BAB II

LANDASAN TEORI

2.1 Aplikasi

Menurut Jogiyanto (2005), aplikasi merupakan program yang berisikan perintah-perintah untuk melakukan pengolahan data. Jogiyanto menambahkan aplikasi secara umum adalah suatu proses dari cara manual yang ditransformasikan ke komputer dengan membuat sistem atau program agar data diolah lebih berdaya guna secara optimal.

Sedangkan menurut Dhanta (2009), aplikasi adalah *software* yang dibuat oleh suatu perusahaan komputer untuk mengerjakan tugas-tugas tertentu. Dari pengertian diatas, dapat disimpulkan bahwa aplikasi merupakan *software* yang ditransformasikan ke komputer yang berisikan perintah-perintah yang berfungsi untuk melakukan berbagai bentuk pekerjaan atau tugas-tugas tertentu seperti penerapan, penggunaan dan penambahan data.

2.2 Penjualan

Definisi penjualan menurut Mulyadi (2008), penjualan merupakan kegiatan yang dilakukan oleh penjual dalam menjual barang atau jasa dengan harapan akan memperoleh laba dari adanya transaksi-transaksi tersebut dan penjualan dapat diartikan sebagai pengalihan atau pemindahan hak kepemilikan atas barang atau jasa dari pihak penjual ke pembeli. Kegiatan penjualan terdiri dari transaksi penjualan barang dan jasa dikelompokan menjadi dua, yaitu penjualan tunai dan penjualan kredit.

Menurut Mulyadi (2008), Penjualan Tunai adalah penjualan yang dilakukan oleh perusahaan dengan mewajibkan pembeli melakukan pembayaran harga produk terlebih dahulu sebelum produk diserahkan oleh perusahaan kepada pembeli. Setelah uang diterima oleh perusahaan, maka produk kemudian diserahkan kepada pembeli dan transaksi penjualan tunai kemudian dicatat oleh perusahaan. Sedangkan penjualan kredit terdapat tenggang waktu antara penyerahan produk dalam penerimaan pembayaran harga produk. Dalam penjualan kredit, pada saat penyerahan produk, penjualan menerima tanda bukti penerimaan produk.

2.3 Piutang

Piutang juga merupakan komponen aktiva lancar yang penting dalam aktivitas ekonomi suatu perusahaan karena merupakan aktiva lancar perusahaan yang paling besar setelah kas. Piutang timbul karena adanya penjualan barang atau jasa secara kredit, bisa juga melalui pemberian pinjaman. Adanya piutang menunjukkan terjadinya penjualan kredit yang dilakukan perusahaan sebagai salah satu upaya perusahaan dalam menarik minat beli konsumen untuk memenangkan persaingan.

Adapun beberapa definisi piutang menurut para pakar adalah sebagai berikut:

1. Menurut Harjito dan Martono (2007), piutang dagang*(account receivable)* merupakan tagihan perusahaan kepada pelanggan atau pembeli atau pihak lain yang membeli produk perusahaan.

2. Menurut Horne (2005) mengatakan piutang meliputi jumlah uang yang dipinjam dari perusahaan oleh pelanggan yang telah membeli barang atau memakai jasa secara kredit.

Pengertian piutang secara umum adalah tuntutan atau klaim antara pihak yang akan memperoleh pembayaran dengan pihak yang akan membayar kewajibannya, atau dapat disebutkan sebagai tuntutan kreditur kepada debitur yang pembayarannya biasanya dilakukan dengan uang. Pengelolaan piutang secara efisien sangat diperlukan karena akan berpengaruh langsung terhadap peningkatan pendapatan. Meningkatnya proporsi piutang dalam laporan keuangan perusahaan akan membuat piutang menjadi bagian yang harus ditangani secara seksama.

2.3.1 Pengendalian Piutang

Menurut Keiso dan Weygandt (1999), pengendalian piutang merupakan suatu cara yang dilakukan oleh perusahaan dalam mengantisipasi kemungkinan adanya piutang yang tak tertagih, sehingga dengan adanya pengendalian piutang dapat mengurangi kerugian yang ditimbulkan dari piutang tak tertagih tersebut. Piutang yang timbul dari transaksi penjualan atau penyerahan barang atau jasa kepada langganan pada umumnya merupakan sebagian besar dari modal kerja suatu perusahaan. Oleh karena itu pengendalian dan kebijakan di dalam pemberian kredit dan pengumpulan piutang merupakan salah satu faktor yang perlu mendapat perhatian serius dari manajemen.

Perputaran piutang harus dikendalikan dengan menyusun tabel umur piutang (*aging schedule of receivables*). Piutang merupakan unsur penting dalam neraca. Prosedur yang wajar dan cara pengamanan yang cukup terhadap piutang

penting bukan saja utk keberhasilan perusahaan, tetapi juga untuk memelihara hubungan yang memuaskan dengan pelanggan. Pengendalian piutang dimulai sebelum ada persetujuan untuk mengirimkan barang dagangan, sampai setelah penyiapan dan penerbitan faktur, dan berakhir dengan penagihan hasil penjualan. Prosedur pengendalian piutang berhubungan erat dengan pengendalian penerimaan kas disatu pihak, dan pengendalian persediaan dilain pihak, sehingga piutang merupakan mata rantai diantara keduanya. Ada 3 (Tiga) bidang pengendalian piutang:

1. Pemberian kredit dagangkebijakan kredit dan syarat penjualan harus tidak menghalangi penjualan kepada para pelanggan yang sehat keadaan keuangannya, dan juga tidak boleh menimbulkan kerugian yang besar karena adanya piutang sangsi yang berlebihan.
2. Penagihan (*Collections*)apabila telah diberikan kredit, harus dilakukan setiap usaha untuk memperoleh pembayaran yang sesuai dengan syarat penjualan dalam waktu yang wajar.
3. Penetapan dan penyelenggaraan pengendalian intern yang layak. Membuat suatu sistem pengendalian *intern* yang memadai untuk memastikan bahwa semua penyerahan barang sudah difakturkan, atau difakturkan sebagai mana mestinya kepada para pelanggan, dan bahwa penerimaan benar-benar masuk kedalam rekening perusahaan.

Mengelola arus kas masuk dan keluar adalah salah satu tugas pokok keuangan karena semua transaksi bisnis bermuara ke dalam kas. Manajer keuangan pada umumnya mengharapkan penjualan dapat dilakukan dengan tunai atau kredit dengan waktu yang sesingkat-singkatnya, agar arus kas masuk cepat.

Untuk mengelola keuangan perusahaan yang baik, manajer keuangan harus menyusun anggaran pengumpulan piutang yang akan digunakan untuk mengendalikan piutang. Makin panjang umur piutangnya, makin buruk kondisi perusahaan karena makin lama piutang tersebut menjadi uang tunai (kas).

2.3.2 Pedoman Piutang Perusahaan

Berdasarkan ketetapan peraturan dari pimpinan perusahaan CV. Berkat Alam Sejahtera limit piutang setiap pelanggan tidak sama dilihat dari kemampuan liquiditas, solvabilitas, dan rentabilitas. Adapun rumusan perhitungan limit piutang setiap pelanggan ditentukan dari rata-rata penjualan per bulan dikali 2. Jika terjadi peningkatan piutang melewati batas limit piutang dikarenakan kenaikan permintaan pelanggan maka akan diberikan toleransi oleh perusahaan tetapi apabila terjadi peningkatan jumlah piutang dikarenakan keterlambatan pembayaran piutang (piutang macet) maka akan dilakukan pendekatan atau negosiasi.

Pelanggan perusahaan dibagi menjadi 3 bagian yaitu pelanggan kecil, pelanggan menengah, dan pelanggan besar. Pelanggan kecil seperti tukang las, pengisian ban nitrogen, atau yang tidak memiliki badan hukum yang jelas maka perusahaan memberlakukan penjualan secara tunai untuk mengurangi resiko piutang macet. Sedangkan untuk pelanggan menengah dan besar yang memiliki badan hukum jelas maka diperbolehkan untuk melakukan pembelian dengan menggunakan cara kredit. Untuk pelanggan menengah dan besar penentuan jatuh tempo pembayaran piutang ada 2 cara menyesuaikan berdasarkan kesepakatan antara perusahaan dengan pelanggan yaitu jatuh tempo pembayaran piutang 1 bulan terhitung dari dibuatnya nota penjualan kepada pelanggan atau jatuh tempo

pembayaran piutang 1 bulan terhitung seluruh penjualan selama 1 bulan berjalan pada bulan berikutnya.

2.4 Pembayaran

Pembayaran adalah proses pertukaran mata uang atau nilai moneter untuk barang, jasa, atau informasi (Chan Kah Sing, 2004) Dapat disimpulkan bahwa pembayaran adalah perpindahan hakatas nilai antara pihak pembeli dan pihak penjual yang secara bersamaan terjadi pula perpindahan hak atas barang atau jasa secara berlawanan.

2.5 *Short Message Service* (SMS)

Para ahli telah mendiskusikan tentang kemungkinan mengirim pesan teks lewat telepon seluler sejak tahun 1980-an. Kemudian di awal tahun 1985, dalam sebuah diskusi yang dipimpin seorang ahli J.Audestad memutuskan untuk menjadikan layanan pengiriman pesan teks atau SMS ini menjadi salah satu fasilitas telepon seluler yang menggunakan system GSM.

Sejarah SMS muncul pada Desember 1992. SMS adalah teknologi yang mampu mengirim dan menerima pesan antara telepon selular. SMS pertama ini dikirimkan oleh seorang ahli bernama Neil Papwort kepada Richard Jarvis menggunakan komputer Pesan itu dikirim dari sebuah komputer ke sebuah telepon seluler dalam jaringan GSM milik operator seluer Vodafone di Inggris.

SMS (*Short Message Service*) merupakan layanan yang banyak diaplikasikan pada sistem komunikasi tanpa kabel (nirkabel), memungkinkan dilakukannya pengiriman pesan dalam bentuk alphanumeric antar terminal pelanggan atau antar terminal pelanggan dengan sistem eksternal. SMS berupa

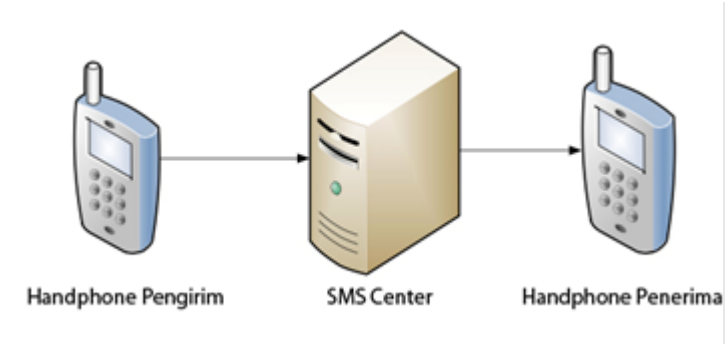
pesan teks, jumlah karakter pada setiap pengiriman bergantung pada operatornya. Operator selular di Indonesia umumnya membatasi 160 karakter untuk satu pengiriman dan penerimaan SMS. Selain itu SMS merupakan metode store dan forward sehingga keuntungan yang didapat adalah pada saat telepon selular penerima tidak dapat dijangkau, dalam arti tidak aktif atau diluar *service area*, penerima tetap dapat menerima SMS-nya apabila telepon selular tersebut sudah aktif kembali.

2.5.1 Cara Kerja SMS

Mekanisme cara kerja sistem SMS adalah melakukan pengiriman short message dari satu terminal pelanggan ke terminal yang lain. Hal ini dapat dilakukan karena adanya sebuah entitas dalam sistem SMS yang bernama *Short Message Service Center (SMSC)*, disebut juga *Message Center (MC)*. SMSC merupakan sebuah perangkat yang melakukan tugas *store and forward trafik short message*. Didalamnya termasuk penentuan atau pencarian rute tujuan akhir dari *short message*.

2.5.2 *Short Message Service Center* (SMSC)

Menurut (Gunawan, 2003), ”Pada saat mengirim SMS dari *handphone*, SMS tersebut tidak langsung dikirim pada handphone tujuan, akan tetapi dikirim terlebih dahule ke SMS Center (SMSC), lalu SMS tersebut diteruskan pada handphone tujuan”.



Gambar 2.1 Skema Cara Kerja SMS

Dengan adanya SMSC ini kita dapat mengetahui status dari pesan SMS yang telah dikirim, apakah telah sampai atau gagal diterima oleh handphone tujuan. Apabila handphone tujuan dalam keadaan aktif dapat menerima SMS yang dikirim, akan mengirimkan kembali pada konfirmasi ke SMSC yang menyatakan bahwa pesan telah diterima. Kemudian SMSC mengirimkan kembali status tersebut pada pengirim. Jika handphone tujuan dalam keadaan tidak aktif, SMS yang dikirim akan disimpan pada SMSC sampai period-validity terpenuhi.

2.5.3 Koneksi SMSC

Ada beberapa cara untuk melakukan koneksi ke SMSC antara lain. Berikut penjelasannya (Gunawan, 2003):

1. Menggunakan terminal baik berupa GSM modem atau handphone, cara ini adalah yang paling mudah tetapi memiliki kekurangan antara lain jumlah pesan yang dikirim per menit sangat terbatas (6-10 pesan permenit). Untuk mengantisipasinya biasanya menggunakan lebih dari satu terminal.
2. Koneksi langsung ke SMSC, dengan melakukan koneksi langsung ke SMSC dapat mengirim SMS dalam jumlah banyak, dapat mencapai 600 SMS per menit bergantung pada kapasitas dari SMSC itu sendiri. Untuk melakukan koneksi langsung diperlukan *protocol* penghubung. Protocol yang umum

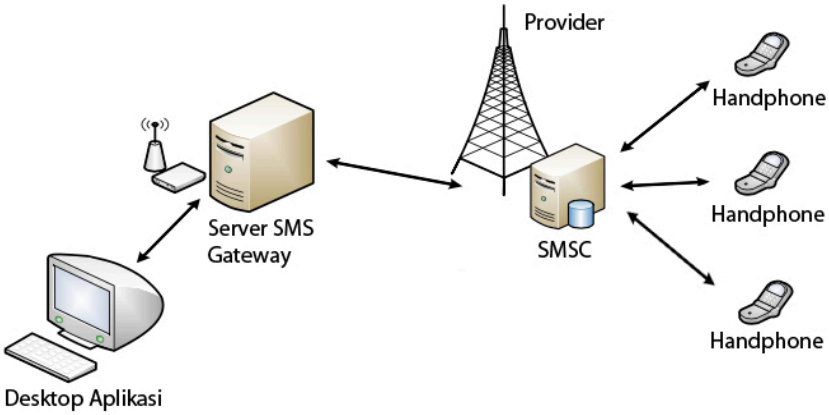
digunakan adalah UCP, SMPP, CIMD, OIS dan TAP. Masing-masing operator GSM menyediakan tipe protocol berbeda.

3. Menggunakan *software* bantu, saat ini banyak vendor telekomunikasi menawarkan *software* bantu untuk melakukan koneksi ke SMSC, dari yang bersifat f*reeware*, *open source* hingga komersial.

2.5.4 *Short Message Service Gateway* (SMS *Gateway*)

Menurut Gunawan (2003), SMS *Gateway* adalah teknologi mengirim, menerima dan bahkan mengolah SMS melalui komputer dan sistem komputerisasi biasanya digunakan pada aplikasi bisnis baik kepentingan promosi, penyebaran informasi pada pengguna. Seperti kita ketahui, pada jaman sekarang, hampir semua individu telah memiliki telepon selular *(handphone)*, bahkan ada individu yang memiliki lebih dari satu handphone. SMS merupakan salah satu fitur pada handphone yang pasti digunakan oleh pengguna *(user*), baik untuk mengirim, maupun untuk menerima SMS.

Bagi perusahaan, hal ini dimanfaatkan dengan baik dalam hal pemasaran dan pengumuman terhadap pelanggan *(customer)* perusahaan mereka. Data nomor handphone disimpan dalam *database* perusahaan dan ketika terdapat informasi atau layanan terbaru dapat memanfaatkan SMS *gateway* dalam proses informasinya (harga murah, cepat, dan mudah) dengan menggunakan sistem komputerisasi.



Gambar 2.2 Model Skema SMS *Gateway*

Aplikasi SMS *Gateway* akan mendeteksi *network* setiap *operator* yang akan digunakan dan setelah itu akan langsung diteruskan ke dalam aplikasi. Modem berfungsi untuk menjalankan aplikasi dan dapat menerima SMS yang dikirimkan oleh *operator* baik *request* dari pelanggan.

Segala *reguest* akan diproses oleh aplikasi dan akan diteruskan oleh SMS *Gateway* agar dapat diterima dan dilakukan pemprosesan data, dan *request* dari pelanggan akan dapat diterima dengan benar. Sistem SMS *Gateway* juga membutuhkan koneksi *database* agar *request* dari pelanggan dapat tersimpan. Database berfungsi menyimpan transaksi yang terjadi setiap harinya. Dan permintaan akan diproses dan dapat diketahui *request* terjadi dalam suatu transaksi.

Dari segi kecepatan SMS, semakin banyak terminal (*handphone* atau modem) yang terhubung ke komputer, maka semakin cepat proses pengiriman SMS. Selain dalam hal mengirim sms, dengan sistem komputerisasi, SMS*gateway* dapat melakukan *auto responder* atau *auto reply*, dimana dapat melakukan SMS kembali ke pelanggan yang *reply* ke sistem SMS *Gateway*.

2.6 *System Development Life Cycle* (SDLC)

System Development Life Cycle (SDLC) adalah suatu kerangka yangmenggambarkan kegiatan-kegiatan yang dilakukan pada setiap tahap pembuatansebuah *software* (Al Fatta, 2007). Terdapat banyak metode untuk mendeskipsikanSDLC ini, pada dasarnya setiap metode menggambarkan tahap-tahap sebagaiberikut.

A. Identifikasi, seleksi dan perencanaan

Tahap ini merupakan tahap *preliminary* dari pembuatan suatu *software*. Pada tahap ini, dikembangkan suatu rancang bangun dari suatu *software*.Langkah-langkah yang dilakukan dalam tahap ini antara lain:

1. Mengidentifikasi kebutuhan *user*.
2. Menyeleksi kebutuhan *user* dari proses identifikasi diatas, dengan menyesuaikan dengan kapasitas teknologi yang tersedia serta efisiensi.
3. Merencanakan sistem yang akan digunakan pada *software* yang dibuat, Dengan kebutuhan-kebutuhan sebagai berikut: kebutuhan fungsional dan non-fungsional, kebutuhan *user*, kebutuhan sistem, kebutuhan dokumen dan perangkat lunak.

B. Analisis sistem

Tahap ini merupakan tahap penyempurnaan, yang bertujuan memperoleh kebutuhan *software* dan *user* secara lebih spesifik dan rinci. Tujuan dilakukan tahap ini adalah untuk mengetahui posisi dan peranan teknologi informasi yang paling sesuai dengan kebutuhan perusahaan yang bersangkutan, serta mempelajari fungsi-fungsi manajemen dan aspek-aspek bisnis terkait

yangakan berpengaruh atau memiliki dampak tertentu terhadap proses desain, konstruksi dan implementasi *software*. Analisis sistem terbagi dua, yaitu:

1. Permodelan data, yang mencakup *Entity Relationship Diagram* (ERD), *Conceptual Data Model*(CDM), dan *Physical Data Model* (PDM).
2. Permodelan proses, dengan *Unified Modeling Language*.

C. Desain sistem

Setelah melakukan identifikasi serta analisis sistem, tahap selanjutnya adalah menerjemahkan konsep-konsep tersebut kedalam suatu sistem yang berwujud. Tahap ini meliputi pembuatan dan pengembangan sebagai berikut:

1. Desain form dan laporan (*reports*).
2. Desain antarmuka dan dialog (*message*).
3. Desain basis data dan *file*(*framework*).
4. Desain proses (*process structure*).

Pada tahap ini akan dihasilkan sebuah dokumen berupa *Software Architecture Document* (SAD). SAD ini adalah dokumen yang menjelaskan tentang arsitektur proyek perangkat lunakyang berhubungan dengan *project*.

D. Implementasi sistem

Tahap implementasi sistem ini diawali dengan pengetesan *software* yang telah dikembangkan. Beberapa tahap pengetesan adalah sebagai berikut:

1. *Developmental*,yakni pengetesan *error* per *module* oleh *programmer*.
2. *Alpha testing*, yakni *error testing* ketika *software* digabungkan dengan antarmuka *user*.

3. *Beta testing*, yakni pengetesan dengan lingkungan dan data yang sebenarnya.

Pada tahap berikutnya dilakukan konversi sistem, yaitu mengaplikasikan perangkat lunak pada lingkungan yang sebenarnya untuk digunakan oleh organisasi yang memesannya. Kemudian, dilakukan tahap dokumentasi, yaitu pencatatan informasi-informasi yang terkait dengan pembuatan sistem ini dan pelatihan, yaitu mengedukasi *end user* mengenai bagaimana cara menggunakan *software* yang bersangkutan. Pemberian pelatihan (*training)* harus diberikan kepada semua pihak yang terlibat sebelum tahap implementasi dimulai. Selain untuk mengurangi risiko kegagalan, pemberian pelatihan juga berguna untuk menanamkan rasa memiliki terhadap sistem baru yang akan diterapkan. Pada tahap ini akan dihasilkan sebuah dokumen berupa *Test Plan*. Dokumen *Test Plan* adalah sebuah dokumen yang digunakan memastikan dan memverifikasi antara rencana yang sudah dibuat dengan hasil yang dicapai, apakah sesuai dengan *planning* yang telah dibuat atau ada perubahan-perubahan dengan seiring pembuatan *software*.

E. Pemeliharaan sistem

Tahap pemeliharaan sistem adalah sebagai berikut:

1. *Korektif*, yaitu memperbaiki desain dan *error* pada program (*troubleshooting*).
2. *Adaptif*, yaitu memodifikasi sistem untuk beradaptasi dengan perubahan lingkungan.
3. *Perfektif*, yaitu melibatkan sistem untuk menyelesaikan masalah baru atau menambah fitur baru pada sistem yang telah ada.

Preventif, yaitu menjaga sistem dari kemungkinan masalah di masa yang akan datang.

2.7 Visual Studio.Net

Menurut Rahadian (2011), *Microsoft Visual Studio.NET* merupakan salah satu *software* buatan Microsoft Corp., yang didesain khusus dalam pembuatan program profesional berbasis *windows platform*. *Microsoft Visual Studio.NET* merupakan perangkat lunak yang terintegrasi, di dalamnya terdapat beberapa paket software yang dapat digunakan oleh programer dalam membangun sebuah program profesional, diantaranya adalah *Visual Basic*, *Visual J#*, *Visual C*, *#Visual C++* dan *Java Runtime* yang sama-sama berada dalam naungan platform *Microsoft .NET Framework*. Bagian dari software ini diantaranya *toolbox*, jendela properties, *server explorer* dan *solution explorer*.

2.8 Microsoft SQL Server

Menurut Marlinda (2004), *database* atau basis data adalah suatu susunan/kumpulan data operasional lengkap dari suatu organisasi/perusahaan yang diorganisir/dikelola dan disimpan secara terintegrasi dengan menggunakan metode tertentu menggunakan komputer sehingga mampu menyediakan informasi optimal yang diperlukan pemakainya. Penyusunan satu basis data digunakan untuk mengatasi masalah-masalah pada penyusunan data yaitu redundansi dan inkonsistensi data, kesulitan pengaksesan data, isolasi data untuk standarisasi, banyak pemakai(*multiple user*), masalah keamanan(*security*), masalah kesatuan(*integration*), dan masalah kebebasan data(*dataindependence*).

2.9 Desain

Menurut Wiyancoko (2000), desain adalah segala hal yang berhubungan dengan pembuatan konsep, analisis data, *project planning, drawing/rendering, cost calculation, prototyping, frame testing,* dan *test riding.*

Analis sistem dapat mendesain model dari sistem informasi yang diusulkan dalam bentuk *physical system* dan *logical model*. Bagan alir sistem (*system flowchart*) merupakan alat yang tepat digunakan untuk menggambarkan *physical system*. Bagan alir sistem adalah adalah suatu bagan dengan simbol-simbol tertentu yang menggambarkan urutan proses secara mendetail dan hubungan antara suatu proses atau instruksi dengan proses lainnya dalam suatu program.

Logical model dari sistem informasi lebih menjelaskan kepada *user* bagaimana nantinya fungsi-fungsi di sistem informasi secara logika akan bekerja. *Logical model* dapat digambarkan dengan menggunakan diagram arus data (*data flow diagram*). *Data flow diagram* (DFD) adalah suatu teknik yang digunakan untuk menjelaskan aliran data yang bergerak didalam sebuah sistem (Whitten, 2004).

2.10 *System Flow*

Flowchart adalah suatu metode untuk menggambarkan tahap-tahap pemecahan masalah dengan mempresentasikan simbol-simbol tertentu yang mudah dimengerti, mudah digunakan dan standar (Sutedjo, 2004). *System flowchart* ini tidak digunakan untuk menggambar urutan untuk memecahkan masalah, tetapi hanya untuk menggambarkan prosedur dalam sistem yang

dibentuk. Berikut ini adalah gambar dari simbol-simbol standar yang telah banyak digunakan dalam penggambaran *system flowchart.*

2.11 *Data Flow Diagram* (DFD)

Data flow diagram (DFD) adalah suatu teknik yang digunakan untuk menjelaskan aliran data yang bergerak di dalam sebuah sistem (Whitten, 2004). Untuk membuat DFD, sistem yang ada harus dipelajari terlebih dahulu termasuk aktivitas dan proses yang terjadi pada sistem tersebut. Dalam menganalisa aliran data, semua kejadian dievaluasi kedalam bentuk proses, tempat penyimpanan *(data store)* serta asal dan tujuan data berupa arus data.

DFD digambarkan secara bertingkat, mulai dari yang paling global *(context diagram)* sampai tingkat yang lebih rinci. Level tertinggi dari DFD disebut *context diagram*, yang bisa juga disebut dengan DFD Level 0. Context diagram, yang menunjukan lingkup dari satu aktifitas yang akan dimodelkan, hanya mempunyai satu simbol proses. Context diagram ini menggambarkan lingkup, batasan, dan lingkupan dari suatu aktifitas. Untuk menunjukan proses apa saja yang akan terjadi didalam suatu DFD digambarkan pada level berikutnya dengan cara merinci kembali proses yang ada dari proses pusat DFD Level 1. Demikian juga level 2 dan seterusnya, hingga pada akhirnya proses yang ada tidak bia lagi dibagi menjadi proses yang lebih kecil atau lebih spesifik.

Terdapat 4 simbol yang digunakan dalam pembuatan DFD menurut Kenneth E. Kendall dalam bukunya pada tahun 2008, yaitu:

1. *External Entity*

   Merupakan kesatuan luar *(entity)* dilingkungan luar sistem yang dapat berupa sekelompok orang, divisi, organisasi, atau sistem lainnya yang berada

dilingkungan luarnya yang akan memberikan input atau menerima output dari sistem. Suatu kesatuan luar dapat disimbolkan dengan suatu notasi kotak atau segi empat. Simbol *External Entity* dapat dilihat pada gambar di bawah ini.

Gambar 2.3*External Entity*

2. *Process*

Adalah kegiatan atau kerja yang dilakukan oleh orang, mesin atau komputer dari hasil suatu arus data yang masuk dalam proses untuk dihasilkan arus data yang akan keluar dari proses atau untuk mengubah input menjadi output. Suatu proses dapat ditunjukkan dengan simbol lingkaran. Simbol proses dapat dilihat pada gambar dibawah ini.

Gambar 2.4 *Process*

3. *Data Flow* (Aliran Data)

Data mengalir melalui sistem, dimulai dengan sebagian input dan diubah atau diproses menjadi output. Aliran data diberi simbol dengan suatu garis panah. Aliran data diberi nama yang jelas dan mengandung arti. Nama pada *Data Flow* ditulis disamping atas. Simbol*Data Flow* dapat dilihat pada gambar di bawah ini.

Gambar 2.5*Data Flow*

4. *Data Storage* (Penyimpanan Data)

Data disimpan untuk keperluan berikutnya. Simpanan data di DFD disimbolkan dengan sepasang garis horisontal paralel yang tertutup di salah

satu ujungnya. *DataStorage* adalah tempat menyimpan data secara manual maupun otomatis. Tempat penyimpanan dapat berupa filling cabinet, lemari, bahkan file komputer. Tujuan dari adanya Data Strorage yaitu mendokumentasikan data. Simbol*Data Storage* dapat dilihat pada gambar di bawah ini.

Gambar 2.6*Data Storage*

2.12 *Entity Relationship Diagram* (ERD)

Menurut Supriyanto (2005), *Entity Relationship Diagram* (ERD) merupakan suatu model untuk menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi. ERD untuk memodelkan struktur data dan hubungan antar data, untuk menggambarkannya digunakan beberapa notasi dan simbol.

Entity Relationship Diagram (ERD) merupakan teknik yang digunakan untuk memodelkan kebutuhan data dari suatu organisasi, biasanya oleh *System Analyst* dalam tahap analisis persyaratan proyek pengembangan *system*. Sementara seolah-olah teknik diagram atau alat peraga memberikan dasar untuk *desain database* relasional yang mendasari sistem *informasi* yang dikembangkan. ERD bersama-sama dengan detail pendukung, merupakan model data yang pada gilirannya digunakan sebagai spesifikasi untuk *database*.

2.12.1 Entitas*(Entity)*

Entity adalah objek yang dapat dibedakan dalam dunia nyata. *Entity* set adalah kumpulan dari *entity* yang sejenis. *Entity* set berupa:

1. Obyek secara fisik misalnya : Rumah, Kendaraan, Peralatan
2. Obyek secara konsep misalnya : Pekerjaan, Perusahaan, Rencana

Setiap entitas memiliki atribut-sifat tertentu yang menggambarkannya. Sebagai contoh, sebuah entitas karyawan dapat digambarkan dengan nama karyawan, umur, alamat, gaji, dan pekerjaan. Sebuah entitas tertentu akan memiliki nilai untuk masing-masing atributnya. Nilai atribut ini yang menggambarkan setiap entitas menjadi bagian utama dari data yang disimpan dalam *database*.

2.12.2 Atribut

Atributadalah karakteristik dari *entity* atau *relationship* yang menyediakan penjelasan detail tentang *entity relationship* tersebut. Atribut terdiri dari enam jenis, yaitu:

1. Atribut Komposit dapat dibagi menjadi subparts lebih kecil, yang merupakan atribut yang lebih mendasar dengan mandiri makna.
2. Atribut Atom adalah atribut sederhana atau atribut yang tidak dapat dibagi lagi ke dalam subparts yang lebih kecil.
3. Atribut Tunggal adalah atribut yang memiliki nilai tunggal untuk suatu entitas tertentu.
4. Atribut *Multivalued* adalah atribut yang memiliki jumlah nilai yang lebih dari satu.
5. Atribut *Derived* adalah nilai atribut dapat berasal dari entitas yang terkait.
6. Atribut*Stored* adalah nilai atribut yang dapat digunakan untuk menghasilkan nilai atribut lain.

2.12.3 *Key*

Jika suatu atribut dijadikan sebagai *key*, maka tidak boleh ada dua atau lebih baris data dengan nilai yang sama untuk atribut tersebut. Jadi atribut *key* harus bersifat unik.

Ada 3 macam key yang dapat diterapkan pada suatu tabel, yaitu:

a. *Superkey* merupakan satu atau lebih atribut (kumpulan atribut) yang dapat membedakan setiap basis data dalam sebuah tabel secara unik. Bahkan bisa ada lebih dari 1 kumpulan atribut yang bersifat seperti itu pada suatu tabel.
b. *CandidateKey* merupakan kumpulan atribut minimal yang dapat membedakan setiap baris data dalam sebuah tabel secara unuik. Sebuah atribut *CandidateKey* tidak boleh berisi atribut atau kumpulan atribut yang telah menjadi *Superkey*yang lain. Jadi sebuah *CandidateKey* pastilah *Superkey*, tapi belum tentu sebaliknya.
c. *PrimaryKey* merupakan atribut yang sangat unik yang dapat membedakan satu sama lain. Salah satu dari *Candidate Key* dapat dijadikan sebagai *Primary Key*.

2.12.4 Kardinalitas Atau Derajat Relasi

Relasi adalah hubungan yang terjadi antara satu atau lebih *entity*. *Relationship Set* adalah kumpulan relationship yang sejenis yang saling berhubungan. Kardinalitas relasi menunjukkan jumlah maksimum *entity* yang dapat berelasi dengan *entity* pada himpunan *entity* yang lain. Kardinalitas *entity*yang terjadiantara dua himpunan dapat berupa:

1. Relasi Satu ke Satu (*One to One*)

   Berarti setiap entitas pada suatu himpunan berhubungan paling banyak dengan satu entitas pada himpunan entitas lainnya, begitupun sebaliknya. Gambar relasi satu ke satu dapat dilihat pada gambar 2.7.

Gambar 2.7*One to one*

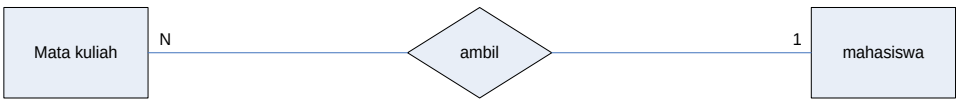
2. Satu ke Banyak (*One to Many*)

   Berarti setiap entitas pada suatu himpunan berhubungan dengan banyak entitas pada himpunan entitas lainnya, tetapi tidak sebaliknya. Gambar relasi satu ke banyak dapat dilihat pada gambar 2.8.

Gambar 2.8*One to many*

3. Banyak ke Satu (*Many to One*)

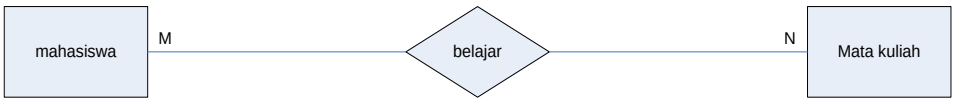
   Berarti setiap entitas pada suatu himpunan berhubungan dengan paling banyak satu entitas pada himpunan lainnya, tetapi tidak sebaliknya. Gambar relasi banyak ke satu dapat dilihat pada gambar 2.9.



Gambar 2.9*Many to one*

4. Banyak ke Banyak (*Many to Many*)

Berarti setiap entitas pada suatu himpunan dapat berhubungan dengan banyak entitas pada himpunan entitas lainnya, dan begitupunsebaliknya. Gambar relasi banyak ke banyak dapat dilihat pada gambar 2.10.



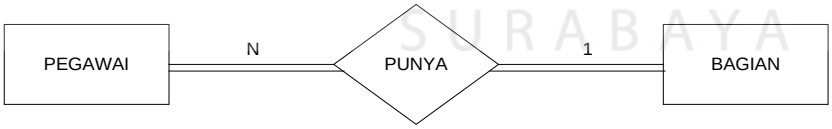
Gambar 2.10*Many to many*

2.12.5*Participation Constrainst*

Menjelaskan apakah keberadaan suatu*entity* tergantung pada hubungannya dengan *entity* yang lain. Terdapat dua macam *participation constrainst* yaitu:

1. *Total Participation*

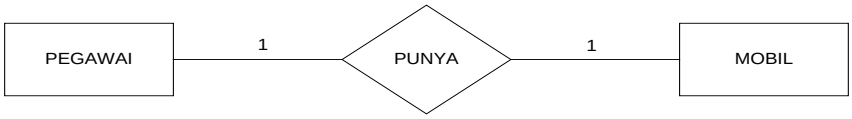
Keberadaan suatu *entity* tergantung pada hubungannya dengan *entity* yang lain. Gambar *Total Participation* dilihat pada gambar 2.11.



Gambar 2.11*Total Participation*

2. *Partial Participation*

Keberadaan suatu *entity* tidak tergantung pada hubungannya dengan *entity* yang lain. Gambar *Partial Participation* dilihat pada gambar 2.12.



Gambar 2.12 *Partial Participation*

2.13 *Construction*

Menurut England, John Wiley & Sons (2004). *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 di hubungkan 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 di butuhkan dalam mengelola sebuah *software* proyek (file sumber, isi, *test cases*, dll).

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 mplementasi yang dilakukan, proses pengetesan dan integritas.

Dalam proses pengimplementasian ini, digunakan beberapa aplikasi pendukung yaitu:

a. Bahasa Pemrograman Visual Basic .NET 2008

Microsoft Visual Basic merupakan sebuah bahasa pemrograman yang menawarkan *Integrated Development Environment* (IDE) visual untuk membuat program perangkat lunak berbasis sistem operasi Microsoft Windows dengan menggunakan model pemrograman. VB.NET 2008 digunakan untuk mengembangkan dan membangun aplikasi yang bergerak di atas sistem .NET Framework dengan menggunakan bahasa BASIC. Dengan menggunakan alat ini, para pengguna dapat membangun aplikasi Windows Forms, Aplikasi Website berbasis ASP.NET, dan juga aplikasi *command-line.*

b. *Database*SQL Server 2008

SQL Server adalah salah satu produk *Relational Database Management System* (RDBMS) yang populer saat ini. Fungsi utamanya adalah sebagai database server yang mengatur semua proses penyimpanan data dan transaksi suatu aplikasi. Versi SQL Server yang digunakan adalah SQL Server 2008.

2.14 Testing dan Implementasi

Menurut standart ANSI/IEEE 1059, testing adalah proses menganalisa suatu entitas software untuk mendeteksi perbedaan antara kondisi yang ada

dengan kondisi yang diinginkan *(defects, errors, bugs)* dan mengevaluasi fitur-fitur dari entitas software.

Secara umum, tujuan dari testing adalah untuk verifikasi, validasi dan deteksi error. Verifikasi dilakukan dengan melakukan pengujian semua proses-proses didalam sistem yang bertujuan untuk menguji sistem yang telah dibuat apakah dapat berjalan dengan baik. Validasi dilakukan untuk melihat kebenaran sistem apakah proses yang telah ditulis dalam spesifikasi adalah apa yang sebenarnya diinginkan atau dibutuhkan oleh pengguna. Deteksi error dilakukan untuk mengetahui apakah ada error yang terjadi atau memungkinkan akan terjadi apabila terjadi kesalahan pemakaian software oleh pengguna.

Macam atau tipe testing secara umum ada tiga macam berdasarkan waktu penggunaannya, yaitu:

1. *Unit Testing*

   Testing penulisan kode-kode program dalam satuan unit terkecil secara individual.

2. *System Testing*

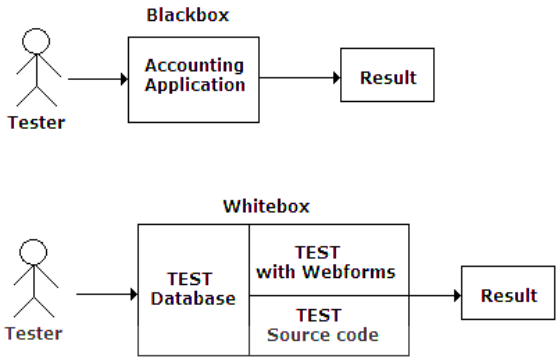
   Proses testing pada sistem terintegrasi untuk melakukan verifikasi bahwa sistem telah sesuai dengan spesifikasi.

3. *Acceptance Testing*

   Testing formal yang dilakukan untuk menentukan apakah sistem telah memenuhi kriteria penerimaan dan memberdayakan pelanggan untuk menentukan apakah sistem dapat diterima atau tidak.

Pengujian sistem untuk sistem ini menggunakan metode *black box*.*Black box testing* adalah strategi testing berbasis hanya pada spesifikasi dan kebutuhan.

Black box testing tidak membutuhkan pengetahuan dari jalur internal, struktur ataupun implementasi dari *software* yang akan diuji(Koirala & Sbeikh, 2008). Pada Gambar 2.13 akan dijelaskan bagaimana *white box*dan *black box testing* bekerja.



Gambar 2.13*Blackbox* dan *Whitebox testing*