BAB II
LANDASAN TEORI

2.1. Sistem Pendukung Keputusan

2.1.1 Pengertian Sistem Pendukung Keputusan

Sistem pendukung keputusan (SPK) adalah bagian dari sistem informasi berbasis komputer termasuk sistem berbasis pengetahuan atau manajemen pengetahuan yang dipakai untuk mendukung pengambilan keputusan dalam suatu organisasi atau perusahaan. Dapat juga dikatakan sebagai sistem komputer yang mengolah data menjadi informasi untuk mengambil keputusan dari masalah semi terstruktur yang spesifik.

Menurut Moore dan Chang, SPK dapat digambarkan sebagai sistem yang berkemampuan mendukung analisis *ad hoc* data, dan pemodelan keputusan, berorientasi keputusan, orientasi perencanaan masa depan, dan digunakan pada saat-saat yang tidak biasa.

Sedangkan menurut Keen dan Scoot Morton Sistem Pendukung Keputusan merupakan penggabungan sumber-sumber kecerdasan individu dengan kemampuan komponen untuk memperbaiki kualitas keputusan. Sistem Pendukung Keputusan juga merupakan sistem informasi berbasis komputer untuk manajemen pengambilan keputusan yang menangani masalah-masalah semi struktur .

Dengan pengertian diatas dapat dijelaskan bahwa SPK bukan merupakan alat pengambilan keputusan, melainkan merupakan sistem yang membantu pengambil keputusan dengan melengkapi mereka dengan informasi dari data yang telah diolah dengan relevan dan diperlukan untuk membuat keputusan tentang suatu masalah dengan lebih cepat dan akurat. Sehingga sistem ini tidak dimaksudkan untuk menggantikan pengambilan keputusan dalam proses pembuatan keputusan.

2.1.2 Konsep Dasar Sistem Pendukung Keputusan

Sistem pendukung keputusan (SPK) mulai dikembangkan pada tahun 1960-an, tetapi istilah sistem pendukung keputusan itu sendiri baru muncul pada tahun 1971, yang diciptakan oleh G. Anthony Gorry dan Micheal S.Scott Morton, keduanya adalah profesor di MIT. Hal itu mereka lakukan dengan tujuan untuk menciptakan kerangka kerja guna mengarahkan aplikasi komputer kepada pengambilan keputusan manajemen.

Sementara itu, perintis sistem pendukung keputusan yang lain dari MIT, yaitu Peter G.W. Keen yang bekerja sama dengan Scott Morton telah mendefenisikan tiga tujuan yang harus dicapai oleh sistem pendukung keputusan, yaitu :

1. sistem harus dapat membantu manajer dalam membuat keputusan guna memecahkan masalah semi terstruktur;
2. sistem harus dapat mendukung manajer, bukan mencoba menggantikannya;
3. sistem harus dapat meningkatkan efektivitas pengambilan keputusan manajer.

2.1.3 Kriteria Sistem Pendukung Keputusan

Sistem pendukung keputusan dirancang secara khusus untuk mendukung seseorang yang harus mengambil keputusan-keputusan tertentu. Berikut ini beberapa karakteristik sistem pendukung keputusan (Oetomo, 2002) :

a. interaktif

   SPK memiliki *user interface* yang komunikatif sehingga pemakai dapat melakukan akses secara cepat ke data dan memperoleh informasi yang dibutuhkan;

b. fleksibel

   SPK memiliki sebanyak mungkin variabel masukkan, kemampuan untuk mengolah dan memberikan keluaran yang menyajikan alternatif-alternatif keputusan kepada pemakai;

c. data kualitas

SPK memiliki kemampuan menerima data kualitas yang dikuantitaskan yang sifatnya subyektif dari pemakainya, sebagai data masukkan untuk pengolahan data. Misalnya: penilaian terhadap kecantikan yang bersifat kualitas, dapat dikuantitaskan dengan pemberian bobot nilai seperti 75 atau 90;

d. prosedur pakar

SPK mengandung suatu prosedur yang dirancang berdasarkan rumusan formal atau juga beberapa prosedur kepakaran seseorang atau kelompok dalam menyelesaikan suatu bidang masalah dengan fenomena tertentu.

2.1.4 Karakteristik dan Kemampuan Sistem Pendukung Keputusan

Menurut (Turban, 2005), ada beberapa karakteristik dari SPK, di antarannya adalah sebagai berikut :

1. mendukung seluruh kegiatan organisasi;
2. mendukung beberapa keputusan yang saling berinteraksi;
3. dapat digunakan berulang kali dan bersifat konstan;
4. terdapat dua komponen utama, yaitu data dan model;
5. menggunakan baik data ekternal maupun internal;
6. memiliki kemampuan *what-if analysis* dan *goal seeking analysis;*
7. menggunakan beberapa model kuantitatif.

Selain itu, Turban juga menjelaskan kemampuan yang harus dimiliki oleh sebuah sistem pendukung keputusan, di antaranya adalah sebagai berikut:

1. menunjang pembuatan keputusan manajemen dalam menangani masalah semi terstruktur dan tidak terstruktur;
2. membantu manajer pada berbagai tingkatan manajemen, mulai dari manajemen tingkat atas sampai manajemen tingkat bawah;

3. menunjang pembuatan keputusan secara kelompok dan perorangan;
4. menunjang pembuatan keputusan yang saling bergantungan dan berurutan;
5. menunjang tahap-tahap pembuatan keputusan antara lain *intelligence, design, choice* dan *implementation;*
6. menunjang berbagai bentuk proses pembuatan keputusan dan jenis keputusan;
7. kemampuan untuk melakukan adaptasi setiap saat dan bersifat fleksibel;
8. kemudahan melakukan interaksi sistem.
9. meningkatkan efektivitas dalam pembuatan keputusan daripada efisiensi;
10. mudah dikembangkan oleh pemakai akhir;
11. kemampuan pemodelan dan analisis dalam pembuatan keputusan;
12. kemudahan melakukan pengaksesan berbagai sumber dan format data.

Disamping berbagai kemampuan dan karakteristik seperti dikemukakan di atas, sistem pendukung keputusan memiliki juga keterbatasan, antara lain:

1. ada beberapa kemampuan manajemen dan bakat manusia yang tidak dapat dimodelkan, sehingga model yang ada dalam sistem tidak semuanya mencerminkan persoalan yang sebenarnya;
2. kemampuan suatu sistem pendukung keputusan terbatas pada pengetahuan dasar serta model dasar yang dimilikinya;
3. proses-proses yang dapat dilakukan oleh sistem pendukung keputusan biasanya tergantung juga pada kemampuan perangkat lunak yang digunakannya;
4. sistem pendukung keputusan tidak memiliki intuisi seperti yang dimiliki oleh manusia. Karena sistem pendukung keputusan hanya suatu kumpulan perangkat keras, perangkat lunak dan sistem operasi yang tidak dilengkapi oleh kemampuan berpikir (Turban, 2005).

Secara implisit, sistem pendukung keputusan berlandaskan pada kemampuan dari sebuah sistem berbasis komputer dan dapat melayani penyelesaian masalah.

2.1.5 Komponen Sistem Pendukung Keputusan

Adapun komponen-komponen dari SPK adalah sebagai berikut:

1. *data Management*

   Termasuk *database*, yang mengandung data yang relevan untuk berbagai situasi dan diatur oleh *software* yang disebut *Database Management Sistem* (DBMS);

2. *model Management*

   Melibatkan model finansial, statistikal, management science, atau berbagai model kualitatif lainnya, sehingga dapat memberikan ke sistem suatu kemampuan analitis, dan manajemen software yang dibutuhkan;

3. *communication*

   User dapat berkomunikasi dan memberikan perintah pada DSS melalui subsistem ini. Ini berarti menyediakan antarmuka.;

4. *knowledge Management*

   Subsistem optional ini dapat mendukung subsistem lain atau bertindak atau bertindak sebagai komponen yang berdiri sendiri (Turban, 2005).

2.2 Object Oriented Programming

Object oriented programming adalah sebuah metode pemrograman dimana pengembang aplikasi tidak hanya mendefinisikan variabel yang berisi *state* dari sebuah struktur data, tetapi juga mendefinisikan fungsi untuk menunjukkan *behavior* yang diaplikasikan pada struktur data. Dalam hal ini, struktur data merupakan objek. Suatu objek dapat saling berkomunikasi satu sama lain dengan menggunakan fungsi yang ada di dalamnya tanpa perlu mengetahui *internal state* masing-masing objek (*data encapsulation*) (Webopedia, n.d).

Salah satu keuntungan dari *object oriented programming* dibandingkan *procedural programming* adalah memungkinkan pengembang aplikasi untuk membuat fungsi yang tidak perlu diubah ketika sebuah objek dengan tipe berbeda ditambahkan.

Seorang pengembang aplikasi hanya perlu membuat objek baru yang mewarisi beberapa fungsi atau tipe data dari objek yang sudah ada (*inheritance*). Hal ini membuat *object oriented programming* mudah dalam
pengembangannya (Webopedia, n.d).

Menurut Bennett, McRobb, dan Farmer (2006), dalam bukunya "*Object-Oriented System Analysis And Design Using UML*", terdapat beberapa konsep dalam *object oriented programming,* yaitu :

a. objek

Objek terdiri dari *state* dan *behavior.* Sebuah objek menyimpan *state* pada variabel dan menunjukkan *behavior* melalui fungsi.

Fungsi bertugas mengubah *internal state* dan melayani komunikasi antara satu objek dengan objek lainnya (Oracle, n.d);

b. class

Class adalah sekumpulan objek yang memiliki *state* dan *behavior* yang sama;

c. instance

Instance adalah objek tunggal yang memiliki nilai *state* yang secara keseluruhan berbeda dengan objek lain;

d. generalisasi

Generalisasi adalah konsep yang mengelompokkan *state* dan *behavior* yang sama dari beberapa *class* menjadi *class* tersendiri.

Berikut adalah ciri-ciri dari generalisasi :

1. *inheritance*

Mekanisme untuk mengimplementasikan generalisasi dalam bahasa pemrograman berorientasi objek. *Class* yang memiliki *state* dan *behavior* dari hasil pengelompokkan disebut *superclass* sedangkan *class* yang memiliki *state* dan *behavior* yang unik karena pengelompokkan disebut *subclass*;

2. operasi transitif

Operasi yang menurunkan *state* dan *behavior* dari *superclass* ke *subclass;*

3. *disjoint nature*

Aspek dimana *class* harus memiliki *state* dan *behaviour* yang unik dari *class* lain yang satu level.

e. message passing

Message passing adalah komunikasi yang dilakukan objek untuk meminta informasi tertentu dari objek lain dengan menggunakan fungsi dari objek lain.

2.2.1 Unified Modeling Language (UML)

Unified modeling language adalah bahasa yang digunakan untuk memodelkan sebuah sistem sehingga dapat mengambil keputusan dan memahami tentang sistem yang harus dibangun (Hermawan, 2004).

Menurut Whitten dan Bentley (2007), UML menawarkan sembilan *diagram* yang dikelompokkan ke dalam lima perspektif yang berbeda untuk memodelkan sebuah sistem. Lima perspektif tersebut adalah :

a. use-Case Model Diagrams

Diagram yang ada dalam kelompok ini adalah *use-case diagram* (*diagram* yang menggambarkan fungsi sistem dari sudut pandang pengguna);

b. static Structure Diagrams

Diagram yang ada dalam kelompok ini adalah *class diagram* (*diagram* yang menggambarkan struktur objek dalam sistem) dan *object diagram* (*diagram* yang menggambarkan objek *instance* yang aktual);

c. interaction Diagrams

Interaction diagram memodelkan sebuah interaksi yang terdiri dari sekumpulan objek, hubungannya, dan pesan yang dikirimkan diantara objek tersebut. *Diagram* yang ada dalam kelompok ini adalah *sequence diagram (diagram* yang menggambarkan interaksi antar objek dimana fokus pada *timing* dari pesan) dan *communication diagram* (*diagram* yang menggambarkan interaksi antara objek dengan *network format*);

d. state Diagrams

Diagram yang ada dalam kelompok ini adalah *statechart diagram (diagram* yang mengilustrasikan daur hidup objek) dan *activity diagram* (*diagram* yang menggambarkan aliran dari sebuah *use case*);

e. Implementation Diagrams

Diagram yang ada dalam kelompok ini adalah *component diagram (diagram* yang menggambarkan bagaimana *code* program dipecah menjadi komponen-kompnone dan ketergantungan diantara komponen tersebut) dan *deployment diagram (*diagram yang menggambarkan arsitektur secara fisik dari *hardware* dan *software* dalam sistem).

2.2.1.1 *Use Case Diagram*

Use case adalah deskripsi fungsi sistem dari sudut pandang pengguna. *Use case diagram* digunakan untuk menunjukkan fungsi yang sistem akan sediakan dan menunjukkan pengguna mana yang akan berkomunikasi dengan sistem. *Use case diagram* ini dikembangkan oleh Jacobson et al. (1992), dan judul buku dimana *use case diagram* ini ditampilkan adalah *A Use Case Driven Approach* (Bennett, McRobb, dan Farmer, 2006).

Menurut Bennett, McRobb, dan Farmer (2006), *use case diagram* menunjukkan tiga aspek dalam sistem yaitu:

a. actors

Actors mewakili peran yang dimiliki orang, sistem lain, atau *device* ketika berkomunikasi dengan *use cases* tertentu dalam sistem dimana satu *actor* dapat mewakili beberapa orang atau pekerjaan;

b. use case

Sebuah *use case* menggambarkan sebuah fungsi yang dilakukan oleh sistem untuk mencapai tujuan pengguna dimana digambarkan dengan bentuk *eclipse;*

c. batasan sistem atau subsistem

Batasan yang mengelilingi fungsi-fungsi dalam sistem atau subsistem dimana berbentuk persegi panjang.

2.2.1.2 Class Diagram

Class diagram adalah sebuah *diagram* yang menggambarkan objek-objek dari sistem dan hubungan antara objek tersebut (Bennett, McRobb, dan Farmer, 2006). Menurut Bennett, McRobb, dan Farmer (2006), *class diagram* memiliki istilah-istilah penting yang perlu diketahui, yaitu :

a. attributes

Attributes adalah *data* yang menampilkan karakteristik utama dari sebuah *class*. Oleh karena itu, setiap objek dari *class* memiliki *attribute* yang dimiliki oleh *class* tersebut;

b. association

Sebuah *association* adalah hubungan antara dua *class* yang berbeda dimana ditandai dengan nama hubungan, garis lurus yang menghubungkan kedua *class* tersebut, dan tanda panah yang membantu pemahaman ketika melihat *class diagram*;

c. multiplicity

Multiplicity adalah istilah yang digunakan untuk menerangkan batasan pada jumlah objek yang dapat berpartisipasi. *Multiplicity* merefleksikan aturan-aturan suatu perusahaan yang merupakan batasan pada cara aktivitas bisnis berjalan. Notasi *multiplicity* dapat berupa kisaran nilai seperti 0..*, 0..3, 1..5, 2..10 atau berupa nilai tunggal, seperti 1, 3, 5, 19;

d. operations

Operations adalah *behavior* yang dapat dilakukan oleh objek atau dilakukan pada objek. *Behavior* tersebut diimplementasikan melalui fungsi. Dalam komunikasi

dengan objek lain, suatu objek hanya perlu memanggil fungsi dari objek tersebut tanpa perlu mengetahui *attribute* yang ada.

2.2.1.3 Sequence Diagram

Sequence diagram adalah salah satu dari beberapa macam UML *interaction diagram.* Sebuah *sequence diagram* menunjukkan sebuah interaksi antara objek-objek yang disusun dalam urutan waktu. Kegunaan dari *sequence diagram* adalah menunjukkan interaksi objek secara rinci yang terjadi untuk satu *use case* (Bennett, McRobb, dan Farmer, 2006).

Dalam *sequence diagram,* dimensi vertikal menunjukkan waktu dan semua objek yang ikut terlibat dalam interaksi tersebar secara horizontal pada *diagram.* Waktu secara normal berjalan ke bawah. Setiap objek dalam sebuah *sequence diagram* ditunjukkan oleh sebuah *lifeline. Lifeline* adalah sebuah garis vertikal putus-putus dengan symbol objek pada bagian paling atas. Sebuah pesan ditunjukkan dengan garis panah horizontal dari satu *lifeline* ke *lifeline* yang lain dan diberi label dengan nama *message.* Setiap nama *message* secara opsional didahului oleh sebuah nomor urut yang menunjukkan urutan dimana *message* dikirim, tetapi ini biasanya tidak perlu pada sebuah *sequence diagram* karena urutan *message* telah disampaikan oleh posisi relatif mereka selama sumbu waktu (Bennett, McRobb, dan Farmer, 2006).

2.2.1.4 Activity Diagram

Activity diagram dapat digunakan untuk memodelkan aspek-aspek yang berbeda pada sebuah sistem. *Activity diagram* dapat digunakan untuk memodelkan aktivitas bisnis pada sistem yang ada, dapat juga digunakan pada tahap awal dalam *system development lifecycle,* yaitu dengan memodelkan sebuah fungsi sistem yang ditunjukkan oleh sebuah *use case*, menggunakan aliran objek untuk menunjukkan objek mana yang terlibat dalam sebuah *use case* (Bennett, McRobb, dan Farmer, 2006).

Activity diagram terdiri atas sekumpulan aksi yang dihubungkan bersama oleh aliran dari satu aktivitas ke aktivitas lainnya. Setiap aksi ditunjukkan sebagai persegi

dengan sudut yang melingkar. Nama dari aksi ditulis di dalam simbol dua dimensi ini. Hal ini menjadi penuh arti dan merangkum aksi tersebut (Bennett, McRobb, dan Farmer, 2006).

Selanjutnya, jika ingin mendeskripsikan beberapa pilihan yang dapat diambil, digunakan *explicit decision node*, yang ditunjukkan dengan ikon berbentuk *diamond*. Setiap aliran alternatif tersebut diberi *label* dengan sebuah *guard condition. Guard condition* ditunjukkan di dalam simbol *bracket* (Bennett, McRobb, dan Farmer, 2006).

2.2.1.5 Deployment Diagram

Deployment diagram adalah tipe *implementation diagram* yang mendeskripsikan arsitektur fisik dari perangkat keras dan perangkat lunak dalam sistem. *Diagram* ini menggambarkan komponen perangkat lunak, *processor,* dan *device* yang membentuk arsitektur sistem. Setiap kotak yang ada pada *diagram* menggambarkan satu buah perangkat keras. Perangkat keras dapat berupa PC, *mainframe, printer,* atau *smartphone.* Perangkat lunak yang terdapat dalam kotak ditandai dengan symbol komponen. Garis yang menghubungi kotak mengindikasikan sebuah komunikasi antar *device.* Komunikasi tersebut ditandai dengan tipe protokol komunikasi yang digunakan (Whitten dan Bentley, 2007).

2.3 PHP

Pemrograman berbasis *web* berkembang cukup pesat akhir-akhir ini dan bahasa yang digunakan adalah HTML (*HyperText Markup Language*), akan tetapi bahasa HTML hanya terbatas pada pembuatan *website* statis (*website* yang tidak dapat berinteraksi dengan *user*). Dalam perkembangannya *website* statis jarang dibuat oleh para *programmer web* dikarenakan *user* lebih tertarik pada *website* yang mempunyai interaktifitas tinggi, untuk membuat *website* yang interaktif para *programmer web* menggunakan bahasa pemrograman PHP (Sidik, 2012).

Bahasa Pemrograman PHP (*HyperText Preprocessor*) merupakan pengembangan dari bahasa pemrograman ASP, perbedaannya adalah dari segi

kompleksitas bagi *programmer web*. PHP cenderung lebih mudah untuk dipelajari karena banyak referensi yang tersedia sehingga para *programmer* dapat mempelajari secara jelas sedangkan referensi untuk ASP lebih sedikit tersedia. Hal tersebut dikarenakan PHP adalah bahasa pemrograman *open source* (gratis) sehingga para *programmer* banyak yang mengembangkan dan menggunakannya tanpa perlu mengeluarkan biaya, sedangkan ASP adalah bahasa pemrograman yang berbayar, sehingga jarang dikembangkan oleh para *programmer* (Sidik, 2012).

PHP merupakan bahasa pemrograman yang sangat kompatibel dengan berbagai macam *platform database*. Macam-macam *platform database* yang komersil maupun non komersil, seperti postgreSQL, SQL, MySQL, Oracle, Microsoft SQL Server, dan lain-lain. PHP juga merupakan bahasa pemograman *web* yang bersifat *server-side* HTML=*embedded scripting*, di mana *script*-nya menyatu dengan HTML dan berada di server. Artinya adalah sintak dan perintah-perintah yang kita berikan akan sepenuhnya dijalankan di *server* tetapi disertakan HTML. PHP sangat didukung oleh *web server* untuk menjalankannya, *web server* ini seperti PWS (*Personal Web Server*), Apache, IIS, AOLServer, fhttpd, phttpd dan sebagainya. PHP juga mendukung komunikasi dengan layanan seperti *protocol* IMAP, SNMP, NNTP, POP3 dan bahkan HTTP (Agus Saputra, 2011).

2.4 MySQL

MySQL merupakan program aplikasi untuk membuat suatu DBMS (*Database* Management System) yang berbasis SQL (*Structured Query Language*). MySQL mempergunakan lisensi *GPL (GNU General Public License)*. Pada sebuah *database* yang dibuat oleh *MySQL* mengandung satu beberapa tabel, tabel terdiri dari sejumlah baris dan kolom. *MySQL* mempunyai beberapa kelebihan dibandingkan dengan yang lainnya misalnya *PostgreSQL*, *Microsoft SQL Server*, dan *Oracle*. Kelebihan *MySQL* adalah pada kecepatan akses, biaya, konfigurasi, tersedia *source code* karena *MySQL* berada di bawah *Open Source License* (Sidik, 2012).

Menurut Ario Santoso, Daniel Cahyadi, Rado Yanu Ardian, Richard Lokasasmita dan Wahyu Mirza (2006), *MySQL* dapat berjalan dengan baik pada banyak jenis *OS platform*. *OS Platform* yang dapat mendukung *MySQL* adalah *AIX, BSDi, FreeBSD, HP-UX, GNU/Linux, Mac OS X, NetBSD, Novell NetWare, OpenBSD, OS/2 Warp, QNX, SGI IRIX, Solaris, SunOS, SCO OpenServer, SCO UnixWare, Tru64, Windows 95, Windows 98, Windows NT, Windows 2000, Windows XP* dan versi-versi *Windows* selanjutnya, sehingga akhir-akhir ini *MySQL* berkembang pesat.

Pada umumnya *MySQL* dikelola dengan beberapa cara yaitu melalui *prompt DOS* dan *program utility*. *Program utility* itu seperti *PHPMyAdmin, MySQLGUI, MySQL Manajer Java Based* dan *MySQL Administrator for windows*. *Program utility* yang sering digunakan adalah *PHPMyAdmin. PHPMyAdmin* merupakan salah satu *tool* manajemen *database MySQL* berbasis *web*, artinya interaksi pemeliharaan dilakukan oleh klien dengan menggunakan antarmuka *browser* (Sidik, 2012).

MySQL mempunyai perintah-perintah yang tergolong *DML* (*Data Manipulation Language*). Menurut Sutarman (2003), *DML* adalah suatu bahasa yang digunakan untuk manipulasi data seperti menambah, menghapus, menampilkan dan mengubah suatu data. Perintah-perintah itu adalah *insert, update, delete dan select.*

2.5 Metode Simple Additive Weighting

Simple Additive Weigting Method merupakan salah satu metode yang dapat digunakan untuk menyelesaikan *MADM (Multi atribut decision making), MADM* merupakan model dari *MCDM (Multiple criteria decision making), MCDM* sendiri adalah suatu metode pengambilan keputusan untuk menetapkan alternatif terbaik dari sejumlah alternatif berdasarkan beberapa kriteria tertentu. Konsep dasar metode ini adalah mencari penjumlahan terbobot dari rating kinerja pada setiap alternatif pada semua atribut. Metode ini membutuhkan proses normalisasi matriks keputusan (X) ke suatu skala yang dapat diperbandingkan dengan semua rating alternatif yang ada (Kusumadewi, 2006).

Metode *Simple Additive Weighting* ini memiliki beberapa kelebihan dan kekurangan di antaranya yaitu :

1. kelebihan :
   a. menentukan nilai bobot dari setiap atribut, kemudian dilanjutkan dengan proses perangkingan yang akan menyeleksi alternatif terbaik dari sejumlah alternatif;
   b. penilaian akan lebih tepat karena didasarkan pada nilai kriteria dan bobot preferensi yang sudah ditentukan.
2. kekurangan :
   a. perhitungan dilakukan dengan bilangan *crisp*;
   b. adanya perbedaan perhitungan normalisasi matriks sesuai dengan nilai atribut (antara nilai *benefit* dan *cost*) (Kusumadewi, 2006).

Langkah Penyelesaian *Simple Additive Weighting (SAW)* sebagai berikut :

1. menentukan kriteria-kriteria yang akan dijadikan acuan dalam pengambilan keputusan, yaitu Ci;
2. menentukan rating kecocokan setiap alternatif pada setiap kriteria;
3. 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;
4. 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 :

$$r_{ij} = \begin{cases} \dfrac{x_{ij}}{\underset{i}{\text{Max}}\, x_{ij}} & \text{jika } j \text{ adalah atribut keuntungan (benefit)} \\ \dfrac{\underset{i}{\text{Min}}\, x_{ij}}{x_{ij}} & \text{jika } j \text{ adalah atribut biaya (cost)} \end{cases}$$

Dimana :

rij = rating kinerja ternormalisasi

Maxij = nilai maksimum dari setiap baris dan kolom

Minij = nilai minimum dari setiap baris dan kolom

Xij = baris dan kolom dari matriks

Dengan rij adalah rating kinerja ternormalisasi dari alternatif Ai pada atribut Cj; 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 :

Vi = Nilai akhir dari alternatif

wj = Bobot yang telah ditentukan

rij = Normalisasi matriks

Nilai Vi yang lebih besar mengindikasikan bahwa alternatif Ai lebih terpilih (aeroyid, 2014).

2.5.1 Contoh Kasus Metode Simple Additive Weighting

Kriteria penilaian karyawan yang layak untuk mendapatkan promosi kenaikan jabatan adalah sebagai berikut:

1. kepemimpinan*/Leadership;*
2. pengetahuan/*Knowledge;*
3. kepribadian*/Personality;*
4. kedisiplinan.

Bobot:

1. Sangat Penting (SP) = 4;
2. Penting (P) = 3;
3. Cukup Penting (CP) = 2;
4. Kurang Penting (KP) = 1.

Tabel-tabel dibawah ini adalah dimana setiap tabel mempunyai kriteria masing-masing dan dengan nilai dan bobotnya. Untuk lebih jelasnya mengenai nilai di masing-masing tabel kriteria dapat dilihat pada tabel-tabel berikut :

Kepemimpinan*/Leadership*	**Bobot**
Nilai >=85 dan <=100	3
Nilai >=70 dan <=84	2
Nilai >=50 dan <=69	1
Nilai >=0 dan <=49	0

Tabel 2.1 Kriteria dan Bobot Kepemimpinan/*Leadership* (C1)

Pengetahuan/*Knowledge*	**Bobot**
Nilai >=85 dan <=100	3
Nilai >=70 dan <=84	2
Nilai >=50 dan <=69	1
Nilai >=0 dan <=49	0

Tabel 2.2 Kriteria dan Bobot Pengetahuan/*Knowledge* (C2)

***Personality*/Kepribadian**	**Bobot**
Nilai >=85 dan <=100	3
Nilai >=70 dan <=84	2
Nilai >=50 dan <=69	1
Nilai >=0 dan <=49	0

Tabel 2.3 Kriteria dan Bobot *Personality*/Kepribadian (C3)

Kedisiplinan	**Bobot**
Nilai >=85 dan <=100	3
Nilai >=70 dan <=84	2
Nilai >=50 dan <=69	1
Nilai >=0 dan <=49	0

Tabel 2.4 Kriteria dan Bobot Kedisiplinan (C4)

Nama Spv	KRITERIA			
	C1	C2	C3	C4
Karyawan 1 (A1)	65	76	80	77
Karyawan 2 (A2)	60	71	75	88
Karyawan 3 (A3)	98	76	87	67
Karyawan 4 (A4)	83	80	65	89

Karyawan 5 (A5)	79	88	60	68

Tabel 2.5 Data Evaluasi/Penilaian Karyawan

Dari tabel penilaian, maka dapat dibuat tabel rating kecocokan setiap alternatif pada setiap kriteria.

Alternatif	KRITERIA			
	C1 (Max)	C2 (Max)	C3 (Max)	C4 (Max)
A1	1	2	2	2
A2	1	2	2	3
A3	3	2	3	1
A4	2	2	1	3
A5	2	3	1	1

Tabel 2.6 Tabel Rating Kecocokan

Langkah-langkah penyelesaian:

1. Vektor bobot : W = [(4), (3), (2), (1)]
2. Matrik Keputusan X berdasarkan kriteria bobot :

$$X = \begin{Bmatrix} 1 & 2 & 2 & 2 \\ 1 & 2 & 2 & 3 \\ 3 & 2 & 3 & 1 \\ 2 & 2 & 1 & 3 \\ 2 & 3 & 1 & 1 \end{Bmatrix}$$

3. Normalisasi Matriks X menggunakan persamaan 1:

Alternatif A1

$$r11 = \frac{1}{\text{Max } (1;1;3;2;2)} = 0{,}33$$

$$r12 = \frac{2}{\text{Max } (2;2;2;2;3)} = 0{,}67$$

$$r13 = \frac{2}{\text{Max } (2;2;3;1;1)} = 0{,}67$$

$$r14 = \frac{2}{\text{Max } (2;3;1;3;1)} = 0{,}67$$

Alternatif A2

$$r21 = \frac{1}{\text{Max } (1;1;3;2;2)} = 0{,}33$$

$$r22 = \frac{2}{\text{Max } (2;2;2;2;3)} = 0{,}67$$

$$r23 = \frac{2}{\text{Max } (2;2;3;1;1)} = 0{,}67$$

$$r24 = \frac{3}{\text{Max } (2;3;1;3;1)} = 1{,}00$$

Alternatif A3

$$r31 = \frac{3}{\text{Max } (1;1;3;2;2)} = 0{,}33$$

$$r32 = \frac{2}{\text{Max } (2;2;2;2;3)} = 0{,}67$$

$$r33 = \frac{3}{\text{Max } (2;2;3;1;1)} = 1{,}00$$

$$r34 = \frac{1}{\text{Max } (2;3;1;3;1)} = 0{,}33$$

Alternatif A4

$$r41 = \frac{2}{\text{Max } (1;1;3;2;2)} = 0{,}67$$

$$r42 = \frac{2}{\text{Max } (2;2;2;2;3)} = 0{,}67$$

$$r43 = \frac{1}{\text{Max } (2;2;3;1;1)} = 0{,}33$$

$$r44 = \frac{3}{\text{Max } (2;3;1;3;1)} = 1{,}00$$

Alternatif A5

$$r51 = \frac{2}{\text{Max } (1;1;3;2;2)} = 0{,}67$$

$$r52 = \frac{3}{\text{Max } (2;2;2;2;3)} = 1{,}00$$

$$r53 = \frac{1}{\text{Max } (2;2;3;1;1)} = 0{,}33$$

$$r54 = \frac{1}{\text{Max } (2;3;1;3;1)} = 0{,}33$$

Dari hasil perhitungan di atas maka didapat matriks ternormalisasi R, yaitu:

$$R = \begin{Bmatrix} 0{,}33 & 0{,}67 & 0{,}67 & 0{,}67 \\ 0{,}33 & 0{,}67 & 0{,}67 & 1{,}00 \\ 1{,}00 & 0{,}67 & 1{,}00 & 0{,}33 \\ 0{,}67 & 0{,}67 & 0{,}33 & 1{,}00 \\ 0{,}67 & 1{,}00 & 0{,}33 & 0{,}33 \end{Bmatrix}$$

4. Mencari alternatif terbaik menggunakan persamaan 2:

V1 = (0,33 x 4) + (0,67 x 3) + (0,67 x 2) + (0,67 x 1) = 5,33

V2 = (0,33 x 4) + (0,67 x 3) + (0,67 x 2) + (1,00 x 1) = 5,67

V3 = (1,00 x 4) + (0,67 x 3) + (1,00 x 2) + (0,33 x 1) = 8,33

V4 = (0,67 x 4) + (0,67 x 3) + (0,33 x 2) + (1,00 x 1) = 6,33

V5 = (0,67 x 4) + (1,00 x 3) + (0,33 x 2) + (0,33 x 1) = 6,67

V3 merupakan peringkat pertama karena memiliki nilai yang lebih besar dari nilai yang lain, **V3** merupakan nilai preferensi dari alternatif A3, sehingga A3

atau dalam kasus ini karyawan A3 yang menjadi alternatif terbaik untuk mendapatkan promosi kenaikan jabatan.