BAB II

LANDASAN TEORI

2.1 Sistem

Definisi sistem dapat dibagi menjadi dua pendekatan yaitu pendekatan secara prosedur dan pendekatan secara komponen. Berdasarkan pendekatan prosedur, sistem didefinisikan sebagai kumpulan dari beberapa prosedur yang mempunyai tujuan tertentu. Sedangkan berdasarkan pendekatan komponen, sistem merupakan kumpulan dari komponen-komponen yang saling berkaitan untuk mencapai tujuan tertentu.

Dalam perkembangan sistem yang ada, sistem dibedakan menjadi dua jenis, yaitu sistem terbuka dan sistem tertutup. Sistem terbuka merupakan sistem yang dihubungkan dengan arus sumber daya luar dan tidak mempunyai elemen pengendali. Sedangkan sistem tertutup tidak mempunyai elemen pengontrol dan dihubungkan pada lingkungan sekitarnya (Herlambang, 2005).

2.2 *Entity Relational Diagram (ERD)*

Pengertian *Entity Relation Diagram* (ERD) menurut Jogiyanto (2001) adalah suatu komponen himpunan entitas dan relasi yang dilengkapi dengan atribut yang mempresentasikan seluruh fakta. ERD digunakan untuk menggambarkan model hubungan data dalam sistem yang didalamnya terdapat hubungan entitas berserta atribut relasinya serta mendokumentasikan kebutuhan sistem untuk pemrosesan data.

2.3 *System Flowchart (SysFlow)*

System Flowchart merupakan diagram alir yang menggambarkan suatu sistem peralatan komputer yang digunakan untuk mengolah data dan menghubungkan antar peralatan tersebut (Oetomo, 2002). Diagram alir sistem ini tidak digunakan untuk menggambarkan langkah-langkah dalam memecahkan masalah tetapi hanya menggambarkan prosedur pada sistem yang dibentuk. Diagram alir sistem digambar dengan menggunakan simbol-simbol tertentu.

2.4 Data Flow Diagram

Menurut Whitten (2004), *Data Flow Diagram* (DFD) merupakan alat yang menggambarkan aliran data melalui sistem. Dalam pembuatan DFD, terdapat beberapa tingkatan yang bertujuan untuk menghindari aliran data yang rumit. Tingkatan tersebut dimulai dari tingkatan tertinggi ke bentuk yang lebih rinci. Tingkatan DFD terdiri atas:

1. Diagram Konteks (*Context Diagram*)

Diagram konteks merupakan sebuah model proses yang digunakan untuk mendokumentasikan ruang lingkup dari sebuah sistem (Whitten, 2004).

2. Diagram level 0

Diagram level 0 merupakan diagram aliran data yang menggambarkan sebuah *event* konteks. Diagram ini menunjukkan interaksi antara *input*, output, dan *data store* pada setiap proses yang ada (Whitten, 2004).

2.5 Pengertian Sistem Pendukung Keputusan

Sistem Pendukung Keputusan (SPK) atau *Decision Support System* (DSS) adalah sebuah sistem yang mampu memberikan kemampuan pemecahan masalah maupun kemampuan pengkomunikasian untuk masalah dengan kondisi semi terstruktur dan tak terstruktur. Sistem ini digunakan untuk membantu pengambilan keputusan dalam situasi semi terstruktur dan situasi yang tidak terstruktur, dimana tak seorangpun tahu secara pasti bagaimana keputusan seharusnya dibuat (Turban, 2001).

SPK bertujuan untuk menyediakan informasi, membimbing, memberikan prediksi serta mengarahkan kepada pengguna informasi agar dapat melakukan pengambilan keputusan dengan lebih baik.SPK merupakan implementasi teori-teori pengambilan keputusan yang telah diperkenalkan oleh ilmu-ilmu seperti *operation research* dan *management science*, hanya bedanya adalah bahwa jika dahulu untuk mencari penyelesaian masalah yang dihadapi harus dilakukan perhitungan iterasi secara manual (biasanya untuk mencari nilai minimum, maksimum, atau optimum), saat ini komputer PC telah menawarkan kemampuannya untuk menyelesaikan persoalan yang sama dalam waktu relatif singkat. Sprague dan Watson mendefinisikan Sistem Pendukung Keputusan (SPK) sebagai sistem yang memiliki lima karakteristik utama yaitu (Sprague et.al, 1993):

1. Sistem yang berbasis komputer.
2. Dipergunakan untuk membantu para pengambil keputusan.
3. Untuk memecahkan masalah-masalah rumit yang mustahil dilakukan dengan kalkulasi manual.

4. Melalui cara simulasi yang interaktif dimana data dan model analisis sebagai komponen utama.

2.6 *Profile Matching*

Menurut Kusrini (2007), metode *profile matching* atau pencocokan profil adalah metode yang sering digunakan sebagai mekanisme dalam pengambilan keputusan dengan mengasumsikan bahwa terdapat tingkat *variable predictor* yang ideal yang harus dipenuhi oleh subyek yang diteliti, bukannya tingkat minimal yang harus dipenuhi atau dilewati. Dalam proses *profile matching* secara garis besar merupakan proses membandingkan antara setiap kriteria setiap penilaian dalam sebuah proposal usulan penelitian yang diajukan sehingga diketahui perbedaan skornya (disebut juga *gap*), semakin kecil *gap* yang dihasilkan maka bobot nilainya semakin besar yang berarti memiliki peluang lebih besar untuk prioritas kelayakan/kelulusan.

GAP = Profil proposal – Profil ideal.....(2.1)

Langkah selanjutnya adalah menghitung nilai *core factor* dan *secondary factor*. *Core factor* merupakan kriteria penilaian yang paling utama harus terkandung dalam sebuah proposal penelitian. Perhitungan *core factor* menggunakan persamaan.

Tabel 2.1 Pembobotan Nilai GAP

No	Selisih	Bobot	Keterangan
1	0	5	Tidak ada selisih skor kriteria
2	1	4,5	Kriteria kelebihan 1 level
3	-1	4	Kriteria kekurangan 1 level
4	2	3,5	Kriteria kelebihan 2 level
5	-2	3	Kriteria kekurangan 2 level

6	3	2,5	Kriteria kelebihan 3 level
7	-3	2	Kriteria kekurangan 3 level

$$NCF = \frac{\sum NC\ (\text{kriteria})}{\sum IC} \quad \text{.....(2.2)}$$

Keterangan:

NCT : Nilai rata–rata *core factor*

NC : Jumlah total nilai *core factor*

IC : Jumlah *item core factor*

Sedangkan *secondary factor* merupakan *item-item* selain yang ada pada faktor utama *(core factor). Secondary factor* dihitung menggunakan persamaan.

$$NSF = \frac{\sum NS\ (\text{kriteria})}{\sum IC} \quad \text{.....(2.3)}$$

Keterangan:

NST : Nilai rata–rata *secondary factor*

NS : Jumlah total nilai *secondary factor*

IS : *Jumlah item secondary factor*

Selanjutnya perhitungan nilai total berdasar nilai dari *core* dan *secondary factor* yang digunakan sebagai kriteria penilaian yang berpengaruh terhadap kelulusan proposal penelitian. Perhitungan dapat dilakukan menggunakan persamaan.

$$N(Tot\_kriteria) = \ (x)\%NCF + (x)\%NSF \quad \text{.....(2.4)}$$

Keterangan:

NCT : Nilai rata–rata *core factor*

NST : Nilai rata–rata *secondary factor*

NT *:* Nilai total kriteria penilaian

Langkah terakhir adalah perhitungan ranking, yang dilakukan dengan menggunakan persamaan

$Ranking = (x)\%N1 + (x)\%N2 + (x)\%Nn$.....(2.5)

Keterangan:

N1, N2, Nn : Nilai total per kriteria

(x)% : Persentase nilai kriteria

2.7 *My Structure Query Language (MySQL)*

MySQL merupakan bahasa pemrograman *open-source* yang paling popular dan banyak digunakan dilingkungan Linux. Kepopuleran ini karena ditunjang oleh performansi *query* dari *database*-nya yang jarang bermasalah. (Allen dan Hornberger, 2002: p220). MySQL adalah sebuah program pembuatan *database* yang bersifat *open source*, artinya siapa saja dapat menggunakannya secara bebas. (Nugroho, 2004: p29)

MySQL dikembangkan oleh MySQL AB, sebuah perusahaan komersial yang membangun layanan bisnisnya melalui *database* MySQL. Awal mula pengembangan MySQL adalah pengguna mSQL untuk koneksi ke tabel mempergunakan rutin level rendah. Setelah beberapa pengujian diperoleh kesimpulan mSQL tidak cukup cepat dan fleksibel untuk memenuhi kebutuhan. Sehingga dihasilkan suatu antarmuka SQL baru pada *database* tetapi dengan API yang mirip mSQL. API ini dipilih sedemikian sehingga memudahkan *porting* kode.

2.8 *Visual Basic.NET*

Visual basic terkenal sebagai bahasa pemrograman yang mudah digunakan untuk membuat aplikasi yang berjalan diatas *platform* Windows. Pada tahun 90-an, Visual Basic menjadi bahasa pemrograman yang paling populer dan menjadi pilihan utama untuk mengembangkan program berbasis Windows. Versi Visual Basic yang terakhir sebelum berjalan diatas .NET Framework adalah VB6 (Kurniawan, 2011).

Visual Basic .NET dirilis pada bulan Februari tahun 2002 bersamaan dengan *platform* .NET 1.0. Kini sudah ada beberapa versi dari Visual Basic yang berjalan pada *platform* .NET, yaitu VB 2002 (VB7), VB 2005 (VB8), VB 2008 (VB9), dan VB 2010 (VB10) yang dirilis bersamaan dengan Visual Studio 2010. Selain Visual Basic 2010, Visual Studio 2010 juga mendukung beberapa bahasa lain yaitu C#, C++, F# (bahasa baru untuk *functional programming*), IronPhyton, dan IronRuby (bahasa baru untuk *dynamic programming*) (Kurniawan, 2011).

2.9 *System Development Life Cycle (SDLC)*

Software Development Life Cycle (SDLC) merupakan suatu proses pengembangan atau perubahan pada suatu perangkat lunak (IEEE Computer Society, 2004). Pengembangan atau perubahan tersebut dilakukan dengan cara menggunakan model-model dan metodologi yang digunakan oleh banyak orang yang telah mengembangkan sistem-sistem perangkat lunak sebelumnya. Hal tersebut berdasarkan *best practice* atau cara-cara yang sudah teruji baik.

2.10 Tahapan SDLC

2.10.1 Analisis dan Desain Perangkat Lunak

Analisis perangkat lunak digunakan untuk mengidentifikasi dan mengevaluasi permasalahan yang terjadi dan kebutuhan yang diharapkan, sehingga dapat diusulkan sebuah perbaikan.

Desain perangkat lunak adalah aktivitas siklus hidup dimana kebutuhan perangkat lunak dianalisis untuk menghasilkan struktur deskripsi internal perangkat lunak yang berfungsi sebagai dasar untuk konstruksi (IEEE Computer Society, 2004). Sebuah desain perangkat lunak menggambarkan arsitektur perangkat lunak, yaitu bagaimana perangkat lunak disusun dalam komponen dan antarmuka antara komponen-komponen tertentu.

A. *System Flowchart*

   System Flowchart merupakan bagan yang menunjukkan alur kerja secara keseluruhan dari sistem yang menunjukkan urutan-urutan dari prosedur yang ada dalam sistem dan menunjukkan apa yang dikerjakan sistem.

B. *Data Flow Diagram*

   Data flow diagram merupakan detail dari *system flowchart* yang digunakan untuk menggambarkan arus data didalam sistem secara terstruktur dan jelas.

C. *Conceptual Data Model*

   Conceptual data model digunakan untuk menggambarkan kebutuhan basis data secara detail dalam bentuk *logic*.

D. *Physical Data Model*

   Physical data model digunakan untuk menghitung perkiraan penyimpanan termasuk rincian alokasi penyimpanan khusus untuk sistem basis data tertentu.

E. Desain Basis Data *(Table)*

Desain basis data digunakan untuk merancang struktur basis data yang sesuai dengan kebutuhan pengguna.

F. Desain Antarmuka Sesuai Dengan Spesifikasi Yang Telah Ditetapkan

Tahapan desain antarmuka merupakan langkah untuk merancang tampilan aplikasi penilaian kinerja pegawai berdasarkan kompetensi.

G. Desain *Programming (Pseudocode)*

Pada tahapan desain *programming* merupakan tahapan yang menjelaskan dimana alur kerja program mulai dirancang untuk mempermudah dalam melakukan pengkodean program

2.10.2 Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak merupakan piranti yang harus dipamerkan untuk memecahkan beberapa masalah yang ada (IEEE Computer Society, 2004). Kebutuhan perangkat lunak bertujuan untuk mengotomatisasi bagian dari tugas seseorang untuk mendukung proses bisnis dari sebuah organisasi, memperbaiki kekurangan dari perangkat lunak yang ada, mengontrol perangkat dari beberapa masalah yang dapat dijadikan solusi perangkat lunak. Terdapat empat tahapan dalam kebutuhan perangkat lunak yaitu sebagai berikut:

1. Elisitasi

Tahapan elisitasi merupakan tahapan awal dalam membangun pemahaman tentang perangkat lunak yang diperlukan untuk memecahkan masalah. Tahapan kegiatan elisitasi digunakan untuk mengidentifikasi darimana asal kebutuhan perangkat lunak dan bagaimana cara mendapatkannya.

2. Analisis

   Pada tahap analisis menjelaskan tentang konflik antar kebutuhan, menemukan batas-batas perangkat lunak dan bagaimana interaksi dengan lingkungan sekitar, dan menguraikan kebutuhan sistem untuk mendapatkan kebutuhan perangkat lunak.

3. Spesifikasi

   Tahap spesifikasi dilakukan pendokumentasian perencanaan perangkat lunak (solusi aplikasi), dan mengajukan solusi aplikasi kepada pihak *stake holder* terkait.

4. Validasi

   Tahapan validasi digunakan untuk memastikan bahwa kebutuhan perangkat lunak telah sesuai dengan kebutuhan pengguna

2.10.3 Konstruksi Perangkat Lunak

Tahapan konstruksi perangkat lunak digunakan untuk melakukan konversi hasil desain ke sistem informasi yang lengkap melalui tahapan pengkodean termasuk bagaimana membuat basis data dan menyiapkan prosedur pengujian, mempersiapkan file pengujian, kompilasi pengkodean, memperbaiki dan membersihkan program serta melakukan peninjauan pengujian. (IEEE Computer Society, 2004).

2.10.4 Uji Coba Perangkat Lunak

Uji coba perangkat lunak terdiri dari verifikasi dinamis yang menyediakan perilaku sebuah perangkat lunak yang diwakili oleh beberapa contoh kasus uji coba (IEEE Computer Society, 2004). Kasus uji coba tersebut dilakukan dengan

memberikan masukan kepada perangkat lunak agar muncul reaksi sesuai yang diharapkan, dan sebaliknya.

Dalam melakukan uji coba perangkat lunak, yang pertama kali diperhatikan adalah fundamental dari uji coba perangkat lunak yang menjelaskan tentang *terminology* dari uji coba terkait, kunci masalah dari uji coba, dan hubungan uji coba tersebut dengan aktifitas lainnya didalam perangkat lunak. Kedua, yang perlu diperhatikan adalah tingkatan dari uji coba yang didalamnya menjelaskan tentang target dari uji coba dan tujuan dari uji coba tersebut. Ketiga, perlu diperhatikan dalam teknik dari uji coba yang meliputi uji coba berdasarkan intuisi dan pengalaman tester, diikuti oleh teknik berdasarkan spesifikasi, teknik berdasarkan kode, teknik berdasarkan kesalahaan, teknik berdasarkan penggunaan, dan teknik berdasarkan *relative* ketergantungan dari aplikasi tersebut. Keempat, perlu diperhatikan bahwa pengukuran dikelompokkan menjadi dua yaitu berhubungan dengan evaluasi ketika uji coba dilakukan serta ketika uji coba telah selesai dilakukan. Kelima, perlu diperhatikan bahwa proses uji coba itu sendiri yang berisi tentang pertimbangan praktis dan aktifitas uji coba.