BAB II

LANDASAN TEORI

2.1 Sistem Informasi

2.1.1 Sistem

Sistem dapat didefinisikan sebagai kumpulan komponen yang saling terkait yang bekerjasama untuk mencapai tujuan bersama. Fungsi sistem adalah untuk menerima masukan menjadi *output* (Bocij, 2008).

2.1.2 Sistem Informasi

Data adalah fakta-fakta atau kejadian-kejadian yang dapat berupa angka-angka atau kode-kode tertentu. Data masih belum mempunyai arti bagi penggunanya. Untuk dapat mempunyai arti data diolah sedemikian rupa sehingga dapat digunakan oleh penggunanya. Hasil pengolahan data inilah yang disebut sebagai informasi. Secara ringkas, informasi adalah data yang telah diolah dan mempunyai arti bagi penggunanya. Sehingga sistem informasi dapat didefinisikan sebagai prosedur-prosedur yang digunakan untuk mengolah data sehingga dapat digunakan oleh penggunanya (Herlambang dan Tanuwijaya, 2005).

2.2 Analisis dan Perancangan Sistem

2.2.1 Analisis Sistem

Menurut Whitten (2004), analisis sistem adalah sebuah teknik pemecahan masalah yang memecahkan sebuah sistem menjadi komponen-komponen untuk tujuan pembelajaran bagaimana komponen-komponen tersebut bekerja dan berinteraksi untuk mencapai tujuannnya.

Menurut Jogiyanto (2003), di dalam tahap analisis sistem terdapat langkah-langkah dasar yang harus dilakukan oleh analisis sistem, yaitu:

a. *Identify* yaitu mengidentifikasikan masalah.
b. *Understand* yaitu memahami kerja dari sistem yang ada.
c. *Analyze* yaitu menganalisis sistem.
d. *Report* yaitu membuat laporan hasil analisis.

2.2.2 Perancangan Sistem

Menurut Whitten (2004), perancangan sistem adalah teknik komplementer pemecahan masalah (yang bekerjasama dengan sistem analisis) yang menyususn kembali komponen-komponen sebuah sistem kembali ke sistem yang utuh dengan harapan menghasilkan sistem yang lebih baik. Teknik ini dapat melibatkan penjumlahan, penghapusan dan perubahan komponen-komponen terhadap sistem sebelumnya.

Rancangan sistem adalah proses identifikasi dari proses-proses dan data yang diperlukan oleh sistem baru. Jika sistem yang dirancang adalah sistem berbasis komputer, perancagnan dapat menyertakan spesifikasi jenis peralatan yang digunakan (McLeod, 2001).

Rancangan sistem terdiri dari dua kelompok, yaitu:

a. Sistem konseptual

   Perancangan dibuat berdasarkan kebutuhan *user* dan dibuat kerangka kerja untuk penerapannya.

b. Sistem fisik

Perancangan dibuat berdasarkan rancangan, kemudian dibuat spesifikasi secara terperinci, yang nantinya dapat dipergunakan untuk pembuatan dan *testing* program.

2.3 Evaluasi

Evaluasi adalah kegiatan untuk mengumpulkan informasi tentang bekerjanya sesuatu, yang selanjutnya informasi tersebut digunakan untuk menentukan alternatif yang tepat dalam mengambil sebuah keputusan. Fungsi utama evaluasi dalam hal ini adalah menyediakan informasi-informasi yang berguna bagi pihak *decision maker* untuk menentukan kebijakan yang akan diambil berdasarkan evaluasi yang telah dilakukan (Arikunto dan Cepi, 2008:2).

Evaluasi perlu dilakukan untuk segala jenis dan ukuran organisasi, maka evaluasi seharusnya mampu merangsang sikap kritis terhadap ekspektasi dan asumsi yang ada, sehingga dapat memicu pengkajian ulang atas tujuan dan nilai-nilai, serta mampu memberikan alternatif dan merumuskan kriteria.

Kegiatan evaluasi dilakukan secara berkala berdasarkan rekapitulasi pencatatan harian operasional alat HMC periode bulan. Evaluasi dilakukan secara berkala tidak hanya pada triwulan atau setelah munculnya permasalahan. Dengan melakukan evaluasi yang berkala atau berkelanjutan memungkinkan penolak ukuran kemajuan dan pemonitoran yang lebih efektif (David, 2009).

2.4 PT Berlian Jasa Terminal Indonesia – Divisi Teknik

PT Berlian Jasa Terminal Indonesia (PT BJTI) merupakan anak perusahaan dari PT Pelabuhan Indonesia III (Persero). PT BJTI Sejak tahun 2002 dipercaya mengelola Terminal Berlian Tanjung Perak Surabaya dan Terminal Peti Kemas di Tenau sejak awal tahun 2012. Sebagai operator pelabuhan selama satu dekade, PT BJTI telah banyak dipercaya oleh berbagai perusahaan Indonesia maupun mancanegara dalam pengelolaan peti kemas internasional, terminal peti kemas domestik, terminal curah kering, layanan intermoda, dan berbagai jasa bongkar muat penunjang lainnya.

PT BJTI melalui Divisi Teknik bertanggung jawab untuk melakukan penyusunan, perencanaan, koordinasi, pelaksanaan, pengendalian dan monitoring di bidang pemeliharaan dan perbaikan peralatan bongkar muat mekanik dan non-mekanik seperti *Harbour Mobile Crane (HMC), Rubber Tyred Gantry (RTG), Top Leader (TL), Forklift, Hopper* dan utilisasi alat. Divisi Teknik PT BJTI sebagai pihak yang bertanggung jawab terhadap operasional alat, melakukan pengawasan dan kontrol operasional alat secara kontinyu, dari operasional alat hingga *maintenance* alat.

2.5 Harbour Mobile Crane

Harbour Mobile Crane (HMC) adalah jenis *shore crane* atau derek penopang yang dirancang khusus untuk keperluan pelayanan bongkar muat di dermaga. Sistem *gantry* menggunakan roda ban karet (*wheel*) untuk memudahkannya dalam *manouver*, untuk keperluan pelayanan bongkar muat tersebut dibutuhkan peralatan tambahan seperti *Spreader* untuk penanganan

bongkar muat peti kemas, *Grab* untuk penanganan *bulk*, dan lain - lain. Saat ini, PT BJTI memiliki sepuluh buah HMC yang beroperasi di pelabuhan Tanjung Perak.

Gambar 2.1. Harbour Mobile Crane (HMC)

2.6 Monthly Report

Monthly Report adalah sebuah pelaporan bulanan yang dibuat oleh PT BJTI yang berfungsi sebagai pelaporan dari keseluruhan aktivitas penggunaan alat khususnya alat HMC, mulai dari performa hingga laporan *maintenance* alat, terdiri atas :

a) *Performance Report* :

Sebuah pernyataan rinci yang mengukur hasil dari beberapa aktivitas peralatan HMC dalam hal keberhasilan selama jangka waktu bulanan, terdiri atas:

a. *Total Hours per Month* (TH) (Jam) : Total beroperasinya alat HMC

b. *Preventive Maintenance* (PM) (Jam) : Pemeliharaan prefentif untuk mencegah dari kerusakan lanjutan.

c. *Total Outage Hours* (TO) (Jam) : Waktu total berhentinya mesin bekerja, Terdiri atas *Breakdown Reliability* (BD Re), *Breakdown Availability* (BD Av), dan *Accident* (A)

d. *Operating Hours (OP)* (Jam)*:* Kondisi dioperasikannya alat, terdiri atas *Plug 1* (OP1) = 08.00 – 19.59 dan *Plug 2* (OP2) = 20.00 – 07.59

e. *Stand by* (S) (Jam) : Waktu mesin dalam kondisi *standby*

f. *Utilization Time* (UT) (%) : Seberapa banyak sebuah sistem menghasilkan produk atau jasa dibandingkan dengan kemampuan sistem tersebut. Untuk mendapatkan nilai *Utilization Time*, dapat dihitung dengan menggunakan rumus berikut :

UT = (TH – TO – S)/TH ..... (2.1)

g. *Reliability Time* (RT) (%) : Ukuran kehandalan dari sistem /alat dalam melakukan performa dengan perbandingan dari kondisi yang telah ditentukan dalam kurun waktu tertentu. Untuk mendapatkan nilai *Reliability Time*, dapat dihitung dengan menggunakan rumus berikut :

RT = (OP I + OP II) – BD Re / (OP I + OP II)..... (2.2)

h. *Avaliability Time* (AT) (%) : Merupakan ketersediaan mesin untuk selalu menyala, sehingga *availability* yang tinggi berarti mempunyai nilai minimal *downtime.* Untuk mendapatkan nilai *Availability Time*, dapat dihitung dengan menggunakan rumus berikut :

AT = (TH-BD Av)/TH ..... (2.3)

Kemudian dilakukan penghitungan Total dan *Average* jumlah hari per bulan dari pencatatan *performance report* alat HMC tersebut. Standart operasional alat HMC terdapat pada *availability time* dan *reliability time* ada pada angka 90,00.

b) *Performance Summary* :

Sebuah pernyataan rinci yang mengukur hasil keseluruhan aktivitas peralatan HMC dalam hal keberhasilan selama jangka waktu bulanan, terdiri atas :

a. *Equipment Type and Number* (ET & EN) : Nama alat dan penomoran alat
b. *Total Hour* (TH) : Jumlah jam per bulan.
c. *Engine Hour per Meter* (EHRM) : Total perputaran mesin dalam satuan jam.
d. *Metered Operating Hours* (OP) :Total jam beroperasinya alat HMC
e. *Preventive Maintenance* (PM) : Pemeliharaan perfentif untuk mencegah kerusakan lanjutan pada dalam satuan jam.
f. *Breakdown Reliability* (BD – Re) : Total waktu berhentinya alat pada kondisi stand by pada dalam satuan jam.
g. *Breakdown Availability* (BD – Av) : Total waktu berhentinya alat pada kondisi operasi pada dalam satuan jam.
h. *Stand by* (S) : Indikator alat sedang dalam posisi *standby* dalam satuan jam.
i. *Number of Breakdown* (NB) : Total *breakdown availability* dan *breakdown reliability*. untuk mendapatkan nilai NB didapat dengan rumus :

$$NB = BD\ -Av + BD\ –Re \quad \text{..... (2.4)}$$

j. *Fuel in Liters* (FU) : Total penggunaan *fuel* per alat.

k. *Total Outage Hours* (TO) : Waktu total berhentinya mesin bekerja,

l. *Mean Time Between Failure* (MTBF) : Ukuran waktu rata-rata dari suatu alat hingga alat tersebut mengalami kerusakan dalam satuan persen.

MTBF = (*OP / NB) * 100%* ..... (2.5)

m. Liter per EHRM : Konsumsi *Fuel* dalam satuan liter per EHRM pada tiap alat HMC. untuk mendapatkan nilai Liter per EHRM didapat dengan rumus :

Liter per EHRM = *FU / EHRM* ..... (2.6)

c) Daily Report

Pelaporan harian penggunaan alat. Jenis aktivitas alat yaitu :

a. *Operation* (O) : Menunjukkan alat sedang beroperasi

b. *Maintenance* (M) : menunjukkan alat sedang dalam perbaikan

c. *Standby* (S) : Indikator alat sedang dalam posisi *standby*

d. *Breakdown* (B) : Indikator alat sedang dalam kondisi ada gangguan/kerusakan

e. *Accident Repair* (A) : indikator terdapat kecelakaan kerja

f. *Etcetera* (E) : Keterangan indikasi kejadian lain-lain

g. Pada kejadian (B), (M), (R), (A) diberikan keterangan apabila terjadi suatu kejadian.

h. EHRM atau perputaran mesin dicatat pada jam 08.00

Terdiri dari 3 *shift* operator : operator *shift* 1 bekerja pada jam 08.00-16.00. *shift* 2 pada jam 16.01-24.00, *shift* 3 pada jam 00.01-08.00. Peletakan alat HMC terdapat pada dua terminal, yaitu terminal berlian dan jamrud.

d) EHRM *And Box*

a. *Engine Hour Meter* (EHRM) : dikenal sebagai pengukur perputaran mesin, adalah perangkat yang merekam penggunaan mesin, umumnya untuk tujuan merekam banyaknya perputaran penggunaan mesin HMC dan digunakan sebagai parameter pemeliharaan alat.

b. Box : Total box yang diangkut

e) Fuel Consumption And EHRM For Fill Up Fuel Engine

Berfungsi sebagai pencatatan konsumsi bahan bakar dan posisi EHRM pada saat bahan bakar diisikan. Terdiri atas posisi EHRM tepat pada saat bahan bakar diisikan, bahan bakar yang diisikan adalah sebanyak 5000 Liter.

f) Oil Consumption

Pencatatan penggantian oli pada alat HMC. Jenis oli yang terdapat pada penggunaan alat yaitu *engine oil, hydraulic oil, dan gear box*.

g) Spare Part Used

Pencatatan terhadap penggantian *spare parts* yang digunakan.

h) Breakdown Report

Pencatatatan terhadap proses *breakdown* yang terjadi

i) Accident Report

Pencatatan terhadap *accident* yang terjadi.

2.7 *Software Engineering Body of Knowledge* (SWEBOK)

Menurut John Wiley England & Sons (2004) menuliskan : "SWEBOK menggambarkan pengetahuan secara umum tentang rekayasa perangkat lunak yang dibagi ke dalam 10 area pengetahuan (*Knowledge Areas*) atau disebut KA's." *Software Engineering Body of Knowledge* (SWEBOK) adalah produk dari komite koordinasi rekayasa perangkat lunak disponsori oleh IEEE *Computer Society*. SWEBOK sendiri mempunyai panduan yang disebut *Guide to the* SWEBOK, panduan ini dibuat untuk 5 tujuan, yaitu :

a. Untuk memperlihatkan kesamaan pandangan tentang rekayasa perangkat lunak di seluruh dunia.
b. Untuk memperjelas tempat dan menetapkan batas dari rekayasa perangkat lunak dan hubungannya dengan disiplin ilmu lain seperti ilmu komputer, manajemen proyek, teknik komputer dan matematika.
c. Untuk memberi karakter isi dari disiplin ilmu rekayasa perangkat lunak.
d. Untuk memberikan akses topik ke SWEBOK
e. Untuk memberikan pengetahuan dasar bagi pengembangan kurikulum dan sertifikasi serta perizinan.

Berikut adalah penjabaran tentang ruang lingkup pengetahuan atau yang disebut juga *Knowledge Area* (KA's) yang digunakan sebagai panduan dalam mengembangkan aplikasi.

2.7.1 *Requirements*

Tahapan awal dalam membangun aplikasi, *Software Requirements* merupakan sebuah properti yang disajikan untuk memenuhi kebutuhan dalam menyelesaikan permasalahan yang ada akan diselesaikan oleh aplikasi tersebut.

Menjabarkan bagaimana mengotomatiskan sebuah permasalahan sebuah tugas yang dihadapi oleh pengguna, membantu menganalisis proses bisnis perusahaan yang telah menggunakan aplikasi, menganalisis kekurangan yang ada, dan lainnya. Berikut penjabaran tentang beberapa tahapan yang ada pada *software requirement* :

a. Requirement Elicitation

Tahapan awal dalam pemenuhan *software requirements* makna dari kebutuhan mendatang ini berhubungan dengan darimana kebutuhan perangkat lunak itu sendiri dan bagaiman para pengembangan perangkat lunak dapat mengumpulkannya. Pada dasarnya, kegiatan yang dijabarkan adalah dari tiap individu dan tiap pemegang kendali sistem tersebut untuk membangun kesinambungan antara pihak pengembang dan pengguna perangkat lunak itu sendiri.

b. Requirement Analysis

Tahapan ini membahas tentang kegiatan menganalisis kebutuhan untuk :

1. Mendeteksi dan menyelesaikan ketidakcocokan yang ada pada tiap-tiap kebutuhan.
2. Menggali batasan yang ada pada perangkat lunak yang dikembangkan dan bagaimana perangkat lunak tersebut akan berinteraksi dengan sistem.
3. Menguraikan kebutuhan sistem yang akan digunakan sebagai kebutuhan perangkat lunak

c. Requirements spesification

Secara teknis pada kata "*specification*" mengacu pada banyaknya jumlah pekerjaan atau kemampuan perangkat lunak tersebut dalam mencapai

tujuannya. Dalam sebuah istilah pengembangan perangkat lunak. "*software requirements specification*" secara khusus mengarah kepada hasil ketepatan, atau penyamaan elektronik, yang dapat ditinjau, dinilai, dan dibenarkan.

d. Requirement Verification and Validation

Beberapa dokumen *requirements* di atas menjadi bahan dari tahapan validasi dan verifikasi. Kebutuhan yang ada divalidasi untuk menjamin bahwa pengembang dari perangkat lunak tersebut dapat memahami kebutuhan yang akan dicapai. Penyesuaian kebutuhan untuk standar perusahaan sangat penting untuk diperhatikan bahwa kebutuhan tersebut dimengerti, konsisten, dan lengkap.

2.7.2 Analisis

Tahap Analisis merupakan tahap identifikasi, seleksi, dan perencanaan sistem yang bertujuan untuk mendeteksi dan memberikan solusi antar kebutuhan serta mengetahui ruang lingkup perangkat lunak dan bagaimana perangkat lunak tersebut berinteraksi dengan lingkungan.

Tahapan analisis kebutuhan, menunjukkan tahapan-tahapan di dalam analisis kebutuhan. Pada dasarnya, aktivitas analisis dibutuhkan dalam setiap proses dalam daur hidup pengembangan perangkat lunak. Dalam proses rekayasa kebutuhan, analisis pun dilakukan dalam setiap aktivitas-aktivitasnya. Aktivitas tersebut antara lain sebagai berikut :

1. *Domain Understanding* : Dalam tahapan ini, pengembang harus mengetahui bagaimana organisasi perusahaan beroperasi dan apa yang menjadi permasalahan pada sistem yang berjalan.

2. *Requirements Collection* : Tahapan ini merupakan tahapan pengumpulan kebutuhan akan sistem yang akan dibangun sehingga diperlukan adanya interaksi secara intensif dengan *stakeholder*.
3. *Classification* : Tahapan ini mengelompokkan hasil dari tahap kebutuhan sehingga menjadi lebih terstruktur untuk selanjutnya diorganisir ke dalam kelompok-kelompok yang koheren.
4. *Conflict Resolution* : Tahapan ini berguna untuk menemukan dan menyelesaikan kebutuhan yang di dalamnya terdapat konflik. Konflik tersebut dapat terjadi antara dua *stakeholder* yang saling terkait tetapi memiliki fasilitas yang tidak sesuai, atau dapat terjadi antara kebutuhan dan sumber daya.
5. *Prioritisation* : Tahap ini melakukan interaksi dengan *stakeholder* untuk mengidentifikasikan kebutuhan-kebutuhan prioritas dari masing-masing kebutuhan agar memenuhi sumber daya yang tersedia pada organisasi.
6. *Requirements Checking*: Menganalisis sekumpulan kebutuhan dari hasil tahapan sebelumnya untuk menverifikasi dan memvalidasi berdasarkan aspek kelengkapan, konsistensi, dan kebutuhan nyata.

Semua jenis kebutuhan yang telah diperoleh tersebut kemudian dituangkan dalam bentuk dokumen yang berisi tentang kebutuhan sistem secara keseluruhan. Dokumen ini menjelaskan secara rinci tentang kesepakatan antara pengembang dengan klien, desain perangkat lunak yang akan dibangun, segala resiko yang akan dihadapi dan jadwal pembuatan perangkat lunak. Secara umum dokumen ini biasa disebut dengan *Software Requirements Spesification* (SRS).

Pada dokumen SRS akan dijelaskan juga mengenai kebutuhan fungsional dan non-fungsional dimana kebutuhan non-fungsional dibuat berdasarkan

dokumen *IEEE standart* 803:1993. *IEEE* 803:1993 mengelompokkan kebutuhan non-fungsional ke dalam sejumlah kategori kualitas dari suatu perangkat lunak. Kategori-kategori tersebut secara umum dibagi ke dalam 2 kelompok, yaitu faktor kualitas eksternal dari perangkat lunak dan faktor kualitas internal perangkat lunak. Faktor kualitas eksternal merupakan kategori kualitas yang dapat diobservasi atau menjadi ketertarikan utama dari pelanggan. Kategori-kategori yang termasuk di dalam kelompok ini antara lain :

a. Ketepatan (*correctness*),
b. *Robustness*,
c. Unjuk Kerja (*performance*),
d. Ketersediaan dan kualitas antar muka (*interface*),
e. Kehandalan (*reliability*), dan
f. Ketersediaan (*availability*)

Sedangkan kualitas faktor internal merupakan kategori kualitas yang dapat diobservasi atau menjadi ketertarikan utama dari pengembang. Seprerti :

a. Kemudahan membaca/memahami struktur perangkat lunak (*readibility*),
b. Kemampuan untuk dilakukan pengujian (*testability*),
c. Ketersediaan dan kualitas dokumentasi (*documentation*),
d. Kemudahan pemeliharaan (*maintainability*), dan
e. Adaptasi terhadap lingkungan berbeda (*portability*)

2.7.3 Desain

Merupakan tahap perancangan pemodelan data yang divisualisasikan melalui *Entity Relationship Diagram* (ERD), Conceptual *Data Model* (CDM), dan *Physical Data Model* (PDM); dan pemodelan proses yang dapat divisualisasikan

melalui *Data Flow Diagram* (DFD) atau melalui *Unified Modelling Language* (UML). Software Design memiliki peran penting dalam pengembangan perangkat lunak, hasil dari analisis kebutuhan menjadi kebutuhan yang sudah lengkap untuk memenuhi fungsi-fungsi yang dibutuhkan. Desain tersebut mencakup desain form dan laporan, desain antarmuka dan dialog, desain basis data dan file (*framework*), dan desain proses atau desain struktur proses. (England, dkk, 2004 : 51-55)

1. Flowchart

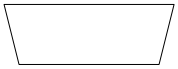
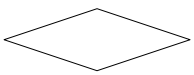
a. Flow Direction Symbol

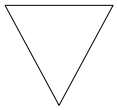
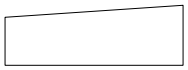
Tabel 2.1 *Flow Direction Symbols*

	Simbol arus / *flow*, yaitu menyatakan jalannya arus suatu proses
	Simbol *connector*, berfungsi menyatakan sambungan dari proses ke proses lainnya dalam halaman yang sama.
	Simbol *off-page connector*, menyatakan sambungan dari proses ke proses lainnya dalam halaman yang berbeda.

b. Processing Symbols

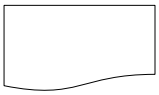
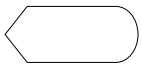
Tabel 2.2 *Processing Symbols*

	Simbol *process*, yaitu menyatakan suatu tindakan (proses) yang dilakukan oleh computer
	Simbol *manual*, yaitu menyatakan suatu tindakan (proses) yang tidak dilakukan oleh computer
	Simbol *decision*, yaitu menunjukkan suatu kondisi tertentu yang akan menghasikan dua

	kemungkinan jawaban : ya / tidak.
	Simbol *preparation*, yaitu menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk memberi harga awal
	Simbol *terminal*, yaitu menyatakan permulaan atau akhir suatu program.
	Simol *offline-storage*, menunjukkan bahwa data dalam simbol ini akan disimpan ke suatu media tertentu
	Simbol *manual-input*, memasukkan data secara manual dengan menggunakan online keyboard

c. *Input / Output Symbol*

Tabel 2.3 *Input / Output Symbol*

	Simbol *input-output* menyatakan proses *input* atau *output* tanpa tergantung jenis peralatannya
	Simbol *storage* menyatakan input berasal dari disk atau *output* disimpan ke disk
	Simbol *document* mencetak keluaran dalam bentuk dokumn (melalui printer)
	Simbol *display* mencetak keluaran dalam layar monitor.

2. *Data Flow Diagram*

DFD adalah diagram yang menggunakan notasi-notasi ini untuk menggambarkan arus dari data sistem, sekarang dikenal dengan nama diagram arus data (*data flow diagram*). DFD sering digunakan untuk menggambarkan

suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika tanpa mempertibangkan lingkungan fisik dimana data tersebut mengalir.

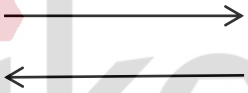
a. *External entity*

External entity merupakan kesatuan di lingkungan luar sistem yang dapat berupa orang, organisasi, atau sistem lainnya yang berada di lingkungan luarnya yang akan memberikan input atau menerima output dari sistem.

Gambar 2.2 Simbol Eksternal Entity

b. *Data flow*

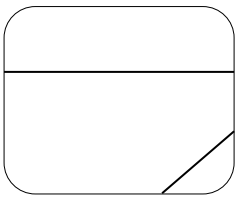
Data flow menunjukkan arus dari data yang berupa masukan untuk sistem atau hasil dari proses sistem dan dapat berbentuk sebagai berikut ini.



Gambar 2.3 Simbol Data flow

c. *Process*

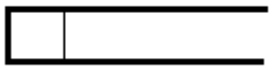
Process adalah kegiatan atau kerja yang dilakukan oleh orang, mesin atau komputer dari hasil suatu arus data yang masuk ke dalam proses untuk dihasilkan arus data yang akan keluar dari proses.



Gambar 2.4 Simbol Process

d. Data Store

Data store adalah simpanan dari data yang berupa, suatu *file database* di sistem komputer, arsip atau catatan manual, dan suatu tabel acuan manual.



Gambar 2.5 Simbol Data Store

3. Entity Relationship Diagram

Atribute adalah kolom di sebuah relasi. (Marlinda, 2004:28)

Macam-macam *atribute* yaitu :

a. *Simple Atribute*

Atribute ini merupakan *atribute* yang unik dan tidak dimiliki oleh *atribute* lainnya, misalnya *entity* mahasiswa yang *atribute*-nya NIM.

b. *Composite Atribute*

Composite Atribute adalah *atribute* yang memiliki dua nilai harga, misalnya nama besar (nama keluarga) dan nama kecil (nama asli).

c. *Single Value Atribute*

Atribute yang hanya memiliki satu nilai harga, misalnya *entity* mahasiswa dengan *atribute*-nya umur (tanggal lahir).

d. *Multi Value Atribute*

Atribute yang banyak memiliki nilai harga, misalnya *entity* mahasiswa dengan *atribute*-nya pendidikan (SD, SMP, SMA).

e. *Null Value Atribute*

Atribute yang tidak memiliki nilai harga, misalnya *entity* tukang becak dengan *atribute*-nya pendidikan (tanpa memiliki ijazah).

ERD ini diperlukan agar dapat menggambarkan hubugan antar *entity* dengan jelas, dapat menggambarkan batasan jumlah *entity* dan partisipasi antar *entity*, mudah dimengerti pemakai dan mudah disajikan oleh perancang *database*. (Kadir, 2008:46)

Untuk itu ERD dibagi menjadi 2 jenis model, yaitu :

a. Conceptual Data Model (CDM)

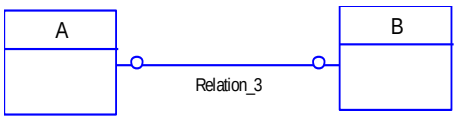
Merupakan jenis model data yang menggambarkan hubungan antar tabel secara konseptual.

b. Physical Data Model (PDM)

Merupakan jenis model data yang menggambarkan hubungan antar tabel secara fisikal.

ERD mempunyai 4 jenis hubungan antara lain :

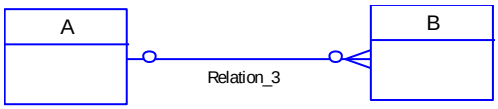
a. Hubungan *one–to–one* (1:1) menyatakan bahwa setiap entitas pada tipe entitas A paling banyak berpasangan dengan satu entitas pada tipe entitas B. Begitu pula sebaliknya. Contoh :



Gambar 2.6 Hubungan *one-to-one*

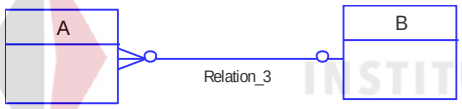
b. Hubungan *one–to–many* (1:M) menyatakan bahwa setiap entitas pada tipe entitas A bisa berpasangan dengan banyak entitas pada tipe entitas B,

sedangkan setiap entitas pada B hanya bisa berpasangan dengan satu entitas pada tipe entitas B. Contoh :



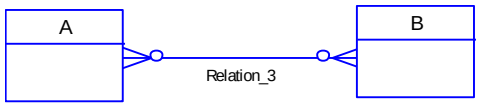
Gambar 2.7 Hubungan *one-to-many*

c. Hubungan *many–to–one* (M:1) menyatakan bahwa setiap entitas pada tipe entitas A paling banyak berpasangan dengan satu entitas pada tipe entitas B dan setiap entitas pada tipe entitas B bisa berpasangan dengan banyak entitas pada tipe entitas A. Contoh :



Gambar 2.8 Hubungan *many-to-one*

d. Hubungan *many–to–many* (M:N) Menyatakan bahwa setiap entitas pada suatu tipe entitas A bisa berpasangan dengan banyak entitas pada tipe entitas B dan begitu pula sebaliknya. Contoh :



Gambar 2.9 Hubungan *many-to-many*

e. Kardinalitas menggambar hubungan antara dua entitas dengan mengindentifikasi berapa banyak *instance* untuk setiap entitas yang nantinya dapat dihubungkan dengan setiap *instance* yang spesifik di entitas yang lain.

2.7.4 *Construction*

Software construction lebih diartikan sebagai pembuatan detail dari suatu pekerjaan, menciptakan satu software yang penting yang dikombinasikan dengan *code*, proses verifikasi, *testing unit*, dan *testing* yang terintegrasi, serta proses *debuging*. *Software construction* lebih sering dihubungkan dengan proses desain dan proses *testing*. Hal ini dikarenakan proses tersebut saling ketergantungan satu sama lain, dimana *software construction* merupakan keluaran dari desain *software* dan juga sebagai masukan dari *software testing*. *Software construction* bertipikal memproduksi volume konfigurasi item yang lebih tinggi dan juga dibutuhkan dalam mengelola sebuah *software* proyek (file sumber, isi, *test cases*, dll). (England, John Wiley & Sons, 2004 : 65-67)

1. *Software Contsruction Fundamentals*

Pada tahap pertama, dilakukan pendefinisian dasar tetang prinsip-prinsip yang digunakan dalam proses implementasi seperti minimalisasi kompleksitas, mengantisipasi perubahan, dan standar yang digunakan.

2. *Managing Costruction*

Bagian ini mendefeinisikan tentang model implementasi yang digunakan, rencana implementasi, dan ukuran pencapaian dari implementasi tersebut.

3. *Practical Considerations*

Bagian ini membahas tentang desain implementasi yang digunakan, bahasa pemrograman yang digunakan, kualitas dari implementasi yang dilakukan, proses pengetesan dan integritas.

Dalam proses pengimplementasian ini, digunakan beberapa aplikasi pendukung yaitu :

1) Codeigniter

Menurut Luqmanul Hakim (2010), *Codeigniter* adalah sebuah *framework* PHP yang dapat membantu mempercepat *developer* dalam pengembangan aplikasi web berbasis PHP dibandingkan dengan menulis semua kode program dari awal. *Codeigniter* menyediakan banyak *library* untuk mengerjakan tugas-tugas yang umumnya ada pada sebuah aplikasi yang dibuat menjadi semakin teratur dan rapi.

2) Bahasa Pemrograman PHP

*Hypertext Preprocessor (*PHP) adalah skrip bersifat *server-side* yang ditambahkan ke dalam *HyperText Markup Language* (HTML). Skrip ini akan membuat suatu aplikasi dapat diintegrasikan ke dalam HTML sehingga suatu halaman web tidak lagi bersifat statis, namun menjadi bersifat dinamis. Sifat *server-side* berarti pengerjaan skrip akan dilakukan di server, baru kemudian hasilnya dikirim ke *browser* (Kurniawan, 2002).

Keunggulan dari sifatnya yang *server-side* tersebut antara lain :

a.) Tidak diperlukan kompatibilitas *browser* atau harus menggunakan browser tertentu, karena serverlah yang akan mengerjakan skrip PHP. Hasil yang dikirimkan kembali ke browser umumnya bersifat teks atau gambar saja sehingga pasti dikenal oleh *browser* apa pun.

b.) Dapat memanfaatkan sumber-sumber aplikasi yang dimiliki oleh server, misalnya koneksi ke *database.*

c.) Skrip tidak dapat "diintip" dengan menggunakan fasilitas *view* HTML *sourcode*.

Kelebihan PHP dapat melakukan semua aplikasi program *Common Gateway Interface* (CGI), seperti mengambil nilai *form*, menghasilkan halaman web yang dinamis, serta mengirim dan menerima *cookie*. PHP juga dapat berkomunikasi dengan layanan-layanan yang mengunakan protokol IMAP, SNMP, NNTP POP3, HTTP, dan lain-lain. Namun kelebihan yang paling signifikan adalah kemampuannya untuk dapat melakukan koneksi yang baik dengan berbagai macam *database*.

3) *Database* MySQL

Database MySQL adalah jenis *database* yang sangat populer dan digunakan pada banyak *website* di internet sebagai bank data, selain itu *Database* MySQL juga dapat dijalankan dibeberapa *platform*, antara lain linux, windows, dan sebagainya.(Madcoms, 2011 : 215).

2.7.5 *Testing* dan Implementasi

Tahap ini mendemonstrasikan sistem perangkat lunak yang telah selesai dibuat untuk dijalankan, apakah telah sesuai dengan kebutuhan yang telah dispesifikasikan dan dapat diadaptasi pada lingkungan sistem yang baru (England, John Wiley & sons, 2004 : 73-74). Terdapat 5 tahapan, yaitu :

a) *Test Plan*

Membuat *Software Testing Fundamentals* yang berisi tentang penjelasan penting mengenai *terminology testing*

b) *Test Levels*

Merancang *Test Levels* yang terbagi antara target dan objektif dari pengetesan.

c) *Test Techniques*

Penjabaran terhadap teknik yang digunakan termasuk dasar-dasar pengetesan berdasarkan intuisi dan pengalaman serta teknik pengetesan secara teknik *coding*, teknik *kesalahan*, teknik penggunaan, dan teknik terkait lainnya.

d) *Test-Related Measures*

Merupakan ukuran-ukuran pencapaian *testing* yang telah dilakukan untuk kemudian dilakukan evaluasi kembali.

e) *Test Process*

Merupakan tahapan terakhir dari *Software Testing*, yaitu pendefinisian yang berisi tentang aktivitas *testing* yang dilakukan.

2.7.6 *Maintenance*

Pada tahap ini akan dilakukan pendeskripsian pekerjaan untuk mengoperasikan dan memelihara sistem informasi pada lingkungan pengguna termasuk implementasi akhir dan proses peninjauan kembali. Pemeliharaan sistem ini terdiri dari beberapa jenis yaitu:

a.) *Corrective*, yaitu memperbaiki desain dan *error* pada program.

b.) *Adaptive*, yaitu memodifikasi sistem untuk beradaptasi dengan perubahan lingkungan.

c.) *Perfective*, yaitu melibatkan sistem untuk menyelesaikan masalah baru atau mengambil kesempatan untuk penambahan fitur.

d.) *Preventive*, yaitu menjaga sistem dari kemungkinan masalah di masa yang akan datang.

Prosedur pemeliharaan tersebut disusun dalam beberapa tahapan.Tahap awal adalah menyusun *software maintenance fundamentals* yang berisi tentang dasar-dasar pemeliharaan, segala yang dibutuhkan untuk melakukan pemeliharaan, dan ketgori pemeliharaan. Selanjutnya adalah mendefinisikan *Key Issues in Software Maintenance*, yang berisi tentang teknik pemeliharaan, manajemen pemeliharaan dan biaya, serta ukuran pemeliharaan perangkat lunak. Tahap selanjutnya adalah mendefinisikan proses dan aktivitas pemeliharaan tersebut ke dalam *Maintenance Process*.(England, John Wiley & Sons, 2004 : 90-91).