BAB II
TINJAUAN PUSTAKA

2.1 Profil Tempat Kerja Praktek

Sesuai dengan tugas dan fungsi Dinas Olahraga dan Pemuda Provinsi Jawa Barat dalam hal menetapkan kebijakan Pembangunan Olahraga dan Pelayanan Pengembangan Pembinaan Olahraga dengan Kelompok Sasaran Pelajar maka kegiatan pembinaan olahraga yang terus menerus dan berkelanjutan bagi setiap individu yang memiliki bakat dan prestasi harus terus ditingkatkan dengan berbagai upaya kegiatan pembinaa maupun peningkatan jumlah pertandingan atau kompetisi yang mendorong peningkatan tersedianya calon-calon atlit berbakat serta pemetaan potensi cabang olahraga di masing-masing Kabupaten dan Kota se Jawa Barat.

2.1.1 Sejarah Singkat DINAS OLAH RAGA DAN PEMUDA

Berdasarkan Peraturan Daerah (Perda) No. 21 Tahun 2008 yang memuat tentang Penyempurna- an Struktur Organisasi Perangkat Daerah (SOPD) pada Pemerintahan Provinsi Jawa Barat yang ditetapkan pada 24 Oktober 2008, Dinas Olah Raga dan Pemuda merupakan salah satu SOPD baru. Sementara pengisian jabatan nya untuk kepala dinas baru berlangsung pada 31 Desember 2008.

Dua substansi pokok wilayah garapan Dinas Olah Raga dan Pemuda Jabar menyangkut keolahragaan dan kepemudaan. Keolahragaan merupakan substansi yang apabila dikelola dan dilaksanakan dengan baik akan sangat bermakna dalam rangka membangun kualitas hidup masyarakat. Sementara kepemudaan merupakan kelompok masyarakat tertentu yang dalam bahasa Undang-Undang Kepemudaan terkelompok dalam batasan usia 16-30 tahun. Namun, sesuai substansi fokus tulisan ini akan lebih mengarah pada pengembangan keolahragaan.

Disorda yang paling tua adalah Jawa Timur yang sudah menembus usia 18 tahun.Terlepas dari usia perjalanan waktu, Disorda Jabar memiliki kapasitas untuk berkembang cepat guna mengejar ketertinggalan dari provinsi lain. Di samping

belum banyak aksi yang dapat dilakukan oleh Disorda Jabar dalam perjalanan waktunya selama satu tahun, kecuali menyelenggarakan berbagai kegiatan yang telah diprogramkan dalam tahun anggaran 2009 sekaligus menata arah, dan menyatukan visi dengan berbagai lembaga mitra. Untuk olah raga, Disorda bersinergi dengan KONI bersama organisasi induk cabang olah raga dan organisasi olah raga yang berkembang di masyarakat.

2.1.2 Badan Hukum DINAS OLAH RAGA DAN PEMUDA

Undang-undang Nomor 9 Tahun 1965 tentang Pembentukan Daerah Provinsi Jawa Barat (Lembaran Negara Republik Indonesia Tahun 1965 Nomor 52, Tambahan Lembaran Negara Republik Indonesia Nomor 2757);

Undang-undang Nomor 22 tahun 1999 tentang Pemerintahan Daerah (Lembaran Negara Republik Indonesia Tahun 1999 Nomor 60, Tambahan Lembaran Negara Republik Indonesia Nomor 3839);

Peraturan Pemerintah Nomor 25 Tahun 2000 tentang Kewenangan Pemerintah dan Kewenangan Provinsi Sebagai Daerah Otonomi;

Peraturan Pemerintah Nomor 84 Tahun 2000 tentang Pedoman Organisasi Perangkat Daerah.

Keputusan Presiden Nomor 44 Tahun 1999 tentang Teknik Penyusunan Peraturan Perundang-undangan dan Bentuk Rancangan Peraturan Pemerintah dan Rancangan Keputusan Presiden (Lembaran Negara Republik Indonesia Tahun 1999 Nomor 70).

Keputusan Menteri Dalam Negeri dan Otonomi Daerah Nomor 50 Tahun 2000 tentang Pedoman Susunan Organisasi dan Tata Kerja Perangkat Daerah Kabupaten/Kota.

Peraturan Daerah Provinsi Jawa Barat Nomor 1 Tahun 2008 tentang Urusan Pemerintahan yang Menjadi Kewenangan Pemerintah Daerah Provinsi Jawa Barat (Lembaran Daerah Provinsi Jawa Barat Tahun 2008 Nomor 1 Seri : E Nomor:1).

Peraturan Daerah Provinsi Jawa Barat Nomor 3 Tahun 2008 tentang Pembentukan Susunan Organisasi dan Tata Kerja Dinas Daerah Provinsi Jawa Barat (Lembaran Daerah Provinsi Jawa Barat Nomor 3 Seri : D Nomor: 2).

2.1.3 Visi, Misi dan Nilai Inti DINAS OLAH RAGA DAN PEMUDA

Visi merupakan keadaan masa depan (nyata dan terukur) yang diinginkan terjadi sebagai hasil dari perusahaan. Visi harus dirasakan oleh seluruh pihak yang terlibat. Visi juga harus mudah untuk dikomunikasikan dan dimengerti oleh seluruh organisasi.

Misi merupakan rumusan umum mengenai upaya-upaya yang akan dilaksanakan untuk mewujudkan visi.

1. **Visi**

"Terwujudnya masyarakat yang aktif dalam setiap penyelenggaran kegiatan olahraga khususnya pemuda Jawa Barat".

2. **Misi**
   a. Meningkatkan Pembinaan, Pemberdayaan, pembudayaan, dan Penghargaan serta pengembangan Sumber Daya Kepemudaan.
   b. Meningkatkan kualitas kehidupan keolahragaan untuk menciptakan prestasi Olahraga Nasional dan Internasional, serta kesehatan dan kesejahteraan.
   c. Meningkakan Kepemudaan Dan Keolahragaan Yang Berkarakter Dan Berdaya Saing Di Jawa Barat.

3. **Nilai Inti**

a. **Religius.** Berperilaku sesuai dengan norma agama yang dianut, berbudaya, dan berbudi pekerti luhur;

b. **Profesionalisme.** Memiliki rasa tanggung jawab, pro-aktif, kreatif, inovatif, produktif, berorientasi mutu, disiplin dan efisien dalam penggunaan sumber daya;

c. **Transparan.** Sesuai dengan mekanisme yang dapat dipercaya dan dipertanggungjawabkan;

d. **Peduli Lingkungan.** Berperilaku tertib, teratur, rapi dan dapat ("berkemauan kuat") menciptakan lingkungan yang bersih, indah dan nyaman;

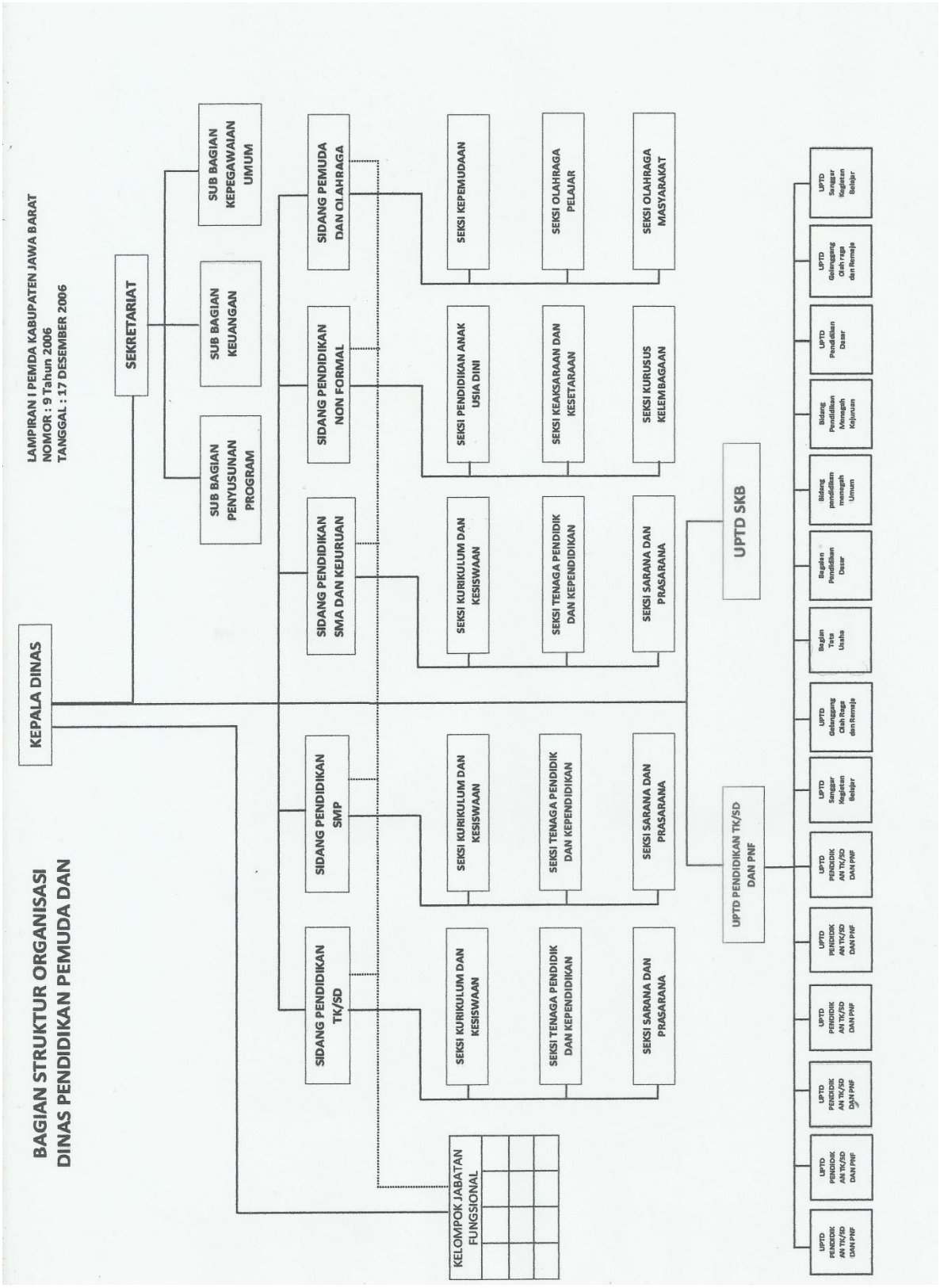
e. **Berwawasan Global.** Mampu beradaptasi, bekerja sama dan bersaing secara sehat sesuai dengan tuntutan dan tantangan zaman;

f. **Akuntabel.** Mampu memberi pertanggungjawaban kepada *stakeholder* dalam melaksanakan visi dan misinya.

2.1.4. Struktur Organisasi dan *Job Description*

Suatu instansi dapat berjalan dengan baik sesuai dengan tujuannya jika ditunjang dengan suatu struktur organisasi dan anggaran yang tersusun dengan baik pula. Dengan adanya struktur organisasi yang teratur maka setiap fungsi organisasi dapat berjalan sesuai dengan tugas dan perannya masing-masing. Dari struktur organisasi tersebut maka dengan sendirinya setiap pegawai dapat mengetahui kepada siapa harus mempertanggung jawabkan hasil pekerjaannya. Kata yang tepat dalam pembagian tugas yaitu ”*The Right Place*” (menempatkan pegawai pada pekerjaan/posisi yang tepat).

Struktur organisasi tersebut perlu dilengkapi dengan suatu gambaran uraian pekerjaan dan tanggung jawab yang harus dilaksanakan oleh setiap pemegang jawaban sehingga dapat menjadi efekif dan efesien. Tanpa dilengkapi dengan suatu uraian tugas, struktur organisasi tidak dapat melaksanakan tugas dan memenuhi tanggung jawabnya.



Gambar 2.1. Struktur Organisasi dan Tata Kerja

Struktur organisasi Dinas Pendidikan Pemuda dan Olah Raga Provinsi Jawa Barat adalah sebagai berikut :

1. Kepala Dinas.
2. Bagian Tata Usaha, membawahkan Sub Bagian Umum dan Sub Bagian Keuangan.
3. Kepala Bidang Pendidikan Dasar, membawahkan Seksi Kurikulum dan Bina Profesi, dan Seksi Sarana dan Prasarana.
4. Bidang Pendidikan Menengah Umum, membawahkan Seksi.
5. Kurikulum dan Bina Profesi, dan Seksi Sarana dan Prasarana.
6. Bidang Pendidikan Menengah Kejuruan, membawahkan Seksi Kurikulum dan Bina Profesi, dan Seksi Sarana dan Prasarana.
7. Bidang Pendidikan Luar Sekolah, Pemuda dan Olah Raga, membawahkan Seksi Pendidikan Luar Sekolah, dan Seksi Pemuda dan Olah Raga.
8. UPTD Pendidikan Dasar (6 buah).
9. UPTD Sanggar Kegiatan Belajar (SKB).
10. UPTD Gelanggang Olah Raga dan Remaja (GOR).

Sesuai dengan Peraturan Provinsi Jawa Barat Nomor 28 tahun 2004 tanggal 6 Desember 2004 tentang Tugas Pokok, Fungsi dan Tata Kerja Dinas Pendidikan Pemuda dan Olah Raga, Dinas Pendidikan Pemuda dan Olah Raga mempunyai fungsi tugas pokok melaksanakan sebagian kewenangan daerah di bidang pendidikan dan mempunyai fungsi:

1. Perumusan kebijakan teknis di bidang pendidikan.
2. Penyelenggaraan kebijakan dan koordinasi di bidang pendidikan.
3. Pemberian perizinan dan pelaksanaan pelayanan umum.
4. Pengendalian sumber daya aparatur, keuangan, prasarana dan sarana Dinas.
5. Penyelenggaraan tugas lain yang diberikan oleh Walikota sesuai dengan tugas dan fungsinya.

Kepala Dinas Pendidikan Pemuda dan Olah Raga membawahkan :

A. Bagian Tata Usaha
B. Bidang Pendidikan Dasar
C. Bidang Pendidikan Menengah Umum
D. Bidang Pendidikan Menengah Kejuruan
E. Bidang Diklusepora
F. UPTD Pendidikan Dasar
G. UPTD Gelanggang Olah raga dan Remaja
H. UPTD Sanggar Kegiatan Belajar.

A. BAGIAN TATA USAHA

Bagian Tata Usaha mempunyai tugas memfasilitasi dan memberikan pelayanan administrasi kepada seluruh satuan organisasi serta mengkoordinasikan kegiatan dalam lingkungan Dinas. Untuk melaksanakan tugas pokok, Bagian Tata Usaha mempunyai fungsi :

1. Pelaksanaan koordinasi dan penyusunan rencana dan program kerja
2. Pelaksanaan tugas administrasi kepegawaian, perlengkapan, keuangan, surat menyurat dan kerumahtanggaan
3. Pengevaluasian dan pelaporan kegiatan.

Bagian Tata Usaha membawahkan Sub Bagian Umum dan Sub Bagian Keuangan.

a. Sub Bagian Umum mempunyai tugas pokok melaksanakan sebagian tugas ketatausahaan di bidang kepegawaian dan urusan lainnya, dan mempunyai fungsi :

1. Pengelolaan tata persuratan
2. Penyiapan bahan pelaksanaan koordinasi dan penyusunan rencana dan program kerja
4. Pengelolaan kerumah tanggaan
5. Pelaksanaan administrasi kepegawaian
6. Pengevalusian dan pelaporan kegiatan.

b. Sub Bagian Keuangan mempunyai tugas pokok melaksanakan sebagian tugas ketatausahaan di bidang keuangan dan urusan lainnya, dan mempunyai fungsi :

1. Penyiapan bahan penyusunan anggaran.
2. Pengelolaan administrasi keuangan.
3. Pembinaan satuan pemegang kas.
4. Pengevaluasian dan pelaporan kegiatan.

B. BIDANG PENDIDIKAN DASAR

Bidang Pendidikan Dasar mempunyai tugas melaksanakan penyiapan perumusan kebijakan teknis dan evaluasi di bidang pengelolaan Pendidikan Dasar. Untuk melaksanakan tugas pokok, Bidang Pendidikan Dasar mempunyai fungsi sebagai berikut:

1. Penyiapan perumusan kebijakan teknis di bidang Pendidikan Dasar.
2. Pelaksanaan dan pengkordinasian kegiatan di bidang Pendidikan Dasar.
3. Pelaksanaan monitoring, evaluasi dan pelaporan.

Bidang Pendidikan Dasar membawahkan Seksi Kurikulum dan Bina Profesi, dan Seksi Sarana Prasarana.

a. Seksi Kurikulum dan Bina Profesi mempunyai tugas pokok melaksanakan tugas bidang Pendidikan Dasar dalam hal pengembangan Kurikulum dan Bina Profesi, dan mempunyai fungsi :

    1. Penyiapan bahan perumusan kebijakan teknis pengembangan Kurikulum dan Binas Profesi.

    2. Penyiapan bahan pelaksanaan pengembangan Kurikulum dan Binas Profesi.

    3. Pelaksanaan monitoring, evaluasi dan pelaporan kegiatan pengembangan Kurikulum dan Binas Profesi.

b. Seksi Sarana Prasarana mempunyai tugas pokok melaksanakan tugas bidang Pendidikan Dasar dalam hal Sarana dan Prasarana, dan mempunyai fungsi :

    1. Penyiapan bahan perumusan kebijakan teknis pengelolaan Sarana dan Prasarana.

    2. Penyiapan bahan pelaksanaan Sarana dan Prasarana.

    3. Pelaksanaan monitoring, evaluasi dan pelaporan kegiatan Sarana dan Prasarana.

C. BIDANG PENDIDIKAN MENENGAH UMUM

Bidang Pendidikan Menengah Umum mempunyai tugas melaksanakan penyiapan perumusan kebijakan teknis dan evaluasi di bidang pengelolaan Pendidikan Menengah Umum. Untuk melaksanakan tugas pokok, Bidang Pendidikan Menengah Umum mempunyai fungsi :

1. Penyiapan perumusan kebijakan teknis di bidang Pendidikan Menengah Umum.
2. Pelaksanaan dan pengkordinasian kegiatan di bidang Pendidikan Menengah Umum.

3. Pelaksanaan monitoring, evaluasi dan pelaporan.

Bidang Pendidikan Menengah Umum membawahkan Seksi Kurikulum dan Bina Profesi, dan Seksi Sarana Prasarana.

a. Seksi Kurikulum dan Bina Profesi mempunyai tugas pokok melaksanakan tugas bidang Pendidikan Menengah Umum dalam hal pengembangan Kurikulum dan Bina Profesi, dan mempunyai fungsi :

    1. Penyiapan bahan perumusan kebijakan teknis pengembangan Kurikulum dan Binas Profesi.

    2. Penyiapan bahan pelaksanaan pengembangan Kurikulum dan Binas Profesi.

    3. Pelaksanaan monitoring, evaluasi dan pelaporan kegiatan pengembangan Kurikulum dan Binas Profesi.

b. Seksi Sarana Prasarana mempunyai tugas pokok melaksanakan tugas bidang Pendidikan Menengah Umum dalam hal Sarana dan Prasarana, dan mempunyai fungsi :

    1. Penyiapan bahan perumusan kebijakan teknis pengelolaan Sarana dan Prasarana.

    2. Penyiapan bahan pelaksanaan Sarana dan Prasarana.

    3. Pelaksanaan monitoring, evaluasi dan pelaporan kegiatan Sarana dan Prasarana.

D. BIDANG PENDIDIKAN MENENGAH KEJURUAN

Bidang Pendidikan Menengah Kejuruan mempunyai tugas melaksanakan penyiapan perumusan kebijakan teknis dan evaluasi di bidang pengelolaan Pendidikan Menengah Kejuruan. Untuk melaksanakan tugas pokok, Bidang Pendidikan Menengah Kejuruan mempunyai fungsi :

1. Penyiapan perumusan kebijakan teknis di bidang Pendidikan Menengah Kejuruan.
2. Pelaksanaan dan pengkordinasian kegiatan di bidang Pendidikan Menengah Kejuruan.
3. Pelaksanaan monitoring, evaluasi dan pelaporan.

Bidang Pendidikan Menengah Kejuruan membawahkan Seksi Kurikulum dan Bina Profesi, dan Seksi Sarana Prasarana.

a. Seksi Kurikulum dan Bina Profesi mempunyai tugas pokok melaksanakan tugas bidang Pendidikan Menengah Kejuruan dalam hal pengembangan Kurikulum dan Bina Profesi, dan mempunyai fungsi :

    1. Penyiapan bahan perumusan kebijakan teknis pengembangan Kurikulum dan Binas Profesi.
    2. Penyiapan bahan pelaksanaan pengembangan Kurikulum dan Binas Profesi.
    3. Pelaksanaan monitoring, evaluasi dan pelaporan kegiatan pengembangan Kurikulum dan Binas Profesi.

b. Seksi Sarana Prasarana mempunyai tugas pokok melaksanakan tugas bidang Pendidikan Menengah Kejuruan dalam hal Sarana dan Prasarana, dan mempunyai fungsi :

    1. Penyiapan bahan perumusan kebijakan teknis pengelolaan Sarana dan Prasarana.

2. Penyiapan bahan pelaksanaan Sarana dan Prasarana.

3. Pelaksanaan monitoring, evaluasi dan pelaporan kegiatan Sarana dan Prasarana.

E. BIDANG PENDIDIKAN LUAR SEKOLAH, PEMUDA DAN OLAH RAGA (DIKLUSPORA)

Bidang Dikluspora mempunyai tugas melaksanakan penyiapan perumusan kebijakan teknis dan evaluasi di bidang Dikluspora. Untuk melaksanakan tugas pokok, Bidang Dikluspora mempunyai fungsi :

1. Penyiapan perumusan kebijakan teknis di bidang Dikluspora;
2. Penyiapan bahan Pelaksanaan kegiatan di bidang Dikluspora;
4. Pelaksanaan monitoring, evaluasi dan pelaporan.

Bidang Dikluspora membawahkan Seksi Pendidikan Luar Sekolah, dan Seksi Pemuda dan Olah Raga.

a. Seksi Pendidikan Luar Sekolah mempunyai tugas pokok melaksanakan tugas bidang Pendidikan Luar Sekolah dalam hal pengembangan Kurikulum dan Bina Profesi, dan mempunyai fungsi :

    1. Penyiapan bahan perumusan kebijakan teknis Pendidikan Luar Sekolah;
    2. Penyiapan bahan pelaksanaan kegiatan Pendidikan Luar Sekolah;
    3. Pelaksanaan monitoring, evaluasi dan pelaporan kegiatan Pendidikan Luar Sekolah.

b. Seksi Pemuda dan Olah Raga mempunyai tugas pokok melaksanakan tugas bidang Dikluspora dalam hal Pemuda dan Olah Raga, dan mempunyai fungsi :

    1. Penyiapan bahan perumusan kebijakan teknis kegiatan Pemuda dan Olah Raga.

2. Penyiapan bahan pelaksanaan kegiatan Pemuda dan Olah Raga.
3. Pelaksanaan monitoring, evaluasi dan pelaporan kegiatan kegiatan Pemuda dan Olah Raga.

F. UNIT PELAKSANA TEKNIS DINAS DIKDAS KECAMATAN

Unit Pelaksana Teknis Dinas (UPTD) Dikdas Kecamatan mempunyai tugas pokok melaksanakan pelayanan di bidang Pendidikan Dasar. Untuk melaksanakan tugas pokok, UPTD Dikdas Kecamatan mempunyai fungsi :

1. Penyusunan rencana dan program kerja UPTD Dikdas Kecamatan.
2. Pengelolaan administrasi, kepegawaian dan administrasi umum di lingkungan UPTD Dikdas Kecamatan.
3. Pelaksanaan kordinasi, pengendalian, pengawasan kegiatan dan penggunaan sarana dan prasarana Pendidikan Dasar Kecamatan.
4. Pelaksanaan pelayanan pendidikan dasar Pengevaluasian dan pelaporan kegiatan.

2.2 Landasan Teori

Dalam melaksanakan kerja praktek ini digunakan dasar-dasar teori sistem informasi sebagai bahan acuan. Berisi teori-teori yang bisa dijadikan dasar dan acuan dalam perancangan sistem informasi ini.

2.2.1 Pengertian Sistem

Definisi sistem adalah suatu kesatuan yang terdiri atas komponen atau elemen yang saling berinteraksi, saling terkait, atau saling bergantung membentuk keseluruhan yang kompleks. Dalam definisi yang paling umum, sebuah sistem adalah sekumpulan objek/benda yang memiliki hubungan diantara mereka.

Karakteristik Sistem

Suatu sistem mempunyai karakteristik atau sifat-sifat yang tertentu, yaitu mempunyai komponen-komponen (*components*), batas sistem (*boundary*), lingkungan luar sistem (*environments*), penghubung (*interface*), masukan (*input*), keluaran (o*utput*), pengolahan (*process*) dan sasaran (*objective*) atau tujuan (*goal*).

a. Komponen Sistem (*components*)

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk satu kesatuan. Komponen-komponen sistem atau elemen-elemen sistem dapat berupa suatu sub sistem atau bagian-bagian dari sistem. Setiap sistem tidak perduli betapapun kecilnya, selalu mengandung komponen-komponen atau sub sistem.

b. Batasan Sistem (*boundary*)

Merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan. Batas sustu sistem menunjukan ruang lingkup (*scope*) dari sistem tersebut.

c. Lingkungan Luar Sistem (*environments*)

Lingkungan Luar Sistem (*environments*) dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkungan luar sistem dapat bersifat menguntungkan dan dapat juga bersifat merugikan sistem tersebut. Lingkungan luar yang menguntungkan merupakan energi dari sistem dan dengan demikian harus tetap dijaga dan dipelihara. Sedang lingkungan luar yang merugikan harus ditahan dan dikendalikan, kalau tidak maka akan mengganggu kelangsungan hidup dari sistem.

d. Penghubung Sistem (*interface*)

Merupakan media penghubung antara satu subsistem dengan subsistem yang lainnya. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem yang lainnya. Keluaran (*output*) dari subsistem menjadi masukan (*input*) untuk susistem yang lainnya dengan melalui penghubung. Dengan penghubung satu subsistem dapat berintegrasi dengan subsistem yang lainnya membentuk satu kesatuan.

e. Masukan Sistem (*input*)

Masukan adalah energi yang dimasukan ke dalam sistem. Masukan dapat berupa masukan perawatan (*maintenance input*) dan masukan sinyal (signal input). Maintenanace input adalah energi yang dimasukan supaya sistem dapat beroperasi. Signal input adalah energi yang diproses untuk mendapatkan keluaran.

f. Pengolahan Sistem (*process*)

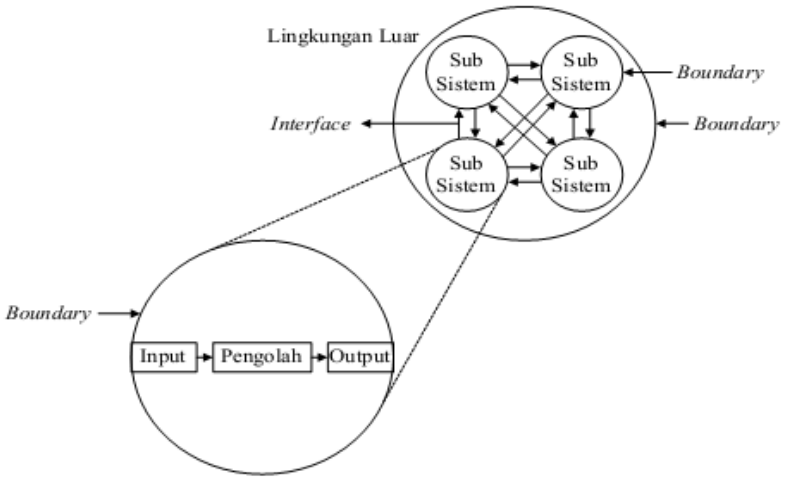
Suatu sistem dapat mempunyai bagian pengolah yang akan mengubah masukan menjadi keluaran.

g. Keluaran Sistem (*output*)

Keluaran (*output*) adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dari sisa pembuangan.

h. Sasaran Sistem

Suatu sistem pasti mempunyai tujuan (*goal*) atau sasaran (*objective*). Kalau suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak akan ada gunanya. Sasaran dari sistem sangat menentukan sekali masukan yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuannya.



Gambar 2.2. Karakteristik Sistem.

2.2.2 Pengertian Informasi

Konsep dasar informasi, data adalah fakta atau yang dapat digunakan sebagai input dalam menghasilkan informasi sedangkan informasi adalah "hasil pengolahan data yang memberikan arti dan manfaat bagi orang yang akan menerimanya".

1. Siklus Informasi

Data merupakan bentuk yang masih mentah yang belum dapat bercerita banyak, sehingga perlu diolah lebih lanjut. Data diolah melalui suatu model untuk dihasilkan informasi. Data yang diolah untuk menghasilkan informasi menggunakan suatu model proses tertentu.

Data yang diolah melalui suatu model menjadi informasi, penerima kemudian menerima informasi tersebut, membuat suatu keputusan dan melakukan tindakan, yang berarti menghasilkan suatu tindakan yang lainnya yang akan membuat sejumlah data kembali. Data tersebut akan ditanggap sebagai input, diproses kembali lewat suatu model dan membentuk suatu siklus. Menurut John Burch disebut dengan siklus informasi (*information cycle*) atau siklus ini disebut juga dengan siklus pengolahan data (*data processing cycles*).

2. Kualitas Informasi

Kualitas dari suatu informasi tergantung dari tiga hal, yaitu informasi harus akurat, tepat pada waktunya dan relevan, diantaranya:

1. Akurat, berarti informasi harus bebas dari kesalahan-kesalahan dan tidak menyesatkan. Akurat juga berarti informasi harus jelas mencerminkan maksudnya. Informasi harus akurat karena dari sumber informasi sampai ke penerima informasi kemungkinan banyak terjadi gangguan yang dapat merubah atau merusak informasi tersebut.
2. Tepat pada waktunya, berarti informasi yang datang pada penerima tidak boleh terlambat.
3. Relevan, berarti informasi tersebut mempunyai manfaat untuk pemakainya.

3. Nilai Informasi

Nilai informasi ditentukan dari dua hal, yaitu manfaat dan biaya mendapatkannya. Suatu informasi dikatakan bernilai bila manfaatnya lebih efektif dibandingkan dengan biaya mendapatkannya.

2.2.3 Sistem Informasi

Computer Based Information System (CBIS) atau yang dalam bahasa Indonesia disebut juga Sistem Informasi Berbasis Komputer merupakan sistem pengolah data menjadi sebuah informasi yang berkualitas dan dipergunakan untuk suatu alat bantu pengambilan keputusan. Sistem Informasi yang akurat dan efektif, dalam kenyataannya selalu berhubungan dengan istilah "*computer-based*" atau pengolahan informasi yang berbasis pada komputer. Sistem Informasi "berbasis komputer" mengandung arti bahwa komputer memainkan peranan penting dalam sebuah sistem informasi.

Secara teori, penerapan sebuah Sistem Informasi memang tidak harus menggunakan komputer dalam kegiatannya. Tetapi pada prakteknya tidak mungkin sistem informasi yang sangat kompleks itu dapat berjalan dengan baik jika tanpa adanya komputer. Sistem Informasi merupakan sistem pembangkit informasi. Dengan integrasi yang dimiliki antar subsistemnya, sistem informasi akan mampu menyediakan informasi yang berkualitas, tepat, cepat dan akurat sesuai dengan manajemen yang membutuhkannya.

2.2.4. Basis Data

Basis data (*database*), atau sering pula disebut basisdata, adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Perangkat lunak yang digunakan untuk mengelola dan memanggil kueri (*query*) basis data disebut sistem manajemen basis data (*database management system, DBMS*). Sistem basis data dipelajari dalam ilmu informasi.

Istilah "basis data" berawal dari ilmu komputer. Meskipun kemudian artinya semakin luas, memasukkan hal-hal di luar bidang elektronika, artikel ini

mengenai basis data komputer. Catatan yang mirip dengan basis data sebenarnya sudah ada sebelum revolusi industri yaitu dalam bentuk buku besar, kuitansi dan kumpulan data yang berhubungan dengan bisnis.

Konsep dasar dari basis data adalah kumpulan dari catatan-catatan, atau potongan dari pengetahuan. Sebuah basis data memiliki penjelasan terstruktur dari jenis fakta yang tersimpan di dalamnya: penjelasan ini disebut skema. Skema menggambarkan obyek yang diwakili suatu basis data, dan hubungan di antara obyek tersebut. Ada banyak cara untuk mengorganisasi skema, atau memodelkan struktur basis data: ini dikenal sebagai model basis data atau model data. Model yang umum digunakan sekarang adalah model relasional, yang menurut istilah layman mewakili semua informasi dalam bentuk tabel-tabel yang saling berhubungan dimana setiap tabel terdiri dari baris dan kolom (definisi yang sebenarnya menggunakan terminologi matematika). Dalam model ini, hubungan antar tabel diwakili dengan menggunakan nilai yang sama antar tabel. Model yang lain seperti model hierarkis dan model jaringan menggunakan cara yang lebih eksplisit untuk mewakili hubungan antar tabel.

Istilah basis data mengacu pada koleksi dari data-data yang saling berhubungan, dan perangkat lunaknya seharusnya mengacu sebagai sistem manajemen basis data (*database management system/DBMS*). Jika konteksnya sudah jelas, banyak administrator dan programer menggunakan istilah basis data untuk kedua arti tersebut.

1. Data, Informasi dan Basis Data

Data merupakan fakta mengenai suatu objek seperti manusia, benda, peristiwa, konsep, keadaan dan sebagainya yang dapat dicatat dan mempunyai arti secara implisit. Data dapat dinyatakan dalam bentuk angka, karakter atau simbol, sehingga bila data dikumpulkan dan saling berhubungan maka dikenal dengan istilah basis data (database).

Sedangkan menurut George Tsu-der Chou basis data merupakan kumpulan informasi bermanfaat yang diorganisasikan ke dalam aturan yang khusus.

Informasi ini adalah data yang telah diorganisasikan ke dalam bentuk yang sesuai dengan kebutuhan seseorang.

Menurut *Encyclopedia of Computer Science and Engineer*, para ilmuwan di bidang informasi menerima definisi standar informasi yaitu data yang digunakan dalam pengambilan keputusan.

Definisi lain dari basis data menurut *Fabbri* dan *Schwab* adalah sistem berkas terpadu yang dirancang terutama untuk meminimalkan duplikasi data.

Menurut Ramez Elmasri mendefinisikan basis data lebih dibatasi pada arti implisit yang khusus, yaitu:

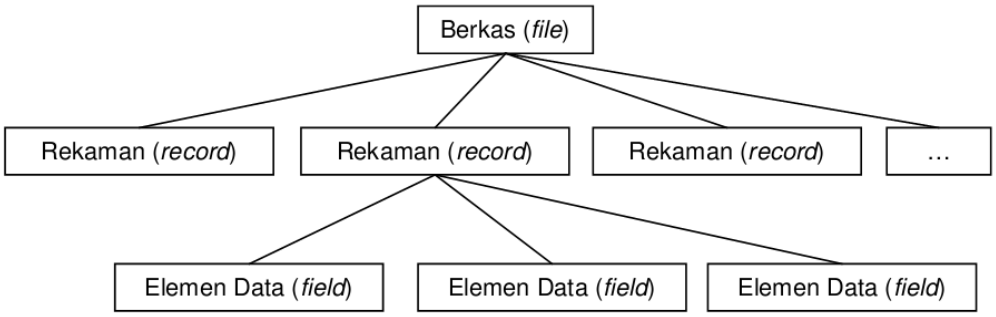
a. Basis data merupakan penyajian suatu aspek dari dunia nyata (real world).
b. Basis data merupakan kumpulan data dari berbagai sumber yang secara logika mempunyai arti implisit. Sehingga data yang terkumpul secara acak dan tanpa mempunyai arti, tidak dapat disebut basis data.
c. Basis data perlu dirancang, dibangun dan data dikumpulkan untuk suatu tujuan. Basis data dapat digunakan oleh beberapa user dan beberapa aplikasi yang sesuai dengan kepentingan user.

Dari beberapa definisi-definisi tersebut, dapat dikatakan bahwa basis data mempunyai berbagai sumber data dalam pengumpulan data, bervariasi derajat interaksi kejadian dari dunia nyata, dirancang dan dibangun agar dapat digunakan oleh beberapa user untuk berbagai kepentingan.

2. Hirarki Data

Data diorganisasikan kedalam bentuk elemen data (field), rekaman (record), dan berkas (*file*). Definisi dari ketiganya adalah sebagai berikut:

a. Elemen data adalah satuan data terkecil yang tidak dapat dipecah lagi menjadi unit lain yang bermakna. Misalnya data siswa terdiri dari NIS, Nama, Alamat, Telepon atau Jenis Kelamin.
b. Rekaman merupakan gabungan sejumlah elemen data yang saling terkait. Istilah lain dari rekaman adalah baris atau tupel.
c. Berkas adalah himpunan seluruh rekaman yang bertipe sama.

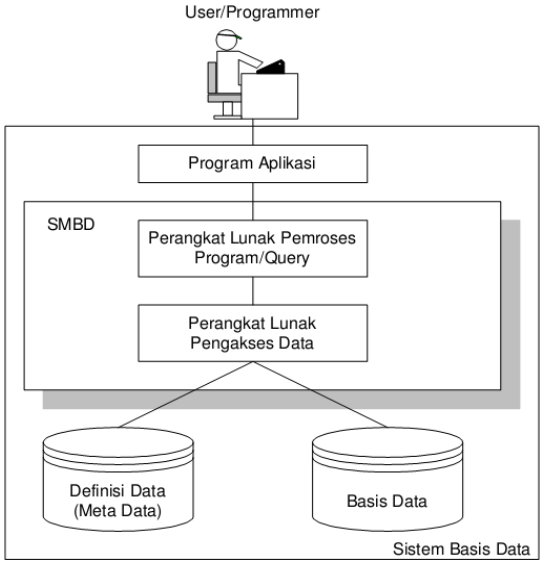


Gambar 2.3. Hirarki Data

3. Sistem Basis Data

Gabungan antara basis data dan perangkat lunak SMBD (Sistem Manajemen Basis Data) termasuk di dalamnya program aplikasi yang dibuat dan bekerja dalam satu sistem disebut dengan Sistem Basis Data.

Sistem basis data dapat dianggap sebagai tempat untuk sekumpulan berkas data yang terkomputerisasi dengan tujuan untuk memelihara informasi dan membuat informasi tersebut tersedia saat dibutuhkan.



Gambar 2.4. Konsep Sistem Basis Data

4. Ikhtisar Basis Data

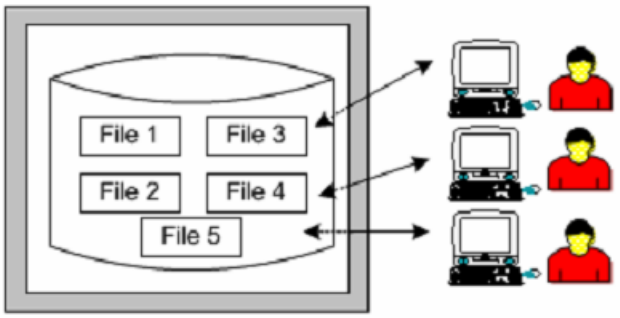
a. **Basis**: markas/gudang, tempat bersarang/berkumpul.

b. **Data**: representasi fakta dunia nyata yang mewakili suatu obyek (spt, manusia: dosen, mhs, pelanggan,dll; barang: buku, meja; peristiwa, konsep, dsb.), yang direkam baik dalam bentuk angka, huruf, teks, gambar atau suara.

c. **Definisi Basis Data**

1. Basis Data adalah sekumpulan data yang saling ber-relasi.
2. Himpunan kelompok data (arsip) yang saling berhubungan, yang diorganisasi sedemikian rupa, sehingga kelak dapat dimanfaatkan kembali dengan cepat.
3. Kumpulan data yang saling berhubungan yang disimpan secara bersama tanpa adanya pengulangan (redudansi) data.
4. Kumpulan file/tabel/arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronik.

d. **Sistem Basis Data.** Sistem yang terdiri atas sekumpulan tabel data yang saling berhubungan dan sekumpulan program (DBMS: *Database Management System*) yang memungkinkan berbagai user dan/atau program lain dapat mengakses dan memanipulasi tabel-tabel tersebut.



Gambar 2.5. Sistem Basis Data

e. **DBMS (*Database Management System*).** Kumpulan program yang digunakan user untuk me-*management* database (*create, maintain*)

DBMS mencakup proses:

1. *Defining*: database mendefiniskan tipe data, struktur dan batasan (*constraint*) dari data yang disimpan dalam database.

2. *Manipulating*: database mencakup berbagai fungsi dan query untuk mendapatkan data yang dicari, termasuk operasi *insert*, *update* dan *delete* serta dalam *generate report data*.
3. *Sharing*: database dapat diatur untuk dapat sharing multiple user dan program untuk mengakses database secara bersama-sama.

Fungsi yang lebih penting dari DBMS adalah proteksi dan maintain database dalam jangka panjang.

1. Proteksi: mengandung *system protection* yang menangani kondisi malfunction (*crash*) baik pada hardware ataupun software, juga mengandung *security protection* yang menangani pengaksesan oleh user terlarang.
2. Maintain: mengandung sistem maintaining yang selalu meningkatkan kebutuhan perubahan tiap waktu.

f. **Sistem Basis Data = DBMS + Basis data**

g. **Struktur File Database**

1. ***Data*** adalah satu satuan informasi yang akan diolah, dimana sebelum diolah dikumpulkan di dalam suatu file database. Pengumpulan data dilakukan secara sistematis menurut struktur file database tersebut.
2. ***RECORD*** adalah data yang isinya merupakan satu kesatuan seperti Nama, Alamat, Nomor Telepon. Setiap keterangan yang mencakup Nama, Alamat dan Nomor Telepon dinamakan satu record. Dan setiap record diberi nomor urut yang disebut nomor record (*Record Number*). Ukuran suatu file database ditentukan oleh jumlah record yang tersimpan di dalamnya.
3. ***FIELD*** adalah sub bagian dari Record. Dari contoh isi *record* diatas maka terdiri dari 3 field, yaitu field Nama, field Alamat dan field Nomor Telepon.

h. **Keuntungan Sistem Basis Data**

1. Terkontrolnya kerangkapan data

Dalam basis data hanya mencantumkan satu kali saja *field* yang sama yang dapat dipakai oleh semua aplikasi yang memerlukannya.

2. **Terpeliharanya keselarasan (ke-konsistenan) data**

   Apabila ada perubahan data pada aplikasi yang berbeda maka secara otomatis perubahan itu berlaku untuk keseluruhan

3. **Data dapat dipakai secara bersama (*shared*)**

   Data dapat dipakai secara bersama-sama oleh beberapa program aplikasi (secara *batch* maupun *on-line*) pada saat bersamaan.

4. **Dapat diterapkan standarisasi**

   Dengan adanya pengontrolan yang terpusat maka DBA dapat menerapkan standarisasi data yang disimpan sehingga memudahkan pemakaian, pengiriman maupun pertukaran data.

5. **Keamanan data terjamin**

   DBA dapat memberikan batasan-batasan pengaksesan data, misalnya dengan memberikan password dan pemberian hak akses bagi *user* (misal : *modify, delete, insert, retrieve*)

6. **Terpeliharanya integritas data**

   Jika kerangkapan data dikontrol dan ke konsistenan data dapat dijaga maka data menjadi akurat

7. **Terpeliharanya keseimbangan (keselarasan) antara kebutuhan data yang berbeda dalam setiap aplikasi**

   Struktur basis data diatur sedemikian rupa sehingga dapat melayani pengaksesan data dengan cepat

8. **Kemandirian data (*Data independence*)**

   Dapat digunakan untuk bermacam-macam program aplikasi tanpa harus merubah format data yang sudah ada

i. Contoh DBMS:

   a. Dbase
   b. FoxPro
   c. Ingres
   d. Postgresql
   e. MySQL
   f. MS Access

g. SQL Server
h. Oracle
i. DB2, dll.

j. **Tujuan Basis Data**

1. Kemudahan dan kecepatan dalam pengambilan data (*speed*).
2. Efisiensi ruang penyimpanan (*space*). Mengurangi / menghilangkan redudansi data
3. Keakuratan (*accuracy*). Pembentukan kode & relasi antar data berdasar aturan / batasan (*constraint*) tipe data, domain data, keunikan data, untuk menekan ketidakakuratan saat entry / penyimpanan data.
4. Ketersediaan (*availability*). Pemilahan data yang sifatnya pasif dari database aktif.
5. Kelengkapan (*completeness*). Kompleksnya data menyebabkan perubahan struktur database.
6. Keamanan (*security*). Memberikan keamanan atas hak akses data.
7. Kebersamaan pemakaian (*sharability*). Bersifat multiuser.

k. **Manfaat penggunaan DBMS**

1. *Controlling Redundancy*

   Redundancy: duplikasi data, penyimpanan data secara berulang.

   Redudancy salah satu syarat larangan dalam database relasional, karena akan menimbulkan *inconsistensi* data.

   Dengan *controlling redundancy*, selain akan meningkatkan *performance query* juga menjaga konsistensi data.
2. *Restricting Unauthorized Access*

   Memberikan pengaturan hak akses / batasan akses *user database*
3. *Providing Persistent Storage for Program Objects*

   Menyediakan ruang penyimpanan khusus untuk obyek-obyek program (ex. *Object-Oriented Database system* yang menyimpan obyek-obyek pemrograman berbasis obyek).
4. *Providing Storage Structures for Efficient Query Processing*

Menyediakan struktur penyimpanan yang bagus untuk efisiensi proses query.

5. *Providing Backup and Recovery*
6. *Providing Multiple User Interface*
7. *Representing Complex Relationship among Data*
8. *Enforcing Integrity Constraints*
9. *Permitting Inferencing and Actions using Rules*
   Menyediakan actions khusus berdasarkan aturan (*rules*) yang telah ditetapkan dalam sistem database.
10. *Additional Implications of using the Database Approach*
    Flexible, *up-to-date* data, ekonomis, dll.

l. Pengguna Basis Data

Para pengguna database dapat dibagi menjadi Pengguna database ("*Actor on the scene*") dan Pekerja dibalik database ("*Worker behind the scene*").

1) "*Actor on the scene*" lebih cenderung menggunakan / ada keterkaitan dengan penggunaan database dan dapat dikelompokkan menjadi:

a. *Database Administrators (DBA)***.** Orang yang memiliki tanggung jawab penuh dalam manajemen database (pengaturan hak akses, koordinasi dan monitoring, kebutuhan hardware/software). Dalam pekerjaannya biasanya dibantu oleh staf Admin.

1) Tugas DBA:
   Mengontrol DBMS dan software-software
   Memonitor siapa yang mengakses basis data
   Mengatur pemakaian basis data
   Memeriksa *security, integrity, recovery* dan *concurency*
2) Program Utility yang digunakan oleh DBA:
   Loading Routines. Membangun versi utama dari database.
   Reorganization Routines. Mengatur / mengorganisasikan kembali database.
   Journaling Routines. Mencatat semua operasi pemakaian database.

Recovery Routines. Menempatkan kembali data, sebelum terjadinya kerusakan.

Statistical Analysis Routines. Membantu memonitor kehandalan sistem.

b. *Database Designers*. Orang bertanggung jawab dalam identifikasi data yang tersimpan dalam database, menentukan struktur data yang tepat untuk disimpan dalam database. Perlu koordinasi akan kebutuhan user database

c. *End Users*. Adalah orang-orang yang pekerjaannya membutuhkan akses ke database untuk melakukan query, update maupun genereate *report database*.

   1) *End user* tak tetap (*Casual end users*): user yang tidak selalu mengakses database, tapi kadang memerlukan informasi terbaru.
   2) *Naïve / parametric end users*: Pemakai yang berinteraksi dengan sistem basis data melalui pemanggilan satu program aplikasi permanen (*executable program*) yang telah ditulis/disediakan sebelumnya. Pekerjaan selalu konstan query dan update data. (spt: bank teller, pegawai reservasi, dll.).
   3) *Sophisticated end users*: Pemakai yang menulis aplikasi basis data non konvensional, tetapi untuk keperluan-keperluan khusus seperti aplikasi AI, Sistem Pakar, Pengolahan Citra,dll, yang bisa saja mengakses basis data dengan/tanpa DBMS yang bersangkutan. Melengkapi kebutuhan database user. (spt: *engineer*, *scientist*, *business analyst*).
   4) *Stand-alone users*: user yang memaintain personal database.

d. *System Analyst dan Application Programmers (Software Engineers)*

   1) *System Analyst*: orang menentukan kebutuhan sistem end user.
   2) *Application Programmers (Software Engineering)*: orang yang kerjaannya berhubungan dengan kebutuhan koneksi database.

2) *"Workers behind the scene"*. Orang-orang yang tidak tertarik pada database, akan tetapi lebih cenderung pekerjaannya men-*develop tool* untuk kebutuhan database. Pengguna dapat dikelompokkan menjadi:

a. *DBMS system designers dan implementer*. Orang-orang yang merancang dan meng-implementasikan modul-modul dan interface paket-paket software DBMS. (ex. Modul: catalog, procs query lang., procs interface, access & buffering data, controlling cuncurrency, handling data recovery & security; interfacing: interface for integrated system)

b. *Tool developers*. Orang-orang yang merancang dan mengimplementasikan tools untuk mendukung software DBMS. (tool untuk meningkatkan performance database, tool untuk monitoring operasional database, dll.)

c. *Operators dan maintenance personnel*. Para personel administrator yang bertanggung jawab akan jalannya operasional database termasuk maintenance (hardware/software) DBMS.

m. **Komponen Sistem Basis Data**

1. Perangkat Keras (*Hardware*). Komputer, memori, storage (*Harddisk*), *peripheral*, dll.
2. Sistem Operasi (*Operating System*). Program yang menjalankan sistem komputer, mengendalikan resource komputer dan melakukan berbagai operasi dasar sistem komputer.
3. Basis Data (*Database*). Menyimpan berbagai obyek database (struktur tabel, indeks,dll).
4. DBMS (*Database Management System*). Perangkat lunak yang memaintain data dalam jumlah besar.
5. Pemakai (*User*). Para pemakai database.
6. Aplikasi (perangkat lunak) lain. Program lain dalam DBMS.

n. **Abstraksi Data.** Dalam database, data disimpan dan diperlihara dengan baik dan terstruktur oleh DBMS. Sistem ini menyembunyikan detail tentang bagaimana data disimpan dipelihara. Sehingga seringkali data yang terlihat

oleh user, berbeda dengan data yang tersimpansecara fisik. Abstraksi data merupakan tingkatan/level dalam melihat bagaimana menampilkan data dalam sebuah sistem database.

Terdapat 3 Level abstraksi data:

1) Level Fisik (Physical Level). Level terendah dalam abstraksi data, yang menunjukkan bagaimana sesungguhnya suatu data disimpan. User melihat data sebagai gabungan dari struktur dan datanya sendiri.

   Tingkatan ini berurusan dengan:

   Alokasi ruang penyimpanan untuk data dan indeks

   Deskripsi record untuk penyimpanan

   Penempatan record data

   Teknik kompresi dan enkripsi data

2) Level Konsepsual (*Conceptual Level*). Menggambarkan data apa yang sebenarnya disimpan dalam database, serta hubungannya (*relationship*) dengan data lainnya. Berisi struktur logika database yang hanya dapat dilihat oleh DBA.

   Tingkat konsepsual ini menyatakan:

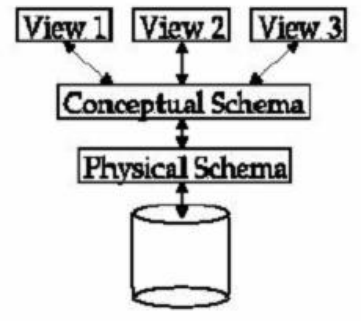
   Entitas, atribut dan relasinya

   Konstrain-konstrain terhadap data

   Informasi semantiks data

   Informasi keamanan dan integritas data

3) Level Pandangan (*View Level*). Level tertinggi dari abstraksi data, yang hanya menampilkan data hanya sebagian dari database. Tidak semua user membutuhkan semua data dalam database.



Gambar 2.6. Abstraksi Data

o. **Bahasa Basis Data**. DBMS merupakan perantara antara user dengan database. Cara komunikasi diatur dalam suatu bahasa khusus yang telah ditetapkan oleh DBMS (Contoh: SQL, dBase, QUEL, dsb.). Bahasa database, dibagi dalam 2 bentuk yaitu *Data Definition Language (DDL)* dan *Data Manipulation Language (DML).*

1) Data Definition Language (DDL)

a) Digunakan dalam membuat tabel baru, indeks, mengubah tabel, menentukan struktur tabel, dsb.

b) Hasil dari kompilasi perintah DDL berupa kumpulan tabel yang disimpan dalam file khusus: Kamus Data (*Data Dictionary*).

c) *Data Dictionary* merupakan metadata (*superdata*), yaitu data yang mendeskripsikan data sesungguhnya. *Data dictionary* ini akan selalu diakses dalam suatu operasi database sebelum suatu file data yang sesungguhnya diakses.

2) Data Manipulation Language (DML)

a) Digunakan dalam memanipulasidan pengambilan data pada database.

b) Manipulasi data, dapat mencakup:

a. Pemanggilan data yang tersimpan dalam database (*query*)

b. Penyisipan/penambahandata baru ke database (*Insert*)

c. Pengubahan data pada database (*Update*)

d. Penghapusandata dari database (*Delete*)

c) Terdapat dua (2) jenis DML:

a. Prosedural. Menghendaki user untuk menspesifikasikandata apa yang diperlukan dan bagaimana cara mendapatkan data itu. (Contoh: bahasa C/C++, PL/SQL, dsb.)

b. Nonprosedural. Menghendaki user untuk menspesifikasikan data apa yang dibutuhkan, tanpa harus menspesifikasikan bagaimana cara mendapatkan data tersebut. (Contoh: SQL).

p. **Istilah - Istilah Dasar Basis Data**

1) **Enterprise.** Suatu bentuk organisasi seperti: bank, universitas, rumah sakit, pabrik, dsb. Data yang disimpan dalai basis data merupakan data operasional dari suatu enterprise. (Contoh data operasional: data keuangan, data mahasiswa, data pasien)
2) **Entitas.** Suatu obyek yang dapat dibedakan dari lainnya yang dapat diwujudkan dalai basis data. Contoh Entitas dalam lingkungan bank terdiri dari : Nasabah, Simpanan, Hipotik Contoh Entitas dalam lingkungan universitas terdiri dari : Mahasiswa, mata kuliah Kumpulan dari entitas disebut **Himpunan Entitas (**Contoh: semua nasabah, semua mahasiswa)
3) **Atribut (Elemen Data).** Karakteristik dari suatu entitas. (Contoh: Entitas Mahasiswa atributnya terdiri dari Npm, Nama, Alamat, Tanggal lahir.)
4) **Nilai Data (Data Value).** Isi data / informasi yang tercakup dalai setiap elemen data. (Contoh: Atribut Nama Mahasiswa dapat berisi Nilai Data: Diana, Sulaeman, Lina)
5) **Kunci Elemen Data (*Key Data Element*).** Tanda pengenal yang secara unik mengidentifikasikan entitas dari suatu kumpulan entitas. (Contoh: Entitas Mahasiswa yang mempunyai atribut-atribut npm, nama, alamat, tanggal lahir menggunakan Kunci Elemen Data npm.)
6) **Record Data.** Kumpulan Isi Elemen data yang saling berhubungan. (Contoh: kumpulan atribut npm, nama, alamat, tanggal lahir dari Entitas Mahasiswa berisikan : "10200123", "Sulaeman", "Jl. Sirsak 28 Jakarta", "8 Maret 1983".)

2.2.5. Pemodelan Data dan Diagram

Pemodelan sistem memainkan peranan yang penting dalam pengembangan sistem. Pemodelan data kadang-kadang disebut pemodelan database karena model data kadang-kadang diimplementasikan sebagai sebuah database. Pemodelan data dapat di gambarkan dengan ERD (*Entity Relationship Diagram*).

Entity Relationship Diagram (ERD) merupakan suatu diagram yang digunakan untuk menghubungkan antar elemen (*Relational Condition*), dimana pada tahap selanjutnya dapat diimplementasikan kedalam bentuk tabel relasi.

ERD menggunakan sejumlah notasi dan simbol untuk menggambarkan struktur dan hubungan antar data, pada dasarnya ada 3 macam simbol yang digunakan, yaitu:

1. Entity

   Adalah suatu objek yang dapat diidentifikasi dalam lingkaran pemakaian dan sesuatu yang penting bagi pemakai dalam konteks sistem yang akan dibuat.

2. Atribut

   Elemen dari sebuah entity yang berfungsi mendeskripsikan karakter entity.

3. Hubungan

   Sebagaimana halnya entity maka dalam hubungan pun harus dibedakan antara hubungan atau bentuk hubungan antara entity dengan isi dari hubungan itu sendiri.

Beberapa macam hubungan antar relasi, antara lain:

a. Satu Ke Satu (*One to One*)

   Bentuk relasi antara satu entitas dengan sejumlah satu ke entitas dengan jumlah yang sama.

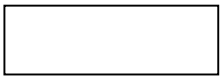
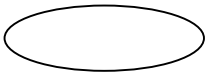
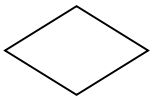
b. Satu Ke Banyak (*One to Many*)

   Bentuk relasi dari entitas dengan jumlah satu ke entitas lain yang berjumlah lebih dari satu (Entitas dengan banyak alternatif tujuan).

c. Banyak Ke Banyak (*Many to Many*)

   Bentuk relasi yang mendeskripsikan permasalahan yang komplek yaitu hubungan antara entitas yang berjumlah lebih dari satu dengan entitas yang sama.

Tabel 2.1. Tabel Notasi yang digunakan pada *Entity Relational Diagram* (ERD)

No	Nama simbol	Fungsi	simbol
1	persegi panjang	menyatakan suatu entitas	
2	Elips	menyatakan atribut	
3	belah ketupat	menyatakan relasi	
4	Garis	menyatakan hubungan antar entitas dengan relasi atau hubungan antar entitas dengan relasi	

2.2.6. Pengertian Metodologi Berorientasi Objek

Metodologi Berorientasi Objek atau *object oriented* merupakan suatu strategi pembangunan atau paradigma baru dalam rekayasa perangkat lunak yang memandang sistem sebagai kumpulan obyek-obyek diskrit yang saling berinteraksi. Yang dimaksud berorientasi objek adalah bahwa mengorganisasikan perangkat lunak sebagai kumpulan obyek-obyek diskrit yang bekerja sama antara informasi atau struktur data dan perilaku (*behavior*) yang mengaturnya. Sebuah sistem yang dibangun dengan berdasarkan metode berorientasi objek adalah sebuah sistem yang komponennya dibungkus (dienkapsulasi) menjadi kelompok data dan fungsi. Setiap komponen dalam sistem tersebut dapat mewarisi atribut dan sifat dan komponen lainnya serta dapat berinteraksi satu sama lainnya.

2.2.7. Karakteristik Sistem Berorientasi Objek

Karakteristik yang dimiliki sebuah sistem berorientasi objek antara lain (Munawar, 2005):

1. Abstraksi (*Abstraction*) adalan prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan. Atau secara sederhana dikatakan sebagai proses memilah beberapa attribut dan beberapa operasi suatu objek hanya

sampai pada yang benar-benar diperlukan saja, dan membuang attribut dan operasi yang tidak diperlukan untuk persoalan yang dihadapi.

2. Pembungkusan (*Encapsulation*) yaitu pembungkusan atribut data dan layanan (operasi-operasi) yang dipunyai objek, untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerja-nya. Dengan kata lain, menyembunyikan kompleksitas dari luar dan hanya membuka operasi-operasi yang diperlukan saja terhadap objek-objek lain.
3. Pewarisan (*Inheritance*) merupakan mekanisme yang memungkinkan suatu kelas mewarisi sebagian atau seluruh definisi ke objek lain sebagai bagian dari dirinya.
4. Pengiriman Pesan (*message sending*) adalah komunikasi antar objek dilakukan lewat pesan (*message*) yang dikirim dari satu objek ke objek lainnya.
5. Banyak Bentuk (*Polymorphism*) ialah Kemampuan suatu objek untuk digunakan di banyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program. Atau dengan kata lain, *polymorphism* yaitu suatu operasi dengan nama yang sama, tetapi jika diberikan pada objek yang berbeda akan mengakibatan operasi yang berbeda.
6. Assosiasi (*Association*) adalah Hubungan antar objek yang saling membutuhkan. Hubungan ini bisa satu arah ataupun lebih dari satu arah. Selain dengan sesama objek, sebuah class bias juga berasosiasi dengan lebih dari satu class. *Multiplicity* adalah sebuah aspek yang penting dalam asosiasi antar objek. Ini berkaitan dengan sejumlah objek dalam satu class yang saling berasosiasi.
7. Aggregasi (*Aggregation*) Adalah bentuk khusus dari assosiasi yang menggambarkan seluruh bagian suatu objek merupakan bagian dari objek yang lain.Salah satu bentuk aggregasi yang melibatkan hubungan yang sangat erat antar objek dan komponen objek disebut

composition. Kata kunci *composition* adalah komponen itu hanya bisa ada sebagai komponen pada objek *composition*.

2.2.8. Konsep Dasar Pemodelan Berorientasi Objek

Terdapat beberapa konsep penting dalam pemodelan sistem informasi menggunakan pendekatan objek atau berorientasi objek, diantaranya:

1. Aktor (*Actor*)

   Dalam pemodelan sistem dengan UML, aktor adalah seseorang atau sesuatu yang berinteraksi dengan sistem yang sedang dikembangkan. Artinya, seorang aktor dapat mengirim atau menerima pesan dari dan ke sistem atau merubah informasi yang dibutuhkan sistem. Aktor adalah sebuah class bukan sebagai objek. Komunikasi antara aktor dan sistem dilakukan dengan cara mengirim dan menerima pesan. Primary actor adalah seseorang atau sesuatu yang menggunakan fungsi utama dari sistem, sedangkan secondary actor adalah seseorang atau sesuatu yang menggunakan fungsi sekunder dari sistem.Aktor dalam use case dilambangkan dengan gambar sebagai berikut :



Gambar 2.7. Simbol Simbol Aktor (Whitten, 2004)

Actor terbagi pada empat macam tipe (Whitten, 2004):

a. *Primary business actor* (Pelaku Bisnis Utama) *Stakeholder* yang terutama endapatkan keuntungan dari kegiatan transaksidengan menerima nilai yang terukur dan terobservasi.
b. *Primary system actor* (Pelaku Sistem Utama) *Stakeholder* yang secara langsung dapat berhadapan dengan sistem untuk menginisiasi atau memicu kegiatan atau sistem. Pelaku sistem utama dapat berinteraksi dengan para pelaku bisnis utama untuk menggunakan sistemaktual. Mereka memfasilitasi kejadian dengan menggunakan sistem secara

langsung demi mencapai keuntungan para pelaku bisnis utama.

c. *External server actor* (Pelaku server eksternal. *Stakeholder* melayani kebutuhan user.
d. *External receiving actor* (Pelaku penerima eksternal) *Stakeholder* yang bukan pelaku utama tetapi menerima output dari kegiatan transaksi.

2. Obyek (*Object*)

Sebuah objek adalah sebuah penggambaran komputer dari beberapa kejadian di dunia nyata.Objek dapat berupa benda konkrit maupun hal-hal yang abstrak. Objek bisa berupa orang, tempat, kejadian, benda atau konsep-konsep lain yang ada di dunia nyata. Setiap objek memiliki identitas, keadaan dan perilaku. Identitas adalah sesuatu yang membedakan suatu objek dari objek lainnya sehingga masing-masing objek dapat dibedakan Setiap objek juga memiliki keadaan. Keadaan dari sebuah objek adalah kondisi objek tersebut atau himpunan dari yang menggambarkan objek tersebut. keadaan dinyatakan oleh nilai dari atribut objeknya. Atribut adalah nilai internal suatu objek yang mencerminkan antara lain karakteristik objek, kondisi sesaat, koneksi dengan objek lain, dan identitas.

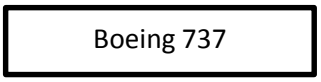
Selain memiliki identitas dan keadaan, setiap objek memiliki perilaku yang mendefinisikan bagaimana sebuah objek bertindak dan memberi reaksi. Perilaku sebuah objek dicerminkan oleh antarmuka dan metode dari objek tersebut. Objek-objek juga dapat saling berhubungan atau berinteraksi dengan objek yang lainnya. Hal ini dimungkinkan karena setiap objek memiliki antarmuka yang memfasilitasi terjadinya interaksi antar objek tersebut.

Contoh dari objek :

a. Objek orang : saya, anda, kita dan lain-lain.
b. Objek tempat : kampus, gedung, komputer dan lain-lain.
c. Objek kejadian : kuliah, survei, pendaftaran dan lain-lain.

3. Kelas (*Class*)

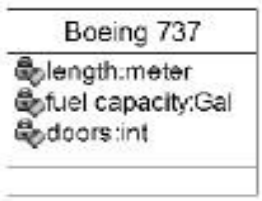
Kelas adalah kumpulan dari objek yang didalamnya terdapat atribut dan operasi yang serupa. Dengan penggolongan objek-objek dalam suatu kelas kita bisa melakukan abstraksi masalah. Atribut dan nama kelas untuk beberapa objek yang sejenis dapat dituliskan sekali saja begitu juga dengan fungsi dan metode yang sama cukup dituliskan satu kali saja dan bisa digunakan ulang oleh objek yang termasuk kedalam kelas yang sama.



Gambar 2.8. Simbol Kelas (Bahrami, 1999).

4. Attribut

Atribut adalah *property* dari sebuah kelas. Atribut ini menggambarkan batas nilai yang mungkin ada pada objek dari kelas. Sebuah kelas mungkin akan mempunyai nol atau lebih atribut. Secara konvensi, jika nama atribut terdiri dari satu kata, maka ditulis dengan huruf kecil. Akan tetapi jika nama atribut mengandung lebih dari suku kata maka semua suku kata digabungkan dengan kata pertama menggunakan huruf kecil dan awal suku kata berikutnya menggunakan huruf besar.



Gambar 2.9. Contoh Kelas dan Atribut-atributnya (Bahrami, 1999)

5. Operasi (*Operator*)

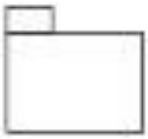
Operasi adalah fungsi yang dapat diaplikasikan ke atau oleh suatu objek dalam kelas. Misalnya kelas window memiliki fungsi close, cancel, open dan lain-lain. Operasi yang sama dapat diterapkan pada kelas yang berbeda.



Gambar 2.10. Contoh Kelas dan Operasi-operasinya (Bahrami, 1999)

6. Paket (*Package*)

Paket adalah pengelompokan untuk menandakan kelompok suatu elemen model. Paket digunakan untuk mempermudah mengorganisasi elemen-elemen model. Sebuah paket dapat mengandung beberapa paket kelas lain didalamnya.

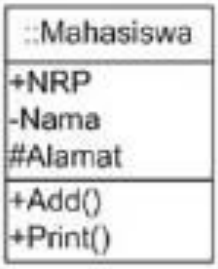


Gambar 2.11. Simbol Paket (Bahrami, 1999)

7. Kelas Obyek (*Object Class*)

Kelas terdiri dari beberapa objek yang memiliki atribut, operasi, semantik dan relationship yang sama. Kelas objek menggambarkan abstraksi dari suatu objek dalam implementasi. Dalam kelas objek terdapat istilah Derajat Pembungkusan (*Visibility*) operasi atau atribut yang terdiri dari (Munawar, 2005).

a. *Protected* : operasi atau atribut yang tampak hanya oleh kelas itu sendiri, subkelas atau teman kelas tersebut. Visibility merupakan default dari sebuah operasi yang kita buat. Visibility ini melindungi operasi dari penggunaan oleh kelas-kelas luar. Protected digambarkan dengan tanda (#).
b. *Public* : operasi atau atribut dapat digunakan oleh kelas lain yangberhubungan dengan kelas tersebut. Digambarkan dengan tanda (+).
c. *Private* : operasi atau atribut yang hanya bisa digunakan oleh kelas itu sendiri. Digambarkan dengan tanda (-).

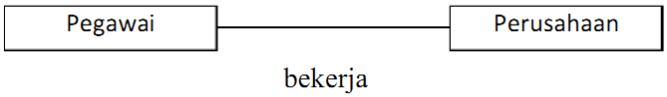


Gambar 2.12. Contoh Kelas Mahasiswa dengan *Visibility*
(Munawar, 2005)

8. Asosiasi

Asosiasi adalah kelas-kelas yang terubungkan satu sama lain secara konseptual. Untuk menunjukan asosiasi ini bisa digunakan segitiga yang diarsir pada arah yang dituju.

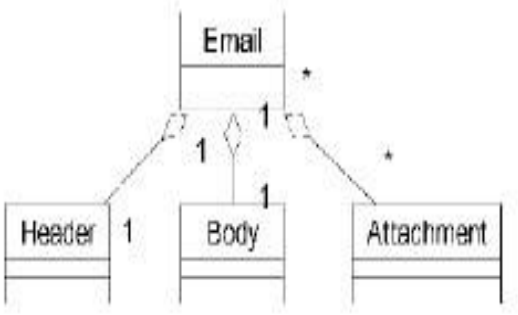
Asosiasi bisa digunakan untuk memodelkan relasi diantara objek-objek. Nama dari asosiasi memberi informasi secara tepat tentang relasi apa yang sedang dimodelkan. Di UML, ada relasi dengan perlakuan khusus yang disebut dengan 'bagian dari (*part of*)' yang menangani antar objek-objek dimana salah satunya adalah bagian dari yang lain. Dengan kata lain sebuah objek terdiri atas objek- objek yang lain.



Gambar 2.13. Asosiasi Antara Pegawai dengan Perusahaan
(Munawar,2005)

9. Agregasi

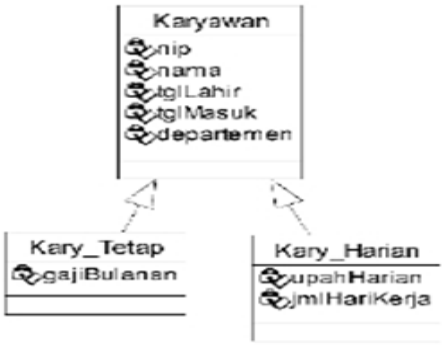
Agregasi adalah terminology yang digunakan di UML untuk menjelaskan hal tersebut. Sebuah agregasi adalah kasus khusus dari asosiasi. Hal ini disimbolkan dengan jajaran genjang yang diletakkan pada kelas yang mengandung objek.



Gambar 2.14. Contoh Agregasi (Munawar, 2005)

10. Generalisasi

Generalisasi digunakan untuk menjelaskan hubungan kesamaan diantara kelas. Objek-objek kelas bisa diatur secara hierarkis. Manfaat yang bisa diambil dengan cara ini diantaranya bisa menampilkan aspek-aspek perbedaan dari kondisi nyata yang ingin dipahami. Dengan menggunakan generalisasi bisa dibangun struktur logis yang bisa menampilkan derajat kesamaan atau perbedaan diantara kelas-kelas.



Gambar 2.15. Contoh Generalisasi (Munawar, 2005)

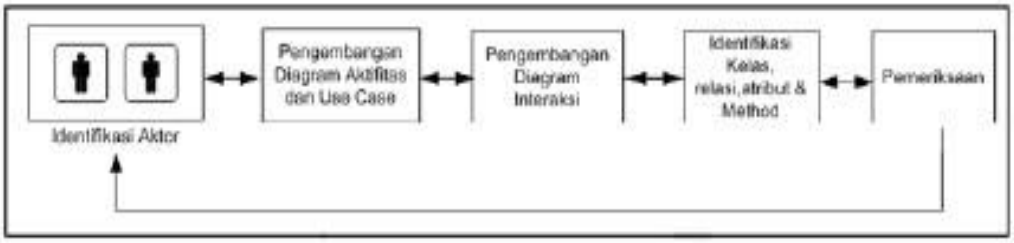
2.2.9. Metode Analisis Berorientasi Objek (OOA-*Object Oriented Analysis*)

2.2.9.1. *Unified Approach*(UA)

Untuk melakukan analisis sistem terdapat beberapa macam pendekatan, diantaranya pendekatan konvensional dan pendekatan berorientasi objek. Pendekatan konvensional terutama mengacu kepada strategi dekomposisi yang berdasar algoritma atau fungsional. Pendekatan ini telah berkembang meliputi seluruh tahap atau aktifitas proses rekayasa perangkat lunak dari mulai pemrograman dengan iterasi perbaikan, pemrograman terstruktur, ditambah dengan perancangan terstruktur kemudian analisis terstruktur dan sebagainya. Sedangkan pendekatan berorientasi objek memusatkan pada rancangan pada objek dan antar muka yang dihasilkan. Objek adalah entiti yang berisi data atau variabel dan tingkah laku. Data atau variabel yang menggambarkan sifat atau keadaan objek dalam dunia nyata (*real world*) didefiniskan sebagai attribute, sedangkan tingkah laku yang menggambarkan aksi-aksi yang dimiliki objek didefinisikan sebagai *method*.

Unified Approach (UA) merupakan metode analisis berorientasi objek dari Ali Bahrami (1999). UA adalah suatu metodologi pengembangan sistem berbasis objek yang menggabungkan proses dan metodologi yang telah ada sebelumnya dan menggunakan UML sebagai standar pemodelannya. Proses dan tahapan yang ada dalam UA merupakan proses-proses terbaik yang diambil dari metode objek yang telah diperkenalkan oleh Booch, Rumbaugh, dan Jacobson. Tahap Analisis dalam UA ditujukan untuk mengidentifikasi kelas-kelas yang terdapat dalam sistem.

Analisis berorientasi objek dengan pendekatan UA (*Unified Approach*) dari Ali Bahrami digambarkan dalam bagan berikut:



Gambar 2.16. Tahap Analisis *Unified Approach* (UA) (Bahrami, 1999)

Keterangan:

1. Identifikasi *actor*

   Identifikasi aktor adalah tahap pertama yang penting dalam OOA. Istilah aktor merepresentasikan peran dari seorang user terhadap sistem. Kandidat aktor dapat ditemukan dengan mencari tahu siapa yang akan menggunakan sistem dan apa yang dilakukan aktor terhadap sistem.

2. Pengembangan Diagram Aktifitas dan Diagram Use Case

   Pada tahap ini akan digambarkan model aktifitas bisnis menggunakan diagramaktifitas UML untuk menggambarkan kinerja sistem. Dalam diagram aktifitas akan digambarkan alur kerja dari sistem. Dengan mengetahui alur kerja sistem yang ada, dapat dilakukan pemodelan diagram use case untuk menggambarkan interkasi user terhadap sistem.

3. Mengembangkan diagram interaksi

   Salah satu dari diagram interaksi adalah sequence diagram. Sequence diagram adalah suatu model untuk menggambarkan interaksi antar objek dalam sistem. Interaksi yang dilakukan oleh objek-objek tersebut dilakukan dengan cara satu objek mengirimkan pesan (message) kepada objek lain. Dalam tahap ini akan ditentukan rangkaian diagram aktifitas sistem yang sedang berjalan.

4. Identifikasi kelas

Dari sequence diagram akan terlihat kelas-kelas apa saja yang ada dalam sistem. Pada tahap ini dilakukan proses identifikasi kelas-kelas, relationship, atribut serta metode-metode yang digunakan pada setiap kelas.

5. Pemeriksaan terhadap tahap sebelumnya.

Proses pemeriksaan terhadap hasil akhir tahap analisis. Bila terdapat kesalahan maka kembali ke tahap awal analisis bila hasilnya benar maka tahap analisis selesai.

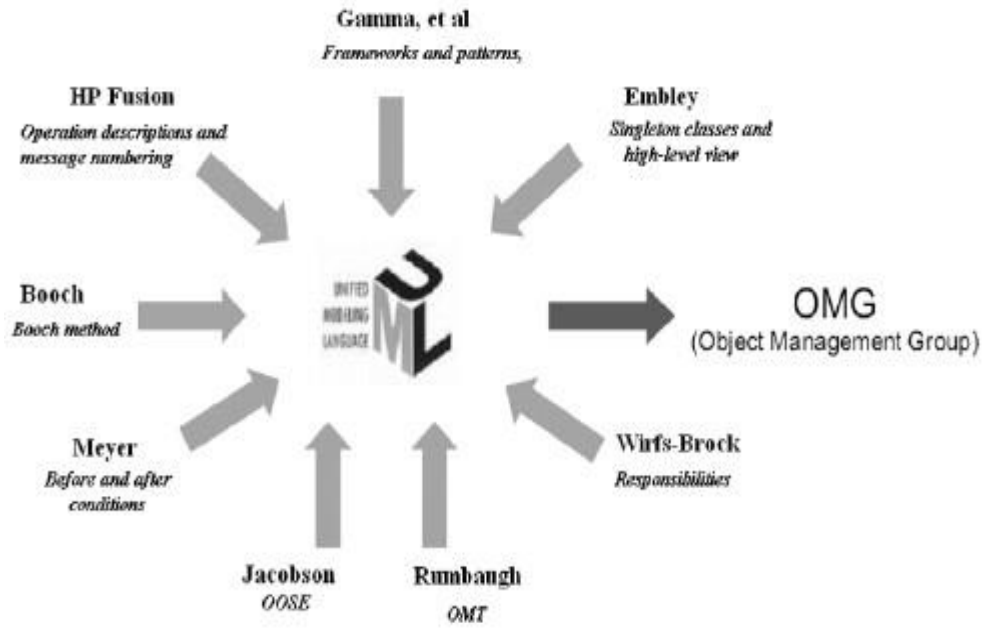
2.2.10. Pemodelan Dalam View UML *(Unified Modelling Language)*

2.2.10.1. Definisi UML *(Unified Modelling Language)*

UMl (*Unified Modelling Language*) adalah salah satu alat bantu yang sangat handal di dunia pengembangan system yang berorientasi obyek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembangan system untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka dengan yang lain.

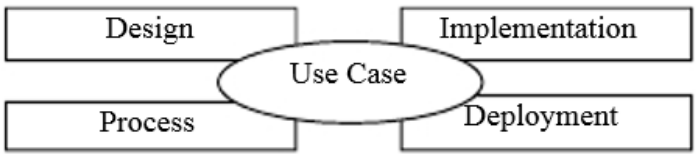
Uml merupakan kesatuan dari bahasa pemodelan yang dikembangkan oleh Booch, *Object Modelling Technique* (OMT) dan *Object Oriented Software Engeenering* (OOSE). Metode Booch dari Grady Booch sangat terkenal dengan nama metode *Design Object Oriented.* Metode ini menjadikan proses analisis dan design ke dalam empat tahapan iteratif yaitu : identifikasi kelas-kelas dan obyek-obyek, identifikasi semantic dari hubungan obyek dan kelas tersebut, perincian interface dan implementasi. Keunggulan metode Booch adalah pada detil dan kayanya dengan notasi dan elemen. Pemodelan OMT yang

dikembangkan oleh Rumbaugh didsarkan pada analisis tersetruktur dan pemodelan entitiy-Relationship.



Gambar 2.17. Unsur-unsur pembentuk UML (Munawar, 2005).

UML dibangun atas model 4+1 view. Model ini didasarkan pada fakta bahwa struktur sebuah sistem dideskripsikan dalam 5 view dimana salah satu diantaranya *use case view*. *Use case view* ini memegang peran untuk mengintegrasikan content ke view yang lain.



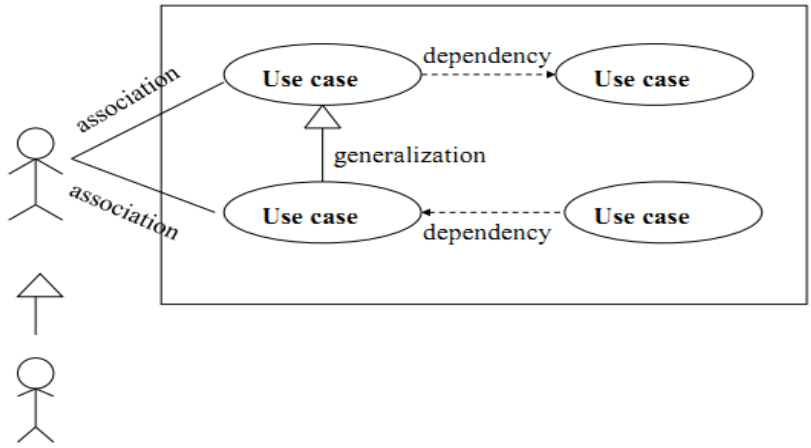
Gambar 2.18. Model 4+1 *View* (Munawar, 2005)

2.2.10.2. *Use Case Diagrams*

Konsep *Use case* diperkenalkan oleh ivan Jacobson di dalam *Object Oriented Software Engineering* (OOSE). Kemampuan suatu sistem

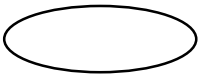
menguraikan sejumlah *use case* yang berbeda, masing-masing menunjukkan secara spesifik suatu arus kejadian yang spesifik di dalam sistem.

Use case adalah deskripsi fungsi dari sebuah sistem dari prespektif pengguna. *Use case* akan menggambarkan cara kerja suatu software dengan aktor. Dalam *use case* diagram akan digambarkan hubungan antara aktor dengan *use case*. Aktor adalah orang atau subsistem lain yang akan berinteraksi dengan sistem. Sementara use case menggambarkan proses yang akan dilakukan oleh aktor terhadap system.



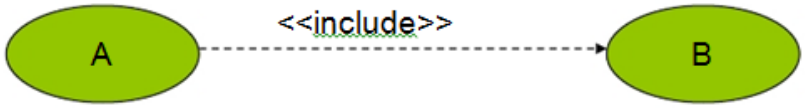
Gambar 2.19. *Overview Use case*

Tabel 2.2. Tabel Notasi yang digunakan pada diagram Use Case

No	Nama simbol	Fungsi	simbol
1	*Actor*	Pihak yang mengakses Use case	
2	*Use Case*	Mewakili apa yang system bias lakukan	

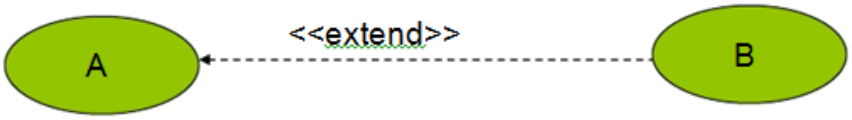
3	*Assosiation*	Merelasikan aktor dengan use case	———————
4	*System Boundary*	Menggambarkan batasan system terhadap lingkungannya.	system

a. *Include Dependency* merupakan satu use case bisa meng-include use case lainnya, Jika use case A meng-include use case B maka use case B harus diimplementasi setiap kali use case A dipanggil. Direpresentasikan dengan garis panah putus-putus bertuliskan <<*include*>> ke arah use case yang diinclude.



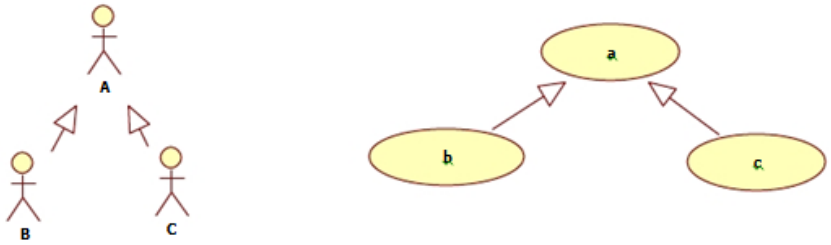
Gambar 2.20. *Include Dependency* pada *Use case*

b. *Extends Depedency* merupakan satu use case bisa di-extend oleh use case lain, Jika use case A di-*extend* oleh use case B maka antara A dan B dapat diimplementasi dan digunakan secara bebas. Direpresentasikan dengan garis panah putus-putus bertuliskan <<*extends*>> A tidak harus selalu memanggil B dalam beberapa batasan.



Gambar 2.21. *Extends Dependency* pada *Use case*

c. *Generalization Depedency* yoitu aktor dan use case bisa digeneralisasi. Generalisasi digunakan untuk membuat aktor atau use case yang lebih spesifik dari suatu aktor dan use case.



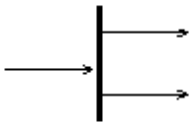
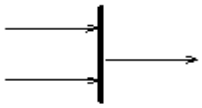
Gambar 2.22. *Generalization Dependency* pada *Use case*

2.2.10.3. *Activity Diagrams*

Activity diagram menggambarkan berbagai alur aktifitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, keputusan yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

Activity diagram merupakan state diagram khusus, dimana sebagian besar state adalah aksi dan sebagian besar transisi di-trigger oleh selesainya state sebelumya (internal processing). Oleh karena itu, Activity diagram tidak menggambarkan behavior internal sebuah sistem (dan interaksi antar sub sistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktifitas dari level atas secara umum. Sebuah aktivitas dapat direalisasikan oleh satu use case atau lebih. Aktifitas menggambarkan proses yang berjalan, sementara use case menggambarkan bagaimana actor mengguanakan sistem untuk melakukan aktifitas.

Tabel 2.3. Tabel Notasi yang digunakan pada diagram *Activity*

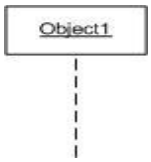
No	Nama simbol	Fungsi	simbol
1	*Start Point / Start state*	*Start state* dengan tegas menunjukkan dimulainya suatu *workflow* pada sebuah *activity* diagram	
2	*End Point/ End state*	*End state* menggambarkan akhir atau terminal dari pada sebuah *activity* diagram.	
3	*Activities*	*Activity* menggambarkan sebuah pekerjaan/tugas dalam *workflow*.	
4	*Fork*	Untuk percabangan	
5	*Join*	Untuk penggabungan	
6	*Decision*	Yang mengindikasikan suatu kondisi dimana ada kemungkinan perbedaan transisi.	
7	*State Transition*	*State transition* menunjukkan kegiatan apa berikutnya setelah suatu kegiatan sebelumnya.	
8	*Swimlane*	Object swimlane untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu	

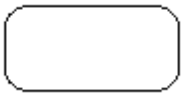
2.2.10.4. *Sequence Diagram*

Sequence diagram menggambarkan interaksi antar objek didalam dan disekitar sistem (termasuk pengguna, display dan sebagainya) berupa message yang digambarkan terhadap waktu. Sequence diagram terdiri atas dimensi vertika (waktu) dan dimensi horizontal (objek-objek yang terkait).

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respon dari sebuah event untuk menghasilkan output tertentu. Sequence diagram terdiri dari sumbu vertikal putus-putus yang merepresentasikan "*lifetime*" objek dan sumbu horizontal yang menunjukan sekumpulan objek yang saling berinteraksi dalam sistem. Diagram ini menjelaskan bagaimana objek berinteraksi dengan objek yang lainnya yaitudengan cara mengirim dan menerima pesan. Komunikasi antar objek tersebut ditandai dengan garis horizontal yang disertai dengan nama operasinya.

Tabel 2.4. Tabel Notasi yang digunakan pada diagram *Sequence*

No	Nama simbol	Fungsi	simbol
1	*Object*	*Object* merupakan instance dari sebuah *class* dan dituliskan tersusun secara horizontal.	
2	*Actor*	*Actor* juga dapat berkomunikasi dengan objek, maka *actor* juga dapat diurutkan sebagai kolom.	
3	*Lifeline*	*Lifeline* mengindikasikan keberadaan sebuah objek dalam basis waktu.	

4	*Activation*	*Activation* mengindikasikan sebuah objek yang akan melakukan sebuah aksi.	
5	*Message*	*Message* mengindikasikan komunikasi antara objek-objek.	

2.2.10.5. *Class Diagrams*

Class Diagram, juga dikenal sebagai objek *modeling*, adalah diagram analisis statis yang utama. Diagram ini menunjukkan struktur yang statis dari suatu model. Suatu class diagram adalah suatu koleksi unsur-unsur modeling yang statis, seperti kelas-kelas dan relationship yang dihubungkan sebagai suatu grafik antara yang satu dengan yang lainnya beserta isi-isinya. Sebagai contoh, hal yang ada (seperti kelas-kelas), struktur-struktur class diagram internal, dan hubungan class diagram dengan kelas-kelas yang lain. Class diagram tidak menunjukkan informasi yang temporal, yang diperlukan di dalam pemodelan yang dinamis.

Class diagram memodelkan struktur kelas dan isinya dengan menggunakan elemen-elemen model seperti *class*, *package*, dan objek. Kelas terdiri dari tiga bagian yaitu nama kelas, *attribut* dan *operations*. Kelas didefinisikan secara global dapat diakses oleh objek diluar kelas tersebut.

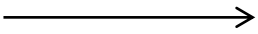
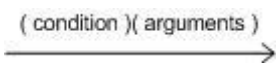
2.2.10.6. *State Chart Diagrams*

Statechart diagram menggambarkan transisi dan perubahan keadaan (dari satu *state* ke *state* lainnya) suatu objek pada sistem sebagai akibat dari *stimuli* yang diterima. Pada umumnya *statechart diagram* menggambarkan *class* tertentu (satu *class* dapat memiliki lebih dari satu *statechart diagram*).

Dalam UML, *state* digambarkan berbentuk segiempat dengan sudut membulat dan memiliki nama sesuai kondisinya saat itu. Transisi antar *state*

umumnya memiliki kondisi *guard* yang merupakan syarat terjadinya transisi yang bersangkutan, dituliskan dalam kurung siku. *Action* yang dilakukan sebagai akibat dari *event* tertentu dituliskan dengan diawali garis miring.

Tabel 2.5. Tabel Notasi yang digunakan pada diagram *State Chart*

No	Nama simbol	Fungsi	simbol
1	*Start Point / Start state*	*Start state* dengan tegas menunjukkan dimulainya suatu *workflow* pada sebuah *activity* diagram	
2	*End Point/ End state*	*End state* menggambarkan akhir atau terminal dari pada sebuah *activity* diagram.	
3	*Activities*	*Activity* menggambarkan sebuah pekerjaan/tugas dalam *workflow*.	
6	*Guard*	*Guard*, yang merupakan syarat terjadinya transisi yang bersangkutan.	[guard]
7	*State Transition*	*State transition* menunjukkan kegiatan apa berikutnya setelah suatu kegiatan sebelumnya.	
8	*Event*	*Event*, digunakan untuk mendeskripsikan kondisi yang menyebabkan sesuatu pada state.	

2.2.11. Pengolahan Data

Pengelolaan data adalah serangkaian operasi atas informasi yang direncanakan guna mencapai tujuan atau hasil yang diinginkan.

Tujuan utama dalam pengolahan data dalam sebuah database adalah agar dapat memperoleh kembali data yang ingin dicari dengan mudah dan cepat, selain itu pemanfaatan database memiliki beberapa tujuan. Secara lengkap pemanfaatan database memliki beberapa tujuan dan dilakukan untuk memenuhi sejumlah tujuan (objektif) antara lain:

a. Kecepatan dan Kemudahan (*Speed*)
b. Efesiensi ruang (*Space*)
c. Ketersediaan (*Availability*)
d. Kelengkapan (*Completely*)
e. Keamanan (*Security*)
f. Kebersamaan Pemakai (*Sharability*)

2.2.12. Teknologi informasi

Teknologi Informasi biasa disebut TI, IT (*Information Technology*) atau Infotech. Berbagai definisi teknologi informasi telah diutarakan oleh beberapa ahli, diantaranya:

a. Haag den Keen (1996), Teknologi Informasi adalah seperangkat alat yang membantu Anda bekerja dengan informasi dan melakukan tugas-tugas yang berhubungan dengan pemrosesan informasi.
b. Martin (1999), Teknologi Informasi tidak hanya terbatas pada teknologi komputer (perangkat keras atau lunak) yang digunakan untuk memproses dan menyimpan informasi, melainkan juga mencakup teknologi komunikasi untuk mengirimkan informasi.
c. Williams dan Swayer (2003), Teknologi Informasi adalah teknologi yang menggabungkan komputasi (komputer) dengan jalur komunikasi berkecepatan tinggi yang membawa data, suara dan video.

Dari definisi diatas terlihat bahwa teknologi informasi baik secara implisit maupun eksplisit tidak sekedar berupa tekologi komputer, tetapi juga teknologi telekomunikasi. Dengan kata lain, yang disebut teknologi informasi adalah gabungan antara teknologi komputer dan telekomunikasi.

2.2.13. Rekayasa Perangkat Lunak (RPL)

Istilah Rekayasa Perangkat Lunak (RPL) secara umum disepakati sebagai terjemahan dari istilah Software Engineering. Istilah Software Engineering mulai dipopulerkan tahun 1968 pada Software Engineering Conference yang diselenggarakan oleh NATO. Sebagian orang mengartikan RPL hanya sebatas pada bagaimana membuat program komputer. Padahal ada perbedaan yang mendasar antara perangkat lunak (*software*) dan program komputer.

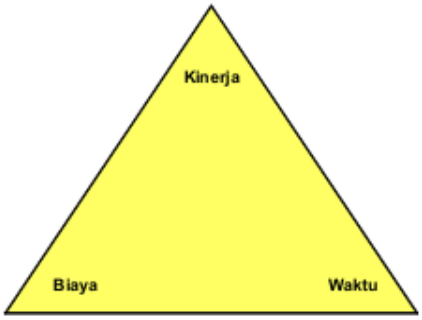
Perangkat lunak adalah seluruh perintah yang digunakan untuk memproses informasi. Perangkat lunak dapat berupa program atau prosedur. Program adalah kumpulan perintah yang dimengerti oleh komputer sedangkan prosedur adalah perintah yang dibutuhkan oleh pengguna dalam memproses informasi (O'Brien, 1999). Pengertian RPL sendiri adalah sebagai berikut:

"Suatu disiplin ilmu yang membahas semua aspek produksi perangkat lunak, mulai dari tahap awal yaitu analisa kebutuhan pengguna, menentukan spesifikasi dari kebutuhan pengguna, disain, pengkodean, pengujian sampai pemeliharaan sistem setelah digunakan."

Jelaslah bahwa RPL tidak hanya be rhubungan dengan cara pembuatan program komputer. Pernyataan " semua aspek produksi" pada pengertian di atas, mempunyai arti semua hal yang berhubungan dengan proses produksi seperti manajemen proyek, penentuan personil, anggaran biaya, metode, jadwal, kualitas sampai dengan pelatihan pengguna merupakan bagian dari RPL.

1. Tujuan Rekayasa Perangkat Lunak

Secara umum tujuan RPL tidak berbeda dengan bidang rekayasa yang lain. Mari kita perhatikan Gambar berikut ini.



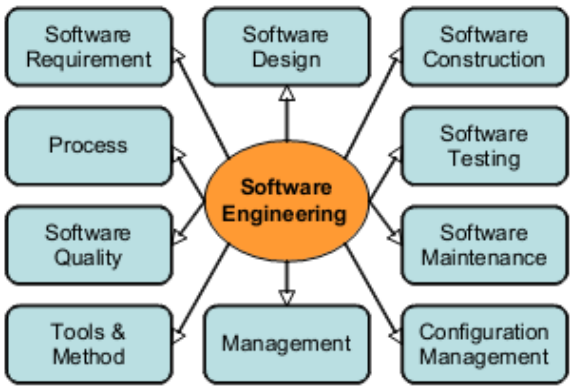
Gambar 2.23. Tujuan RPL

Dari Gambar diatas dapat diartikan bahwa bidang rekayasa akan selalu berusaha menghasilkan output yang kinerjanya tinggi, biaya rendah dan waktu penyelesaian yang tepat. Secara lebih khusus kita dapat menyatakan tujuan RPL adalah:

a. Memperoleh biaya produksi perangkat lunak yang rendah.
b. Menghasilkan perangkat lunak yang ki nerjanya tinggi, andal dan tepat waktu.
c. Menghasilkan perangkat lunak yang dapat bekerja pada berbagai jenis platform.
d. Menghasilkan perangkat lunak yang biaya perawatannya rendah.

2. Ruang Lingkup

Sesuai definisi yang telah disampaikan sebelumnya, maka ruang lingkup RPL dapat digambarkan sebagai berikut.



Gambar 2.24. Ruang Lingkup RPL

a. *Software requirements* berhubungan dengan spesifikasi kebutuhan dan persyaratan perangkat lunak.
b. *Software design* mencakup proses penentuan arsitektur, komponen, antarmuka, dan karakteristik lain dari perangkat lunak.
c. *Software construction* berhubungan dengan detil pengembangan perangkat lunak, termasuk algoritma, pengkodean, pengujian, dan pencarian kesalahan.
d. *Software testing* meliputi pengujian pada keseluruhan perilaku perangkat lunak.
e. *Software maintenance* mencakup upaya-upaya perawatan ketika perangkat lunak telah dioperasikan.
f. *Software configuration* management berhubungan dengan usaha perubahan konfigurasi perangkat lunak untuk memenuhi kebutuhan tertentu.
g. *Software engineering* management berkaitan dengan pengelolaan dan pengukuran RPL, termasuk perencanaan proyek perangkat lunak.
h. *Software engineering tools and methods* mencakup kajian teoritis tentang alat bantu dan metode RPL.
i. *Software engineering process* berhubungan dengan definisi, implementasi, pengukuran, pengelolaan, perubaha n dan perbaikan proses RPL.
j. *Software quality* menitikberatkan pada kualitas dan daur hidup perangkat lunak.

2.2.14. Pengertian SQL

SQL (*Structured Query Language*) adalah sebuah bahasa yang dipergunakan untuk mengakses data dalam basis data relasional. Bahasa ini secara *de facto* merupakan bahasa standar yang digunakan dalam manajemen basis data relasional. Saat ini hampir semua server basis data yang ada mendukung bahasa ini untuk melakukan manajemen datanya.

Secara umum, SQL terdiri dari dua bahasa, yaitu *Data Definition Language* (DDL) dan *Data Manipulation Language* (DML). Implementasi DDL dan DML berbeda untuk tiap sistem manajemen basis data (SMBD), namun secara umum implementasi tiap bahasa ini memiliki bentuk standar yang

ditetapkan ANSI. Artikel ini akan menggunakan bentuk paling umum yang dapat digunakan pada kebanyakan SMBD.

Data Definition Language (DDL). Digunakan untuk mendefinisikan, mengubah, serta menghapus basis data dan objek-objek yang diperlukan dalam basis data, misalnya tabel, view, user, dan sebagainya. Secara umum, DDL yang digunakan adalah CREATE untuk membuat objek baru, USE untuk menggunakan objek, ALTER untuk mengubah objek yang sudah ada, dan DROP untuk menghapus objek. DDL biasanya digunakan oleh administrator basis data dalam pembuatan sebuah aplikasi basis data.

Data Manipulation Language (DML). Digunakan untuk memanipulasi data yang ada dalam suatu tabel. Perintah yang umum dilakukan adalah:

a) SELECT untuk menampilkan data
b) INSERT untuk menambahkan data baru
c) UPDATE untuk mengubah data yang sudah ada
d) DELETE untuk menghapus data

MySQL adalah database yang cepat dan tangguh, sangat cocok jika digabungkan dengan PHP, pengunaan database dapat dipakai untuk menyimpan, mencari dan mengklasifikasikan data dengan lebih akurat dan professional. MySQL menggunakan SQL language (*Structured Query Language*) artinya MySQL menggunakan query atau bahasa pemprogaman yang sudah standar di dalam dunia database. Kelebihan MySQL diantaranya:

1. Dari segi performa, MySQL tidak bisa diragukan, pemmrosesan database sangat cepat
2. Open source
3. Mudah untuk dipelajari
4. Kompabilitas dengan berbagai sistem operasi dan *web server* yang ada

Keunggulan MySQL

Pertama, MySQL tersedia di berbagai platform Linux dan berbagai varian Unix. Sesuatu yang tidak dimiliki *Access*, misalnya—padahal Access amat popular di *platform Windows*. Banyak server Web berbasiskan Unix, ini

menjadikan *Access* otomatis tidak dapat dipakai karena ia pun tidak memiliki kemampuan *client-server/networking*.

Kedua, fitur-fitur yang dimiliki MySQL memang yang biasanya banyak dibutuhkan dalam aplikasi Web. Misalnya, klausa *LIMIT SQL*-nya, praktis untuk melakukan paging. Atau jenis indeks *field FULLTEXT*, untuk full text searching. Atau sebutlah kekayaaan fungsi-fungsi builtinnya, mulai dari memformat dan memanipulasi tanggal, mengolah string, regex, enkripsi dan hashing. Yang terakhir misalnya, praktis untuk melakukan penyimpanan password anggota situs.

Ketiga, MySQL memiliki overhead koneksi yang rendah. Soal kecepatan melakukan transaksi atau kinerja di kondisi load tinggi mungkin bisa diperdebatkan dengan berbagai benchmark berbeda, tapi kalau soal yang satu ini MySQL-lah juaranya. Karakteristik ini membuat MySQL cocok bekerja dengan aplikasi CGI, di mana di setiap *request* skrip akan melakukan koneksi, mengirimkan satu atau lebih perintah SQL, lalu memutuskan koneksi lagi.

PHP & MySQL adalah sebuah pasangan yang sangat cocok dan dapat berjalan di hampir semua sistem operasi dan *web server* yang ada. Alasan lain mengapa memilih PHP & MySQL adalah karena perkembangannya yang sangat cepat apalagi PHP & MySQL adalah bahasa pemprogaman yang bersifat open source, setiap waktu selalu muncul fungsi-fungsi tambahan yang dapat meningkatkan performa keduanya.

2.2.15. WampServer

WampServer atau wamp adalah sebuah aplikasi yang dapat menjadikan komputer pemakai menjadi sebuah server. Kegunaan wamp ini untuk membuat jaringan lokal sendiri dalam artian pemakai dapat membuat website secara offline untuk masa coba-coba di komputer sendiri. Kelebihan wamp, yaitu praktis dan tidak perlu banyak melakukan setting.



Gambar 2.25. Wamp Server

2.2.16. Apache

APACHE merupakan salah satu jenis program yang bertujuan untuk mengirimkan kembali sintaks yang telah diberikan oleh user dan menampilkan kembali dalam bentuk dalam format hasilnya saja. Program *APACHE* ini hanya terdapat pada sebuah *server* yang terdapat pada ISP (Internet Service Provider) atau di Indonesia disebut Jasa Penyelenggara Internet. Saat ini program yang paling banyak terdapat pada *server-server* di dunia adalah *APACHE*. Hal ini disebabkan karena beberapa keunggulan yang diberikan.

Keunggulan-keunggulan tersebut antara lain :

1. Memiliki kemapuan pembacaan yang tinggi dan mudah disesuaikan
2. Memiliki tingkat keamanan yang tinggi. Hal ini dibutuhkan bagi para penyimpan data yang memiliki tingkat kerahasian tertentu.
3. Dapat dijalankan untuk berbagai macam operasi sistem.

APACHE merupakan suatu nama yang diambil dari istilah "*A PAtCHy*" *server* yang dapat diartikan dengan mengambil suatu patch atau bagian dari program yang diberikan.

APACHE sudah berkembang sejak versi pertamanya. Sampai saat ditulisnya artikel ini versi terakhirnya yang ada yaitu *APACHE* ver 2.0.54.

APACHE bersifat open source, artinya setiap orang boleh menggunakannya, mengambil dan bahkan mengubah kode programnya.

Tugas utama *APACHE* adalah menghasilkan halaman web yang benar kepada peminta, berdasarkan kode PHP yang dituliskan oleh pembuat halaman web. Jika diperlukan juga berdasarkan kode PHP yang dituliskan, maka dapat saja suatu *database* diakses terlebih dahulu (misalnya dalam *MySQL*) untuk mendukung halaman web yang dihasilkan.

2.2.17. PHP MyAdmin

Pengelolaan *database* dengan *MYSQL* harus dilakukan dengan mengetikkan baris-baris perintah yang sesuai (*command line*) untuk setiap maksud tertentu. Jika ingin membuat *database*, pengguna harus mengetikkan baris perintah yang sesuai untuk membuat *database*. Tetapi hal tersebut tentu cukup menyulitkan karena pengguna harus hafal dan mengetikkan perintahnya satu persatu.

Banyak sekali perangkat lunak yang dapat dimanfaatkan untuk mengelola *database* dalam *MySQL*, salah satunya adalah PHP*MyAdmin*. Dengan PHP*MyAdmin* pengguna dapat membuat tabel, mengisi data dan lain-lain dengan mudah tanpa harus hafal perintahnya. Untuk mengaktifkan PHP*MyAdmin* langkah-langkahnya adalah : yang pertama setelah meng-*install XAMPP*, pengguna harus mengaktifkan web *server APACHE* dan *MySQL* dari control panel *XAMPP*. Yang kedua, jalankan di *browser* yang ada *(IE, Mozilla Firefox* atau *Opera)* lalu ketikkan alamat web berikut : http://localhost/PHPmyadmin/ pada address bar lalu tekan Enter. Langkah ketiga apabila telah nampak *interface* (tampilan antar muka) PHP*MyAdmin* pengguna bisa memulainya dengan mengetikkan nama *database*, nama tabel dan seterusnya.

2.2.18. Graphic User Interface GUI

Antar muka komputer yang berbasiskan grafis. Adalah citra grafis yang ditampilkan di layar komputer yang memungkinkan untuk mengakses aplikasi software dengan memakai menu dropdown, dialog box, radio button, check box,

panel, tabs, toolbar, icon shortcuts dan tool lain. Atau bisa juga dikatakan jenis antarmuka pengguna yang memungkinkan orang untuk berinteraksi dengan program dengan lebih banyak dengan gambar daripada perintah teks. Sebuah GUI grafis menawarkan ikon, dan visual indikator, sebagai lawan dari antarmuka berbasis teks. Contoh perangkat yang menggunakan GUI: perangkat genggam seperti MP3 Players, *Portable Media Player* dan perangkat Permainan. GUI ini dimulai pertama kali dengan X Windows hasil penelitian di MIT, kemudian Macintosh, Sun View, lalu disusul oleh Microsoft Windows, dan sistem operasi lainnya.

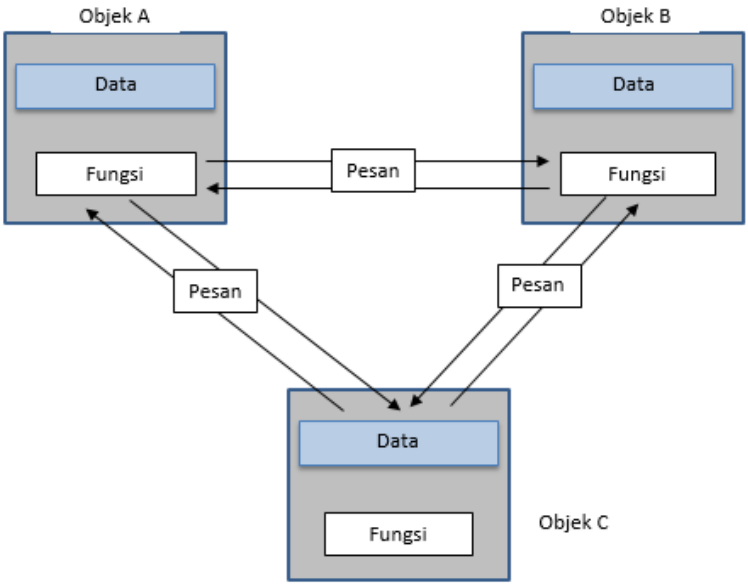
2.2.19. JAVA

Java merupakan suatu teknologi perangkat lunak yang di dalamnya mencakup bahasa pemrograman. Selain itu Java juga merupakan suatu platform yang memiliki *virtual machine* dan *library* yang diperlukan untuk menulis dan menjalankan suatu program. Java sendiri pada awalnya dikembangkan pada lingkungan komputer oleh *Sun Microsystem Inc*. dengan tujuan untuk menghasilkan suatu bahasa komputer sederhana tanpa harus terikat pada arsitektur tertentu. Maka tak heran jika dalam perkembangannya Java dapat dijalankan pada berbagai platform sistem operasi seperti Linux, Windows maupun Unix.

Karena keunikan dan kehandalannya, Java pun semakin banyak dikenal. Pada tahun 1995 Sun meluncurkan sebuah browser berbasis Java dengan julukan Hot Java, kemudian diikuti Netscape yang memutuskan untuk membuat browser dengan dilengkapi bahasa Java di dalamnya. Setelahnya ikut bergabung pula berbagai pengembang ternama diantaranya IBM dan Microsoft. Kemudian pada tahun berikutnya, Sun Microsystem Inc. merilis Java Software Development Kit (JDK) pertamanya, JDK 1.1. Dan terus berkembang dari pemrograman applet yang berjalan di browser menjadi pemrograman kelas dunia yang banyak digunakan untuk pengembangan aneka ragam aplikasi. Tidak hanya pada komputer, Java juga dapat berjalan pada bermacam-macam perangkat, mulai telepon genggam sampai aplikasi berskala enterprise pada komputer server.

2.2.20. Object Oriented Programing (OOP)

Pemrograman Berorientasi Objek adalah suatu cara baru dalam berfikir dan belogika dalam menghadapi masalah-masalah dengan bantuan computer. Konsep objek dalam OOP adalah konsep atau abstraksi tentang sesuatu yang memiliki arti bagi aplikasi yang akan dikembangkan. Objek biasanya adalah kata benda, namun objek dalam konteks OOP bukan hanya objek nyata yang bias diraba dan dilihat secara asat mata seperti benda yang biasa kita temukan. Dibawah ini merupakan gambar pengorganisasian data dan fungsi pada pendekatan berorientasi objek.



Gambar 2.26. Pengorganisasian Data dan Fungsi pada Pendekatan Berorientasi Objek.

2.2.21. Java Deplopment Kit (JDK)

Java Deplopment Kits (JDK) berisi sekumpulan kakas baris perintah (*Command-line tool*), utilitas, dokumentasu serta *applet* untuk menciptakan program java.

Beberapa komponen yang ada pada *Java Deplopment Kits* (JDK) diantaranya :

d. Kompilator berfungsi untuk mengkompilasi *file* kode sumber java menjadikelas *bytcode*.

e. Interpreter merupakan modul utama system Java yang digunakan aplikasi Java.
f. *Applet Viewer* adalah kakas untuk pengujian *java applet* secara minimal, berisi argumen *option* yang menspesifikasikan cara menjalankan *java applet.* Argumen *URL* yang menspesifikasikan dokumen *URL* berisi halaman HTML dengan *embedded Java*
g. *Java Debugger* adalah utilitas baris perintah untuk melakukan*debugging* aplikasi Java.
h. *Java class file disassembler* berfungsi untuk melakukan penguraian (*disassambler*) terhada *file class*. Keluaran *default* aktifias *disasambler* berisi daftar dari data public (*public data*) dan metode publik (*public method*) di kelas.
i. *Java header and stub file generator* berfungsi untuk menhasilkan C *header* dan *file* kode sumber untuk mengimplementasi metode-metode *Java* dalam bahasa C.
j. *Java documentation generator* adalah kakas untuk menhasilakan dokumentasi API secara lengsung dari kode sumber Java.
k. *Applet demo* contoh-contoh program Java yang ada pada JDK, semuanya berisi kode sumber.

l. *Source code java API* kode sumber secara lengkap untuk seua kelasyang membentuk Java API dan secara otomatis dipasang di *hard drive* saat melakukan dekompres JDK tapi masih dalam bentuk yang *file* dekompres.

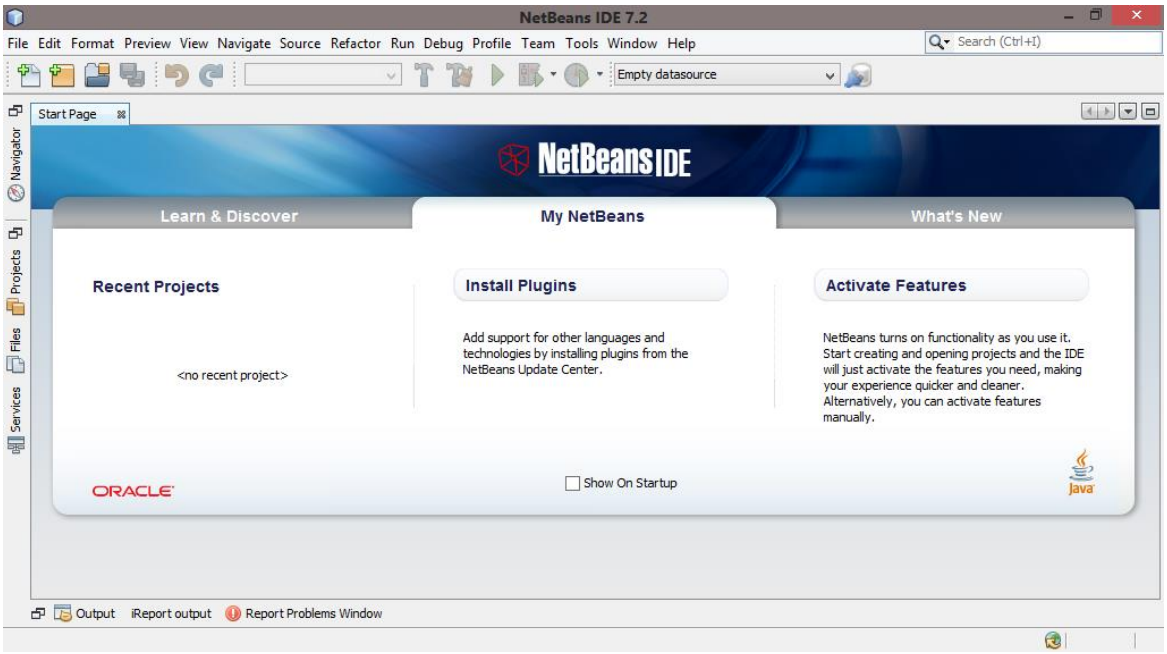
2.2.22. Java Runtime Environtment (JRE)

Java Runtime Environment adalah salah satu teknologi yang dibuat dan dikembangkan oleh Oracle. dengan *Java Runtime Environment* (JRE), dimungkinkan untuk menjalankan aplikasi yang disebut "*Applet*" yang ditulis dalam bahasa pemrograman berbasiskan Java. *Applet* memungkinkan Anda untuk merasakan pengalaman yang jauh lebih mengesankan daripada sekadar berinteraksi dengan halaman HTML yang statis dan kaku. *Java Plug-in*

technology, yang termasuk bagian *Java Runtime Environment Standard Edition* (JRE), menghubungkan antara browser dengan *platform* berbasiskan *Java*.

2.2.23. Netbeans IDE

Netbeans merupakan sebuah aplikasi Integrated Development Environment (IDE) yang berbasiskan Java dari *Sun Microsystems* yang berjalan di atas *swing*. *Swing* merupakan sebuah teknologi Java untuk pengembangan aplikasi desktop yang dapat berjalan pada berbagai macam *platform* seperti windows, linux, Mac OS X dan Solaris. Sebuah IDE merupakan lingkup pemrograman yang di integrasikan ke dalam suatu aplikasi perangkat lunak yang menyediakan Graphic User Interface (GUI), suatu kode editor atau text, suatu compiler dan suatu *debugger* menulis, meng-*compile*, mencari kesalahan dan menyebarkan program netbeans yang ditulis dalam bahasa pemrograman java.



Gambar 2.27. Netbeans IDE 7.2