BAB II

TINJAUAN PUSTAKA

2.1. Tinjauan Perusahaan

2.1.1. Sejarah Perusahaan

POLITEKNIK MANUFAKTUR BANDUNG (POLMAN Bandung) adalah Politeknik Negeri pertama di Indonesia yang dahulu bernama Politeknik Mekanik Swiss (PMS-ITB). POLMAN Bandung berdiri sejak tahun 1976 merupakan hasil kerjasama bilateral antara pemerintah RI dan pemerintah Konfederasi Swiss, yang dalam pelaksanaannya Pemerintah Indonesia diwakili oleh ITB dan Swisscontact mewakili Pemerintah Swiss yang berakhir pada tahun 1995.

POLMAN Bandung adalah politeknik pertama di Indonesia yang diresmikan oleh Menteri Pendidikan dan Kebudayaan Dr.Sjarif Thajeb pada tanggal 24 Maret 1977 meskipun angkatan mahasiswa pertama masuk tahun 1976.

POLMAN Bandung mendapatkan nilai akreditasi A untuk semua program studi dari BAN pada tahun 2002 diantaranya adalah:

1. Perancangan Manufaktur.
2. Pengecoran Logam
3. Otomasi Manufaktur
4. Teknik Manufaktur.

Setelah satu tahun melakukan persiapan, akhirnya POLMAN Bandung berhasil meraih sertifikat ISO 9001 - 2000 dari KEMA untuk :

1. Jasa Pendidikan Tinggi.
2. Perancangan dan Manufaktur Produk Cor Logam, Perkakas Presisi, Mesin Produksi dan Sistem Kendali.

VISI

Menjadi ujung tombak pendidikan dan penerapan teknologi manufaktur.

MISI

Menyiapkan lulusan yang kompeten dalam bidang teknologi manufaktur dan mampu bersaing dalam pasar global, dengan membangun dan mengembangkan pendidikan, rekayasa dan produksi.

2.1.2. Tempat dan Kedudukan Perusahaan

Politeknik Manufaktur Bandung (POLMAN) terletak di Jl.Kanayakan no.21 Dago Bandung.

2.1.3. Divisi / Bidang Pekerjaan Tempat Penelitian

Bidang pekerjaan tempat penelitian adalah di Jurusan Teknik Pengecoran Logam *(Foundry)*.

2.1.4. Struktur Organisasi

Struktur organisasi ada pada lampiran.

2.2. Landasan Teori

2.2.1. Konsep Dasar Sistem Informasi

2.2.1.1.Definisi Sistem Informasi

Telah diketahui bahwa informasi merupakan hal yang sangat penting bagi manajemen di dalam pengambilan keputusan. Pertanyaannya adalah darimana informasi tersebut bisa didapatkan?. Informasi dapat diperoleh dari sistem informasi (*information systems*) atau disebut juga dengan *processing systems* atau *information processing systems* atau *information-generating systems*. Sistem infromasi didefinisikan oleh Robert A. Leitch dan K. Roscoe Davis sebagai berikut :

Sistem informasi adalah suatu sistem didalam organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan.

2.2.1.2. Komponen Sistem Informasi

John Burch dan Gary Grudnitski mengemukakan bahwa sistem informasi terdiri dari komponen-komponen yang disebutnya dengan istilah blok bangunan (*building block*), yaitu blok masukan (*input block*), blok model (*model block*), blok keluaran (*output block*), blok teknologi (*technologi block*), blok basis data (*database block*), dan blok kendali (*komputers block*). Sebagai suatu sistem,

keenam blok tersebut masing-masing saling berinteraksi satu dengan yang lainnya membentuk satu kesatuan untuk mencapai sasarannya.

1. Blok Masukan

Input mewakili data yang masuk ke dalam sistem informasi. Input disini termasuk metode-metode dan media untuk menangkap data yang akan dimasukkan, yang dapat berupa dokumen-dokumen dasar.

2. Blok Model

Blok ini terdiri dari kombinasi prosedur, logika dan model matematik yang akan memanipulasi data input dan data yang tersimpan di basis data dengan cara yang sudah tertentu untuk menghasilkan keluaran yang diinginkan.

3. Blok Keluaran

Produk dari sistem informasi adalah keluaran yang merupakan informasi yang berkualitas dan dokumentasi yang berguna untuk semua tingkatan manajemen serta semua pemakai sistem.

4. Blok Teknologi

Teknologi merupakan “kotak alat” (*tool box*) dalam sistem informasi. Teknologi digunakan untuk menerima output, menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan keluaran dan membantu pengendalian dari sistem secara keseluruhan. Teknologi terdiri dari 3 bagian utama, yaitu teknisi (*humanware* atau *brainware*), perangkat lunak (*software*), dan perangkat keras (*hardware*). Teknisi dapat berupa orang-orang yang mengetahui teknologi dan membuatnya dapat beroperasi. Misalnya teknisi adalah operator

komputer, pemrogram, operator pengolah kata, spesialis telekomunikasi, analis sistem, penyimpanan data dan lain sebagainya.

5. Blok Basis Data

Basis data (*database*) merupakan kumpulan dari data yang saling berhubungan satu dengan yang lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya. Data perlu disimpan di dalam basis data untuk keperluan penyediaan informasi lebih lanjut. Data di dalam basis data perlu diorganisasikan sedemikian rupa, supaya informasi yang dihasilkan berkualitas. Organisasi basis data yang baik juga berguna untuk efisiensi kapasitas penyimpanannya. Basis data diakses atau dimanipulasi dengan menggunakan perangkat lunak paket yang disebut dengan DBMS (*Database Management Systems*).

6. Blok Kendali

Banyak hal yang dapat merusak sistem informasi, seperti misalnya bencana alam, api, temperatur, air, debu, kecurangan-kecurangan, kegagalan-kegagalan sistem itu sendiri, kesalahan-kesalahan, ketidak efisienan, sabotase dan lain sebagainya. Beberapa pengendalian perlu dirancang dan diterapkan untuk meyakinkan bahwa hal-hal yang dapat merusak sistem dapat dicegah ataupun bila terlanjur terjadi kesalahan-kesalahan dapat langsung cepat diatasi.

2.2.1.3. Arsitektur Informasi

Arsitektur Informasi adalah suatu pemetaan atau rencana kebutuhan-kebutuhan informasi di dalam organisasi. Arsitektur ini berguna sebagi penuntun bagi operasi sekarang atau menjadi *blue print* untuk arahan di masa mendatang.

Tujuan dari arsitektur ini adalah agar bagian teknologi informasi memenuhi kebutuhan-kebutuhan bisnis strategi organisasi oleh karena itu arsitektur informasi memadukan kebutuhan informasi, komponen sistem informasi, dan teknologi pendukung.

2.2.2. Rekayasa Perangkat Lunak

2.2.2.1.Evolusi Perangkat Lunak

Sebuah perangkat lunak yang disimpan atau digunakan dalam jangka waktu yang cukup lama akan menimbulkan masalah apabila sering diabaikan atau tidak diperhatikan perkembangannya. Permasalahan yang akan timbul dari penggunaan perangkat lunak yang cukup lama adalah :

1. Adanya kebutuhan baru. Sebagai contoh pada saat ini berkembangnya perangkat lunak berbasis *Web*.
2. Untuk mengurangi kompleksitas, biaya , waktu pemasaran. Sebagai contoh pemanfaatan dari bahasa perkembangan bahasa pemrograman tingkat tinggi.
3. Mengurangi cacat. Sebagai contoh dengan cara membuat standarisasi dari pengkodean yang tidak sesuai.

Untuk dapat menyelesaikan permasalahan yang timbul diatas, maka diperlukannya melakukan evolusi dari perangkat lunak tersebut.

Evolusi dari perangkat lunak itu sendiri meliputi :

1. Pembangunan Perangkat Lunak

   Yang termasuk dalam pembangunan perangkat lunak itu sendiri meliputi dari daur hidup perangkat lunak, yaitu : permintaan (*requirement*), spesifikasi, perancangan (*design*), testing dan sebagainya.

2. Perawatan Perangkat Lunak

   Merupakan proses-proses untuk memperpanjang waktu penggunaan sistem perangkat lunak yang ada, sehingga tetap dapat dipergunakan sebagaimana mestinya sesuai dengan baik.

3. Migrasi Perangkat Lunak

   Merupakan proses-proses yang memindahkan sistem yang ada ke sistem yang baru dikarenakan perkembangan dari kebutuhan perangkat lunak tersebut.

Untuk dapat menjaga kualitas dari perangkat lunak tersebut dalam melakukan evolusi perangkat lunak, maka dibuatlah *"Laws"* dari perangkat lunak tersebut. Adapun *"Laws"* dari perangkat lunak tersebut menurut Lehman adalah :

1. Perubahan yang terus menerus secara kontinyu.

   Program yang berada pada lingkungan dunia nyata harus mengalami perubahan atau program tersebut menjadi tidak berguna pada lingkungan tersebut.

2. Meningkatnya kompleksitas.

   Seiring dengan berkembangnya sebuah program maka program tersebut menjadi lebih kompleks. Kemudian penambahan sumber daya juga dibutuhkan untuk memelihara dan menyederhanakan struktur programnya.

   Ada sebuah pernyataan yang mengatakan :

   "Most often overlooked risk in software engineering: As the system grows over time, it will become too complex or disjointed to understand or make work reliably." Deutsch (1998).

3. Aturan yang fundamental dari sebuah program evolusi.

   Pada saat membuat sebuah program evolusi telah ditentukan ukuran, metriks serta indikatornya.

4. Tetap dijaga stabilitas dari organisasinya.

   Sebagai contoh dengan menambahkan sumber daya (misal manusia) tetapi tidak mengubah produktivitas.

5. Tetap *familiarity*.

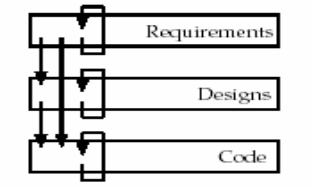
   Dimana pada saat membuat fungsi yang baru perbedaannya tidak jauh dari fungsi-fungsi yang sebelumnya.

Strategi Evolusi

Dalam melakukan evolusi perangkat lunak ada beberapa strategi yang dapat digunakan, diantaranya :

1. *Forward Engineering*

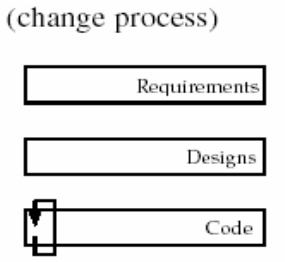
   Forward Engineering adalah sebuah proses pengubahan dari abstraksi level yang paling tinggi(*Requirement*) dan logik ke level design sampai ke level fisik (C*ode*)dari sistem.



Gambar 2.1 Proses *Forward Engineering*

2. *Restructuring*

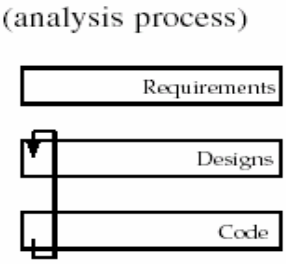
   Restructuring adalah merupakan proses perubahan perangkat lunak yang terjadi pada level phisik(*Code*).



Gambar 2.2 Proses *Restructuring*

3. *Redocumenting*

   Redocumenting adalah proses revisi terhadap dokumentasi system yang telah ada pada setiap level abstraksi.



Gambar 2.3 Proses *Redocumenting*

4. *Reverse Engineering*

   Reverse Engineering adalah proses untuk mengindentifikasi sistem yang bermula dari level abstraksi yang paling rendah (misal *object code*), untuk menghasilkan spesifikasi formal.

5. *Reengineering*

   Reengineering adalah proses untuk mengindentifikasi sistem yang bermula dari level abstraksi yang paling rendah (misal *object code*), untuk menghasilkan spesifikasi formal sehingga terbentuk *source code* baru.

6. *Roundtrip Engineering*

   Roundtrip Engineering merupakan proses untuk menjaga sinkronisasi antara *requirements*, *designs*, dan *code*.

7. *Retirement*

   Retirement adalah proses dimana sebuah perangkat lunak secara keseluruhan sudah tidak dipergunakan kembali (dipensiunkan).

Setiap strategi evolusi dapat dilakukan otomatisasi, dimana dapat dilakukan pada level fisik (*Code*). Proses tersebut terjadi pada bagian source code dengan mekanisme tertentu.

Perawatan Perangkat Lunak

Istilah perawatan perangkat lunak digunakan untuk menjabarkan aktivitas dari analis sistem (*software engineering*) yang terjadi pada saat hasil produk perangkat lunak sudah digunakan oleh pemakai (*user*).

Biasanya pengembangan produk perangkat lunak memerlukan waktu antara 1 sampai dengan 2 tahun, tetapi pada fase perawatan perangkat lunak menghabiskan 5 sampai dengan 10 tahun.

Aktivitas yang tejadi pada fase pemeliharaan antara lain :

1. Penambahan atau peningkatan atau perbaikan untuk produk perangkat lunak.
2. Penambahan fungsi-fungsi baru.
3. Perbaikan tampilan dan modus interaktif.
4. Perbaharui dokumen eksternal.
5. Perbaharui dokumen internal.
6. Perbaharui karakteristik perfomasi dari system.
7. Adaptasi produk dengan lingkungan mesin yang baru.
8. Pemindahan perangkat lunak ke sistem yang berlainan.
9. Modifikasi untuk dapat mempergunakan protokol atau disk drive tambahan.
10. Perbaikan permasalah yang timbul.
11. Pembenaran kesalahan yang timbul setelah produk perangkat lunak dipergunakan oleh *user* (pemakai).

Salah satu hal dari pelaksanaan evolusi perangkat lunak adalah perawatan. Perawatan perangkat lunak adalah melakukan modifikasi dari perangkat lunak yang telah jadi dan telah dikirimkan untuk memperbaiki kesalahan-kesalahan, untuk meningkatkan performansi atau hal lain yang berkaitan dengan perangkat lunak tersebut juga untuk melakukan adaptasi terhadap perubahan lingkungan dari penggunaan perangkat lunak tersebut. Hal tersebut berdasarkan diatas ANSI/IEEE

Std. 729-1983. Aktivitas yang dilakukan pada saat perawatan perangkat lunak tersebut adalah :

a. Perawatan yang dilakukan untuk penyesuaian (*Adaptive Maintenance*)

   Melakukan pengawasan terhadap sistem yang akan berubah, seperti melakukan pertemuan untuk membahas mengenai permintaan dari kebutuhan baru.

b. Perawatan yang dilakukan untuk perbaikan (*Corrective Maintenance*)

   Melakukan pengawasan setiap saat sehingga sistem berjalan sesuai dengan fungsinya, seperti dengan cara membuat laporan dari kesalahan yang timbul.

c. Perawatan yang dilakukan untuk penyempurnaan (*Perfective Maintenance*)

   Memperbaiki beberapa aspek agar sistem dapat meningkatkan kebutuhan yang diperlukan dimasa yang akan datang, seperti melakukan serangkaian tes.

d. Perawatan yang dilakukan untuk pencegahan (*Preventative Maintenance*)

   Melakukan perubahan sistem untuk menghindarkan kegagalan dimasa yang akan datang, seperti meningkatkan penanganan dari kesalahan.

Faktor-faktor yang ada dalam perawatan dibagi 2 yaitu :

1. Faktor Staf
   a. Menemukan orang dengan keterampilan yang tepat.
   b. Mengkoordinasi staf.
   c. Perancangan tidak lama dibutuhkan untuk memberikan masukan.
   d. Programmer *maintenance* terlihat sebagai warga kelas dua.
   e. Kemampuan komprehensif.
   f. Prioritas bisnis dan manajemen mungkin diatas bidang teknis.

2. Faktor Teknik

   a. Perawatan pengkodean dapat menambah kesalahan.

   b. Memperkecil dokumentasi.

   c. Sistem baru terlalu rumit bagi perangkat keras.

   d. Model pemrograman.

   e. Waktu yang terbatas untuk pengujian.

   f. *Path* dinamis.

   g. Ukuran dan kerumitan perangkat lunak.

Untuk menghadapi masalah perawatan perangkat lunak membutuhkan pemikiran yang berbeda dibandingkan pengembang, misalnya menyelesaikan :

1. Hilangkan kode program.
2. Pencarian.

2.2.3. Model Proses Rekayasa Perangkat Lunak

Model proses untuk rekayasa perangkat lunak dipilih berdasarkan sifat aplikasi dan proyeknya, metode dan alat-alat bantu yang akan dipakai, dan kontrol serta penyampaian yang dibutuhkan. Perkembangan perangkat lunak bisa dianggap sebagai lingkaran pemecahan masalah dimana terdapat empat keadaan berbeda, yaitu status quo, definisi masalah, perkembangan teknis memecahkan masalah di keseluruhan aplikasi dari banyak aplikasi, dan integrasi pemecahan menyampaikan hasil kepada siapa yang membutuhkan pertama kali. Bermacam-macam model proses yang berbeda pada perangkat lunak sebagai berikut:

1. Model Sekuensial Linier atau *Waterfall*

   Sekuensial linier mengusulkan sebuah pendekatan kepada perkembangan perangkat lunak yang sistematik dan sekuensial yang mulai pada tingkat dan kemajuan sistem pada seluruh analisis, desain, kode, pengujian, dan pemeliharaan.

2. Model Prototipe

   Prototyping paradigma dimulai dengan pengumpulan kebutuhan. Pengembang dan pelanggan bertemu dan mendefinisikan obyektif keseluruhan dari perangkat lunak, mengidentifikasi segala kebutuhan yang diketahui, dan area garis besar dimana definisi lebih jauh merupakan keharusan kemudian dilakukan “perancangan kilat”. Perancangan kilat berfokus pada penyajian dari aspek-aspek perangkat lunak tersebut,yang akan nampak bagi pelanggan / pemakai (contohnya pendekatan input dan format output).

3. Model RAD

   Rapid Application Development (RAD) adalah sebuah model proses perkembangan perangkat lunak sekuensial linier yang menekankan siklus perkembangan yang sangat pendek. Model RAD ini merupakan sebuah adaptasi “kecepatan tinggi” dari model sekuensial linier di mana perkembangan cepat dicapai dengan menggunakan pendekatan konstruksi berbasis komponen.

4. Model Proses Perangkat Lunak Evolusioner

Model evolusioner adalah model iteratif. Model itu ditandai dengan tingkah laku yang memungkinkan perekayasa perangkat lunak mengembangkan versi perangkat lunak yang lebih lengkap sedikit demi sedikit.

a. Model Pertambahan

Model inkeremental menggabungkan elemen-elemen model sekuensial linier (diaplikasikan secara berulang) dengan filosofi prototipe iteratif. Model pertambahan memakai urutan-urutan linier di dalam model yang membingungkan, seiring dengan laju waktu kalender. Model pertambahan berfokus pada penyampaian produk operasional dalam setiap pertambahannya.

b. Model Spiral

Model spiral (*spiral model*) adalah model proses perangkat lunak yang evolusioner yang merangkai sifat iteratif dari prototipe dengan cara komputer dan aspek sistematis dari model sekuensial linier.

c. Model Rakitan Komponen

Model ratikan komponen menggabungkan beberapa karakteristik model spiral. Model ini bersifat evolusioner, sehingga membutuhkan pendekatan iteratif untuk mencapai perangkat lunak. Tetapi model rakitan komponen merangkai aplikasi dari komponen perangkat lunak sebelum dipaketkan (kadang-kadang disebut “kelas”).

d. Model Perkembangan Konkuren

Model proses yang konkuren dapat disajikan secara skematis sebagai sederetan aktivitas teknik mayor, tugas-tugas, dan keadaannya yang lain. Contohnya aktivitas rekayasa yang dibatasi untuk model spiral dipenuhi dengan melakukan tugas-tugas sebagai berikut: prototyping dan atau pemodelan analisis, spesifikasi kebutuhan, dan rancangan.

5. Model Formal

Model metode formal mencakup sekumpulan aktivitas yang membawa kepada spesifikasi matematis perangkat lunak komputer. Metode formal memungkinkan perekayasa perangkat lunak untuk mengkhususkan, mengembangkan, dan memverifikasi sistem berbasis komputer dengan menggunakan notasi matematis yang tepat.

2.2.4 Konsep Merancang Basis Data

2.2.4.1 Definisi

Istilah basis data banyak menimbulkan interpretasi yang berbeda. Anthoni J. Fabbri dan A. Robert Schwab, mendefinisikan basis data sebagai berikut :

"Basis data adalah sistem berkas terpadu yang dirancang terutama untuk meminimalkan pengulangan data"

Menurut George Tsu-der Chou, basis data dapat didefinisikan sebagai berikut :

"Basis data sebagai kumpulan informasi bermanfaat yang diorganisasikan ke dalam tatacara yang khusus".[3]

Basis data dimaksudkan untuk mengatasi problem pada sistem yang memakai pendekatan berbasis berkas.

Sistem basis data adalah suatu sistem menyusun dan mengelola record-record menggunakan komputer untuk menyimpan atau merekam serta memelihara data opersional lengkap sebuah organisasi/perusahaan sehingga mampu menyediakan informasi yang optimal yang diperlukan pemakai untuk proses mengambil keputusan.

Untuk mengelola basis data diperlukan perangkat lunak yang disebut DBMS. DBMS adalah perangkat lunak sistem yang memungkinkan para pemakai membuat, memelihara, mengontrol, dan mengakses basis data dengan cara yang praktis dan efisien.

2.2.4.2 Tahap Perancangan Basis Data

Perancangan basis data merupakan langkah untuk menentukan basis data yang diharapkan dapat mewakili seluruh kebutuhan pengguna. Perancangan basis data terdiri atas perancangan basis data secara konseptual, perancangan basis data secara logis, dan perancangan basis data secara fisis.[3]

Beberapa komponen yanng terdapat pada perancangan basis data secara konseptual antara lain:

1. Entitas

   Entitas terkadang disebut tipe entitas atau kelas entitas. Entitas adalah objek yang dapat dibedakan dari objek-objek lainnya.

2. Atribut

   Atribut adalah item data yang menjadi bagian dari suatu entitas. Istilah lain dari attribut adalah properti.

3. Hubungan

   Hubungan adalah asosiasi atau kaitan antara dua entitas.

4. Kekangan

   Kekangan digunakan untuk melindungi integritas data (misalnya, melindungi kesalahan sewaktu pengisian data).

5. Domain

   Domain adalah himpunan yang berlaku bagi suatu atribut. Kekangan domain mendefinisikan nama, tipe, format, panjang, dan nilai masing-masing item data.

6. Integritas Referensial

   Integritas referensial adalah aturan-aturan yang mengatur hubungan antara kunci primer dengan kunci tamu milik tabel-tabel yang berbeda dalam suatu basis data relasional untuk menjaga konsistensi data.

2.2.5 Metode Perancangan Sistem

2.2.5.1 Diagram Konteks

Diagram konteks menggambarkan hubungan antara sistem dengan entitas luarnya. Diagram konteks berfungsi sebagai transformasi dari satu proses yang melakukan transformasi data *input* menjadi data *output*. Entitas yang dimaksud adalah entitas yang mempunyai hubungan langsung dengan sistem.

Suatu diagram konteks selalu mengandung satu dan hanya satu proses saja. Diagram konteks ini menggambarkan hubungan *input* atau *output* antara sistem dengan dunia luarnya (kesatuan luar).

2.2.5.2 *Entity- Relationship Diagram*

ERD hanya berfokus pada data, dengan menunjukkan “jaringan data” yang ada untuk suatu sistem yang diberikan. ERD sangat berguna bagi aplikasi di mana data dan hubungan yang mengatur data sangatlah kompleks. ERD pada mulanya diusulkan oleh Peter Chen untuk desain sistem *database* relasional dan telah dikembangkan oleh yang lainnya. Serangkaian komponen utama diidentifikasikan untuk ERD : objek data, atribut, hubungan dan berbagai tipe indikator. Tujuan utama dari ERD adalah untuk mewakili objek data dan hubungan mereka.

Kardinalitas model data harus dapat merepresentsikan jumlah peristiwa dari objek di dalam hubungan yang diberikan. Tillmann mendefinisikan kardinalitas dari object-relationship pair dengan cara sebagai berikut: kardinalitas merupakan spesifikasi dari sejumlah peristiwa dari satu [objek] yang dapat dihubungkan ke sejumlah peristiwa dari [objek] yang lain. Dengan mempertimbangkan semua kombinasi dari ‘satu’ dan ‘banyak’, dua [objek] dapat dihubungkan sebagai:

1. Satu-ke-satu (1:1)
2. Satu-ke-banyak (1:N)
3. Banyak-ke-satu (N:1)
4. Banyak-ke-banyak (M:N)

2.2.5.3 *Data Flow Diagram*

Data Flow Diagram (DFD –DAD/Diagram Alir Data) memperlihatkan hubungan fungsional dari nilai yang dihitung oleh sistem, termasuk nilai masukan, nilai keluaran, serta tempat penyimpanan internal. DAD adalah gambaran grafis yang memperlihatkan aliran data dari sumbernya dalam objek kemudian melewati proses yang mentransformasinya ke tujuan yang lain, yang ada pada objek lain. DAD sering digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika tanpa mempertimbangan lingkungan fisik dimana data tersebut mengalir. DFD merupakan alat yang digunakan pada metodologi pengembangan sistem yang terstruktur (*structured analysis and design*). DFD merupakan alat yang cukup populer sekarang ini, karena dapat menggambarkan arus data di dalam sistem dengan terstruktur jelas.

Beberapa simbol yang digunakan dalam *Data Flow Diagram* (DFD) antara lain:

1. *External Entity* (kesatuan luar) atau *boundary* (batas sistem)

   Setiap sistem pasti mempunyai batas sistem (*boundary*) yang memisahkan suatu sistem dengan lingkungan luarnya. Sistem akan menerima input dan menghasilkan output kepada lingkungan luarnya. Kesatuan luar (*external entity*) merupakan kesatuan (*entity*) 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.

2. *Data Flow* (arus data)

   Arus data (*data flow*) di DFD diberi simbol suatu panah. Arus data ini mengalir diantara proses (*process*), simpanan data (*data strore*) dan kesatuan luar (*external entity*). Arus data ini menunjukkan arus dari data yang dapat berupa masukan untuk sistem atau hasil dari proses sistem.

3. *Process* (proses)

   Suatu proses adalah kegiatan atau kerja yang dilakukan oleh orang , mesin atau kompuiter dari hasil suatu arus data yang masuk ke dalam proses untuk dihasilkan arus data yang akan keluar dari proses. Untuk *physical data flow diagram* (PDFD), proses dapat dilakukan oleh orang, mesin atua komputer, sedangkan untuk *logical data flow diagram* (LDFD), suatu proses hanya menunjukkan proses dari komputer. Setiap proses harus diberi penjelasan yang lengkap meliputu identifikasi proses, nama proses dam pemroses.

4. *Data Store* (simpanan luar)

   Simpanan data (*data store*) merupakan simpanan dari data yang dapat berupa, yaitu suatu file atau *database* di sistem komputer, suatu arsip atau catatan manual, suatu kotak tempat data di meja seseorang, suatu tabel acuan manual, dan suatu agenda atau buku.

2.2.5.4 Kamus Data

Kamus data (KD) atau *data dictionary* (DD) atau disebut juga dengan istilah *system data dictionary* adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi. Dengan menggunakan kamus

data, analisis sistem dapat mendefinisikan data yang mengalir di sistem dengan lengkap. Kamus data dibuat pada tahap analisis sistem dan digunakan baik pada tahap analisis maupun pad tahap perancangan sistem.

Pada tahap analisis dan perancangan, kamus data dapat digunakan sebagai alat komunikasi antara analsisis sistem dengan pemakai sistem tentang data yang mengalir di sistem, yaitu tentang data yang masuk ke sistem dana tentang informasi yang dibutuhkan oleh pemakai sistem.

Pada tahap perancangan, kamus data digunakan untuk merancang *input*, merancang laporan-laporan dan *database*. Kamus data dibuat berdasarkan arus data yang ada di DFD. Arus data di DFD sifatnya global, hanya ditunjukkan nama arus datanya saja. Keterangan lebih lanjut tentang struktur data dari arus data di DFD secara lebih rinci dapat dilihat di kamus data. Kamus data harus dapat mencerminkan keterangan yang jelas tentang data yang dicatatnya, maka kamus data harus memuat nama arus data, alias, bentuk data, arus data, penjelasan, periode, volume, dan struktur data.[1]

2.2.6. Borland Delphi 7.0.

Borland delphi dirilis pertama kali pada bulan Februari 1995, dan sekarang sudah mencapai perkembangannya sampai dengan Delphi 8.

Delphi merupakan bahasa pemrograman yang mudah, karena delphi adalah bahasa pemrograman tingkat tinggi (high level) sehingga sangat memudahkan *user* untuk bermain-main di tingkat ini, apa lagi bagi mereka yang malas berurusan dengan level-level yang rendah.

Pemrograman delphi sangatlah mudah, kita tinggal klik dan drag, dan jadilah program apilkasi yang kita inginkan. Delphi adalah bahasa kompiler / penterjemah bahasa Delphi (awalnya dari pascal) yang merupakan bahasa tingkat tinggi sekelas dengan Basic, C. Bahasa pemrograman di delphi disebut bahasa prosedural artinya bahasa / sintaknya mengikuti urutan tertentu / prosedur.

Ada jenis pemrograman non-prosedural seperti pemrograman untuk kecerdasan buatan seperti bahasa prolog. Delphi termasuk keluarga visual sekelas visual basic, visual c, artinya perintah-perintah untuk membuat objek dapat dilakukan secara visual. Pemrogram tinggal memilih objek apa yang ingin dimasukkan kedalam Form/Window, lalu tingkah laku objek tersebut saat menerima event/aksi tinggal dibuat programnya. Delphi merupakan bahasa berorientasi objek, artinya nama objek, properti dan *methode/procedure* dikemas menjadisatu kemasan (*encapsulate*). Delphi adalah sebuah perangkat lunak (bahasa pemrograman) untuk membuat program/aplikasi komputer berbasis *windows*.

2.2.7. SQL Server

SQL Server adalah bahasa komputer standar ANSI (*American National Standard Institute*) untuk mengakses dan memanipulasi sistem basisdata. Pernyataan SQL digunakan untuk mengambil dan meng-update data dalam basisdata.

SQL adalah kependekan *Structured Query Language* yang merupakan bahasa basisdata standar industri. SQL menyediakan Data Definition Language

(DDL) dan Data Manipulation Language (DML). Walaupun dalam beberapa bagian saling berkaitan, perintah DDL memungkinkan dapat membuat dan mendefinisikan tabel (CREATE TABLE), dan indeks (CREATE INDEKS). Sebaliknya, perintah DML memungkinkan dapat memanipulasi data dan membangun query untuk mengambil data (SELECT) bebrapa tabel, menyisipkan (INSERT) data baru, memperbaiki (UPDATE) data, dan menghapus (DELETE) data.

SQL (*Structured Query Language*) pertama kali dikembangkan pada akhir tahun 1970-an di Laboratorium IBM San Jose, California. SQL umumnya dibaca “sequel” dan pada mulanya, dikembangkan untuk produk DB2 yang dimiliki oleh IBM. SQL adalah bahasa nonprosedural kontras dengan bahasa generasi ketiga (*Generation Language* / 3GL).