekumpulan data yang saling terkait sehingga memudahkan aktivitas untuk memperoleh informasi. Basis data di maksudkan untuk mengatasi problem pada sistem yang memakai pendekatan berbasis berkas.

Untuk mengelola basis data diperlukan perangkat lunak yang disebut *Database Management System* (DBMS). DBMS adalah perangkat lunak sistem yang memungkinkan para pemakai membuat, memelihara, mengontrol, dan mengakses basis data dengan cara yang praktis dan efisien. DBMS dapat digunakan untuk mengakomodasikan berbagai macam pemakai yang memiliki kebutuhan akses yang berbeda-beda.

Gambar di atas menunjukan bahwa suatu aplikasi dapat berkomunikasi melalui DBMS untuk mengakses basis data dan kemudian membetuk laporan-laporan Selain itu, pemakai juga bisa berinyeraksi secara langsung dengan DBMS untuk mengakses basis data, baik untuk keperluan memintaataupun untuk melakukan perubahan data. Interaksi secara langsung denganbasis data memungkinkan pemakai untuk memperoleh informasi-informasi yang dibutuhkan sewaktu-waktu dan bersifat sementara, tanpa memerlukan bantuan pemrogram.

Umumnya DBMS menyediakan fitur-fitur sebagai berikut :

a. Independensi data program
   Karena basia data ditangani oleh DBMS, program dapat ditulis sehingga tidak tergantung pada stuktur data dalam basis data. Dengan perkataan lain, program tidak akan terpengaruh sekiranya bentuk fisik data diubah.
b. Keamanan

Keamanan dimaksudkan untuk mencegah pengaksesan data oleh orang yang tidak berwewenang.

c. Integritas

Hal ini ditujukan untuk menjaga agar data selalu dalam keadaan yang valid dan konsisten.

d. Konkurensi

Konkurensi memungkinkandata dapat diakses oleh banyak pemakai tanpa menimbulkan masalah.

e. Pemulihan (*recovery*)

DBMS menyediakan mekanisme untuk mengembalikan basis data ke keadaansemula yang konsisten sekiranya terjadi gangguan perangkat keras atau kegagalan perangkat lunak.

f. Katalog sistem

Katalog sistem adalah deskripsi tentang data yang terkandung dalam basis data yang dapat diakses oleh pemakai.

g. Perangkat produktivitas

Untuk menyediakan kemudahan bagi pemakai dan meningkatkan produktivitas, DBMS menyediakan sejumlah perangkakat produktivitas seperti pembangkit query dan pembangkit laporan.

Komponen-komponen yang menyusun lingkungan DBMS terdiri atas:

a. Perankat keras

Perangkat keras digunakan untuk menjalankan DBMS beserta aplikasi-aplikasinya. Perangkat keras berupa komputer dan periferal pendukungnya. Komputer dapat berupa PC, minikomputer, mainframe, dan lain-lain.

b. Perangkat lunak

Komponen perangkat lunak mencakup DBMS itu sendiri, program aplikasi, serta perangkat lunak pendukung untuk komputer dan jaringan. Program aplikasi dapat dibangun dengan menggunakan bahasa pemrograman seperti C++, Pascal, Delphi, atau Visual BASIC.

c. Data

Bagi sisi pemakai, komponen terpenting dalam DBMS adalah data karena dari data inilah pemakai dapat memperoleh informasi yang sesuai dengan kebutuhan masing-masing.

d. Prosedur

Prosedur adalah petujuk tertulis yang berisi cara merancang hingga menggunakan basis data. Beberapa hal yang dimasukkan dalam prosedur:

1. Cara masuk ke DBMS (login).
2. Cara memekai fasilitas-fasilitas tertentu dalam DBMS maupun cara menggunakan aplikasi.
3. Cara mengaktifkan dan menhentikan DBMS.
4. Cara membuat cadangan basis data dan cara mengembalikan cadangan ke DBMS.

e. Orang

Komponen orang dapat dibagi menjadi tiga kelompok, yaitu :

a. Pemakai akhir (end-user).
b. Pemogram aplikasi.
c. Administrator basis data.

2.4.3.1 Konsep Basis Data

Basis data adalah kumpulan dari data yang saling berhubungan dengan satu dengan yang lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya (Rosa, 2011). Dari definisi ini terdapat tiga hal yang berhubungan dengan basis data, yaitu:

a. Data itu sendiri yang di organisasikan dalam bentuk basis data *(database).*
b. Simpanan permanen *(storage)* untuk menyimpan basis data tersebut. Simpanan ini merupakan bagian dari teknologi perangkat keras yang digunakan di sistem informasi.
c. Perangkat lunak untuk memanipulasi basis datanya. Perangkat lunak ini dapat dibuat sendiri dengan menggunakan bahasa pemrograman komputer atau dibeli dalam bentuk suatu paket.

Tedapat beberapa elemen basis data, yaitu :

a. Database

Database atau basis data adalah kumpilan tabel yang mempunyai kaitan antara suatu tabel dengan tabel lainya sehingga membentuk suatu bangunan data.

b. Tabel

Tabel adalah kumpulan *record-record* yang mempunyai panjang elemen yang sama dan atribut yang sama namun berbeda data *value*nya.

c. Entitas

Entitas adalah sekumpulan objek yang terdefinisikan yang mempunyai karakteristik sama dan bisa dibedakan satu dengan lainnya. Objek dapat berupa barang, orang, tempat atau suatu kejadian.

d. Atribut

Atribut adalah deskripsi data yang bisa mengidentifikasi entitas yang membedakan entitas tersebut dengan entitas yang lain. Seluruh atribut harus cukup untuk menyatakan identitas objek atau dengan kata lain, kumpulan atribut dari setiap entitas dapat mengidentifikasi keunikan suatu individu.

e. *Data Value* (Nilai Data)

Data value adalah data aktual atau informasi yang disimpan pada tiap data, elemen atau atribut. Atribut nama pegawai menunjukan tempat dimana informasi nama karyawan disimpan, nilai datanya misalnya adalah Anjang, Arif, Suryo dan lain-lain yang merupakan isi data nama pegawai tersebut.

f. File

File adalah kumpulan *record* sejenis yang mempunyai panjang elemen yang sama, atribut yang sama namun berbeda nilai datanya.

g. Record/Tuple

Kumpulan elemen-elemen yang saling berkaitan menginformasikan tentang suatu entitas secara lengkap. Satu *record* mewakili satu data atau informasi.

2.5 Kebutuhan Perangkat Lunak

2.5.1 NetBeans

Netbeans adalah suatu *tool* untuk membuat program dengan menggunakan bahasa pemrograman *Java* berbasis grafis. Cara membuat program dengan merancang tampilan menggunakan komponen visual dan proses diletakan pada *event driven*. *Netbeans* menyediakan sekumpulan perangkat lunak modular yang disebut modul yang dipakai untuk membangun suatu aplikasi. Sebuah modul adalah merupakan arsip Java *(Java Archive)* yang memuat kelas-kelas *Java* yang berinteraksi dengan *netbeans Open API*. Kemudian rancangan tampilan yang dibuat menggunakan *netbeans* programnya secara otomatis akan di*generate* menjadi kode (Wahana Komputer, 2010).

2.5.2 Java

Java adalah bahasa pemrograman berorientasi objek yang dikembangkan oleh Sun Microsystems sejak tahun 1991. Bahasa ini dikembangkan dengan model yang mirip dengan bahasa C++ dan *smalltalk*, namun dirancang agar lebih mudah dipakai dan *platform independent*, yaitu dapat dijalankan di berbagai jenis sistem operasi dan arsitektur komputer. Bahasa ini juga dirancang untuk pemrograman di internet sehingga dirancang agar aman dan portabel.

Platform independent berarti program yang ditulis dalam bahasa *Java* dapat dengan mudah dipindahkan antar berbagai jenis sistem operasi dan berbagai jenis arsitektur komputer. Aspek ini sangat penting untuk dapat mencapai tujuan *Java* sebagai bahasa pemrograman internet di mana sebuah program akan dijalankan oleh berbagai jenis komputer dengan berbagai jenis sistem operasi. Sifat ini berlaku untuk level *source code* dan *binary code* dari program *Java*. Berbeda dengan bahasa C dan C++, semua tipe data dalam bahasa Java mempunyai ukuran yang konsisten disemua jenis *platform. Source code* program *Java* sendiri tidak perlu dirubah sama sekali jika Anda ingin mengkompile ulang di *platform* lain. Hasil dari mengkompile *source code Java* bukanlah kode mesin atau instruksi prosesor yang spesifik terhadap mesin tertentu, melainkan berupa

bytecode yang berupa *file* berekstensi *.class. Bytecode* tersebut dapat langsung dieksekusi di tiap *platform* yang dengan menggunakan *Java Virtual Machine (JVM)* sebagai interpreter terhadap *bytecode* tersebut (Joice, 2007).

2.5.3 MySQL

MySQL adalah salah satu jenis *database server* yang terkenal dan banyak digunakan untuk membangun aplikasi web yang menggunakan *database* sebagai sumber dan pengelolaan data. Kepopuleran *MySQL* antara lain karena *MySQL* menggunakan *SQL (Structured Query Language)* sebagai bahasa dasarnya untuk mengakses *database*nya sehingga mudah untuk digunakan.

MySQL termasuk *RDBMS (Relational Database Management System).* Pada *MySQL*, sebuah *database* mengandung satu atau sejumlah tabel. Tabel terdiri atas sejumlah kolom dan baris, dimana setiap kolom berisi sekumpulan data, dan baris merupakan sekumpulan data yang saling berkaitan dan membentuk informasi. Kolom biasanya disebut sebagai *field* dan informasi yang tersimpan dalam setiap baris disebut *record* (Rudyanto, 2011).

2.6 Penelitian Terkait

Penelitian yang terkait dengan penelitian yang sudah dilakukan sebelumnya adalah sebagai berikut :

a. Menurut Aji dalam penelitiannya "Sistem Pendukung Keputusan Menentukan Guru Teladan di SMPN 24 Semarang dengan Menggunakan Metode *Simple Additive Weighting*" menyimpulkan bahwa sistem yang telah dibuat ini membuktikan bahwa metode *Simple Additive Weighting* dapat diterapkan dan telah dibuktikan dalam tahap pengujian sistem.
b. Menurut Ardi dalam penelitiannya "Analisis dan Perancangan Sistem Pendukung Keputusan Penilaian Kinerja Guru (PKG) Menggunakan Metode *Simple Additive Weighting* (SAW) Pada SD Negeri 1 Wonoroto Berbasis *Website*" menyimpulkan bahwa sistem yang telah dibuat dapat digunakan

untuk menyelesaikan pengambilan keputusan dengan beberapa kriteria yang akan menjadi bahan pertimbangan menggunakan *Simple Additive Weighting* (SAW) untuk mendapakan alternatif keputusan dengan nilai tertinggi. Pengujian antara hasil perhitungan SAW dengan perhitungan manual menunjukkan hasil yang sama.

c. Menurut Isnaini dalam penelitiannya "Sistem Pendukung Keputusan Pemilihan Guru Berprestasi dengan *Simple Additive Weighting*" menyimpulkan bahwa sistem pendukung keputusan pemilihan guru berprestasi ini dapat membantu mempermudah pelaksanaan pemilihan guru berprestasi, dalam hal pendaftaran, pengumpulan dokumen, hingga proses perhitungan nilai dan penentuan hasil perangkingan yang pada sistem sebelumnya dilakukan secara manual. Metode *Simple Additive Weighting* (SAW) dapat diterapkan pada sistem pendukung keputusan pemilihan guru berprestasi untuk memberikan alternatif hasil perangkingan dan penentuan sebuah alternatif yang memiliki nilai preferensi terbaik dari alternatif yang lain.
d. Menurut Purwi A dalam penelitiannya "Rancang Bangun Sistem Pendukung Keputusan Penilaian Kinerja Pegawai Negeri Sipil(PNS) di Lingkungan Kantor Imigrasi Kalianda Menggunakan Metode *Simple Additive Weighting* (SAW)" menyimpulkan bahwa sistem pendukung keputusan penilaian kinerja pegawai ini dapat membantu dalam mengidentifikasi masalah dalam sistem perangkingan yang dilakukan berdasarkan penilaian secara manual dan dihitung berdasarkan kriteria-kriteria yang telah di tentukan.proses perangkingan pegawai dan pencarian data penilaian kinerja pegawai dapat dilakukan secara mudah dan cepat.Sistem informasi yang dihasilkan merupakan sebuah sistem informasi yang berbasiskan komputer,dengan sistem ini proses penunjang keputusan penilaian kinerja pegawai yang terbaik akan lebih efektif.
e. Menurut Dika dalam penelitiannya "Rancang Bangun Sistem Pendukung Keputusan Penentuan Prestasi Kepengurusan Pada Organisasi Kemahasiswaan IBI Darmajaya Menggunakan Metode *Simple Additive Weighting* (SAW)" menyimpulkan bahwa menerapkan metode SAW (*Simple Additive Weighting)* pada sistem penentuan prestasi kepengurusan pada organisasi kemahasiswaan

ibi darmajaya guna membantu dalam menentukan dan memilih organisasi dalam satu periode masa jabatan serta mempermudah proses penilaian dalam melakukan perangkingan,mempermudah dan mempercepat proses pembuatan laporan penilaian organisasi kemahasiswaan Ibi Darmajaya