# BAB 2

# LANDASAN TEORI

## 2.1 Teori – teori Dasar/Umum

Dalam menganalisis dan merancang suatu sistem, perlu adanya pertimbangan – pertimbangan yang didasari oleh berbagai landasan teori yang dikenal secara umum. Beberapa landasan teori umum tersebut diuraikan di bawah ini.

### 2.1.1 Proyek

Menurut Kamus Besar Bahasa Indonesia, proyek adalah rencana pekerjaan dengan sasaran khusus dan dengan saat penyelesaian yang tegas.

Menurut Wikipedia, suatu proyek dalam bisnis dan ilmu pengetahuan adalah sebuah kolaborasi yang biasanya melibatkan penelitian dan rancangan, yang dimana direncanakan dengan hati-hati untuk mendapatkan tujuan tertentu.

Menurut Schwalbe (2000, p4), proyek adalah suatu usaha yang bersifat sementara untuk menghasilkan suatu produk atau layanan yang unik.

Menurut Gray (2000, p4), proyek adalah sesuatu yang kompleks, tidak rutin, usaha yang tepat waktu yang dibatasi oleh waktu , sumberdaya , dan spesifikasi performa yang didisain untuk kebutuhan pelanggan.

Menurut Olson (2003, p2), suatu proyek melibatkan aktivitas yang baru dan kompleks, tujuan yang dapat didefinisikan melintasi berbagai tingkatan organisasi dan merupakan aktivitas yang unik.

### 2.1.2 *Software Project Management*

Menurut Wikipedia, *Software Project Management* adalah seni dan ilmu perencanaan dan proyek perangkat lunak terkemuka. Ini adalah sub-disiplin manajemen proyek dalam proyek-proyek perangkat lunak mana yang direncanakan, dipantau dan dikendalikan.

Tujuan perencanaan proyek adalah untuk mengidentifikasi ruang lingkup proyek, estimasi kerja yang terlibat, dan membuat jadwal proyek. Perencanaan proyek dimulai dengan persyaratan yang mendefinisikan perangkat lunak untuk dikembangkan. Rencana proyek ini kemudian dikembangkan untuk menggambarkan tugas-tugas yang akan mengarah pada penyelesaian.

Sementara tujuan dari pemantauan dan pengendalian proyek adalah untuk menjaga tim dan manajemen tetap *up to date* pada kemajuan proyek. Jika proyek menyimpang dari rencana, maka *project manager* dapat mengambil tindakan untuk memperbaiki masalah. Pemantauan dan pengendalian proyek melibatkan status pertemuan untuk mengumpulkan status dari tim. Ketika perubahan harus dilakukan, perubahan kendali digunakan untuk menyimpan produk tetap *up to date*.

### 2.1.3 Manajemen Waktu Proyek

Menurut Schwalbe (2000, p11), didefinisikan meliputi proses - proses yang dibutuhkan untuk memastikan ketepatan waktu pengerjaan suatu proyek. Proses – proses utama yang terlibat pada manajemen proyek ini adalah :

1. Definisi Aktivitas *(Activity definition)* Pendefinisian aktivitas menghasilkan *Work Breakdown Structure* (WBS) yang lebih spesifik dan penjelasan *support*

oleh tim proyek. Tujuan dari proses ini ialah untuk memastikan tim proyek memiliki pengertian yang mendalam akan aktivitas atau tahapan yang harus dijalankan sebagai bagian dari ruang lingkup proyek.

2. Barisan Aktivitas *(Activity Sequencing)* Setelah mendefinisikan aktivitas, langkah selanjutnya adalah barisan aktivitas atau *activity sequencing*. Yaitu meliputi memeriksa kembali aktivitas pada detail *Work Breakdown Structure* (WBS), detail deskripsi produk, asumsi, dan batasan untuk menentukan hubungan antar aktivitas.
3. Estimasi Durasi Aktivitas *(Activity Duration Estimating)* Proses selanjutnya ialah mengestimasi durasi aktivitas. *Output* dari proses ini ialah estimasi durasi untuk setiap aktivitas.

### 2.1.4 Pengertian *Monitoring*

Menurut Kamus Besar Bahasa Indonesia (2001), *monitoring* adalah mengawasi, mengamati, atau mengecek dengan cermat, terutama untuk tujuan khusus, memantau. Menurut Djamin (1993, p114), untuk dapat melaksanakan *monitoring* yang baik diperlukan :

a. Sistem pelaporan yang baik, yang memerlukan adanya komunikasi di antara penanggung jawab masing – masing bagian kegiatan *(project team members)* sehingga dapat diketahui apa yang sedang terjadi di lapangan.
b. Orang – orang yang tepat *(right people)*, maksudnya,
    a) penanggung jawab terhadap setiap kegiatan *(project participants responsible for activities)*;

b) pimpinan *(supervisors)* yang dapat mengintegrasikan laporan dari suatu kegiatan dengan kegiatan lainnya, untuk dilakukan penyesuaian - penyesuaian kegiatan.

c) Informasi yang benar. Informasi yang benar hanya dapat diperoleh bila penanggung jawab setiap kegiatan atau langkah dipegang oleh orang – orang yang tepat *(right people)*.

c. Waktu yang tepat. Gejala – gejala hendaknya di laporkan atau di atasi sebelum terjadi, agar dapat dilakukan tindakan – tindakan pengaman *(corective action)* jauh sebelumnya.

#### 2.1.4.1 Tujuan *Monitoring*

Menurut Departemen Sosial Republik IndonesiaBiro Perencanaan (2007), yang bersumber dari http://perencanaan.depsos.go.id/.php?mod=news&id=44&t=p,tujuan *monitoring* yaitu :

1. Mengkaji apakah kegiatan - kegiatan yang dilaksanakan telah sesuai dengan rencana.
2. Mengidentifikasi masalah yang timbul agar langsung dapat diatasi.
3. Melakukan penilaian apakah pola kerja dan manajemen yang digunakan sudah tepat untuk mencapai tujuan proyek.
4. Mengetahui kaitan antara kegiatan dengan tujuan untuk memperoleh ukuran kemajuan.
5. Menyesuaikan kegiatan dengan lingkungan yang berubah, tanpa menyimpang dari tujuan.

### 2.1.4.2 Manfaat *Monitoring*

Menurut Departemen Sosial Republik Indonesia Biro Perencanaan (2007), yang bersumber dari http://perencanaan.depsos.go.id/index.php?mod=news&id=44&t=p, manfaat *monitoring* yaitu :

1. Bagi pihak penanggung jawab program :
    - Salah satu fungsi manajemen yaitu pengendalian atau supervisi.
    - Sebagai bentuk pertanggung-jawaban (akuntabilitas) kinerja.
    - Untuk meyakinkan pihak - pihak yang berkepentingan.
    - Membantu penentuan langkah - langkah yang berkaitan dengan kegiatan proyek selanjutnya.
    - Sebagai dasar untuk melakukan *monitoring* dan evaluasi selanjutnya.

2. Bagi pihak pengelola proyek, yaitu :
    - Membantu untuk mempersiapkan laporan dalam waktu yang singkat.
    - Mengetahui kekurangan - kekurangan yang perlu diperbaiki dan menjaga kinerja yang sudah baik.
    - Sebagai dasar (informasi) yang penting untuk melakukan evaluasi proyek.

### 2.1.4.3 Tipe dan Jenis *Monitoring*

Menurut Departemen Sosial Republik Indonesia Biro Perencanaan (2007), yang bersumber dari http://perencanaan.depsos.go.id/index.php?mod=news&id=44&t=p, tipe dan jenis *monitoring* yaitu :

- Aspek masukan (*input*) proyek antara lain mencakup : tenaga manusia, dana, bahan, peralatan, jam kerja, data, kebijakan, manajemen yang dibutuhkan untuk melaksanakan kegiatan proyek.
- Aspek proses / aktivitas yaitu aspek dari proyek yang mencerminkan suatu proses kegiatan, seperti : penelitian, pelatihan, proses produksi, pemberian bantuan.
- Aspek keluaran (*output*), yaitu aspek proyek yang mencakup hasil dari proses yang terutama berkaitan dengan kuantitas (jumlah).

#### 2.1.4.4 Memonitor dan mengontrol kerja proyek

Menurut *a guide to the Project Management Body of Knowledge* (2004, p95) :

1. Perencanaan proyek manajemen

   Isi perencanaan proyek manajemen merubah pandangan yang terlalu mengandalkan area aplikasi dan kompleksitas dari proyek. Proses ini menghasilkan perencanaan proyek manajemen yang diperbaharui dan diperbaiki melalui proses penggantian kontrol yang terintegrasi. Perencanaan proyek manajemen mendefinisikan bagaimana proyek dijalankan, dimonitor, dikontrol dan ditutup. Perencanaan proyek manajemen melakukan pengumpulan dokumen dari *output*, proses perencanaan dari sekumpulan proses termasuk :

   - Proses manajemen proyek dipilih oleh tim manajemen proyek.
   - Tingkatan implementasi pada masing – masing proses dipilih.
   - Deskripsi dari *tool* dan teknik digunakan untuk menyelesaikan proses.

- Bagaimana proses yang dipilih akan digunakan untuk mengatur proyek khusus, termasuk ketergantungan dan interaksi di antara *input*, proses, *output*.
- Bagaimana pekerjaan dijalankan untuk tercapainya tujuan projek.
- Bagaimana penggantian yang dilakukan akan dimonitor dan dikontrol.
- Bagaimana konfigurasi manajemen akan dilaksanakan.
- Bagaimana integritas pelaksanaan pengukuran akan dipertahankan dan digunakan.
- Kebutuhan dan teknik untuk berkomunikasi di antara pemegang saham.
- Siklus hidup proyek dipilih dan, untuk berbagai tahapan proyek, diasosiasikan tahapan proyek.
- Kunci manajemen mere*view* isi, luas dan waktu untuk menfasilitasi isu – isu yang ada dan menunda pengambilan keputusan.

2. Informasi pelaksanaan kerja

Status informasi dari pengerjaan proyek dilaksanakan untuk menyelesaikan pekerjaan proyek secara rutin dikumpulkan sebagai bagian dari pelaksanaan perencanaan proyek manajemen. Yang termasuk dalam informasi ini, tetapi tidak dibatasi untuk :

- Kemajuan jadwal menunjukkan status informasi.
- Dapat disampaikan yang telah dilengkapi dan tidak dilengkapi.
- Jadwal kegiatan yang dimulai dan telah diselesaikan.
- Perluasan untuk standar kualitas ditemukan.

- Biaya diperhitungkan dan diperhatikan.
- Perkiraan untuk kelengkapan jadwal kegiatan dimulai.
- Kelengkapan persentase secara fisik dari kegiatan jadwal tidak dikembangkan.
- Pelajaran dokumentasi di*post* untuk pelajaran yang diajarkan berbasis pengetahuan.
- Pemanfaatan sumber sepenuhnya.

3. Menolak permintaan perubahan

   Menolak permintaan perubahan termasuk permintaan perubahan yang mendukung dokumentasi, dan mere*view* perubahan status yang menunjukkan pembagian dari menolak permintaan penggantian.

**2.1.4.5 Memonitor dan mengontrol kerja proyek : *Tools* dan *Techniques***

Menurut *a guide to the Project Management Body of Knowledge* (2004, p95) :

1. Metodologi manajemen proyek

   Metodologi manajemen proyek didefinisikan sebagai suatu proses yang membantu tim manajemen proyek dalam *monitoring* dan *controlling* kerja proyek yang dilaksanakan berikut dengan perencanaan proyek manajemen.

2. Sistem informasi manajemen proyek

   Sistem informasi manajemen proyek, suatu sistem automatisasi, yang digunakan oleh tim manajemen proyek untuk memonitor dan mengontrol pemilihan

kegiatan yang direncanakan dan dijadwalkan pada perencanaan proyek manajemen.

3. Teknik penilaian yang diterima

   Teknik penilaian yang diterima mengukur kinerja proyek seperti pergerakan dari proyek awal melalui penutupan proyek. Metodologi penilaian manajemen yang diterima bermaksud untuk peramalan yang akan datang berdasarkan kinerja masa lampau.

4. Pertimbangan ahli

   Pertimbangan ahli digunakan oleh tim manajemen proyek untuk memonitor dan mengontrol kerja proyek

### 2.1.5 Pengertian Sistem

Lucas (1992, p2) mengatakan, sistem adalah suatu himpunan komponen dan variable yang terorganisasi, saling berinteraksi, saling bergantungan satu sama lain dan terpadu.

Menurut Hall (2001, p5) mengatakan, sistem adalah sekelompok dua atau lebih komponen – komponen yang saling berkaitan atau sub elemen–sub elemen yang bersatu untuk mencapai tujuan yang sama.

Menurut McLeod (2001, p10) sistem adalah sekelompok elemen yang berintegrasi dengan maksud yang sama untuk mencapai suatu tujuan.

## 2.16 Pengertian Informasi

Menurut Kamus Komputer dan Teknologi informasi mengatakan, informasi adalah Keterangan, penerangan. Data yang telah diproses ke dalam suatu bentuk yang mempunyai arti bagi si penerima dan mempunyai nilai nyata, sehingga dapat dipakai sebagai dasar untuk mengambil keputusan, dan terasa bagi keputusan saat itu atau keputusan mendatang.

Menurut Jogiyanto (1990, p11) Informasi adalah data yang dapat diolah yang lebih berguna dan berarti bagi yang menerimanya.

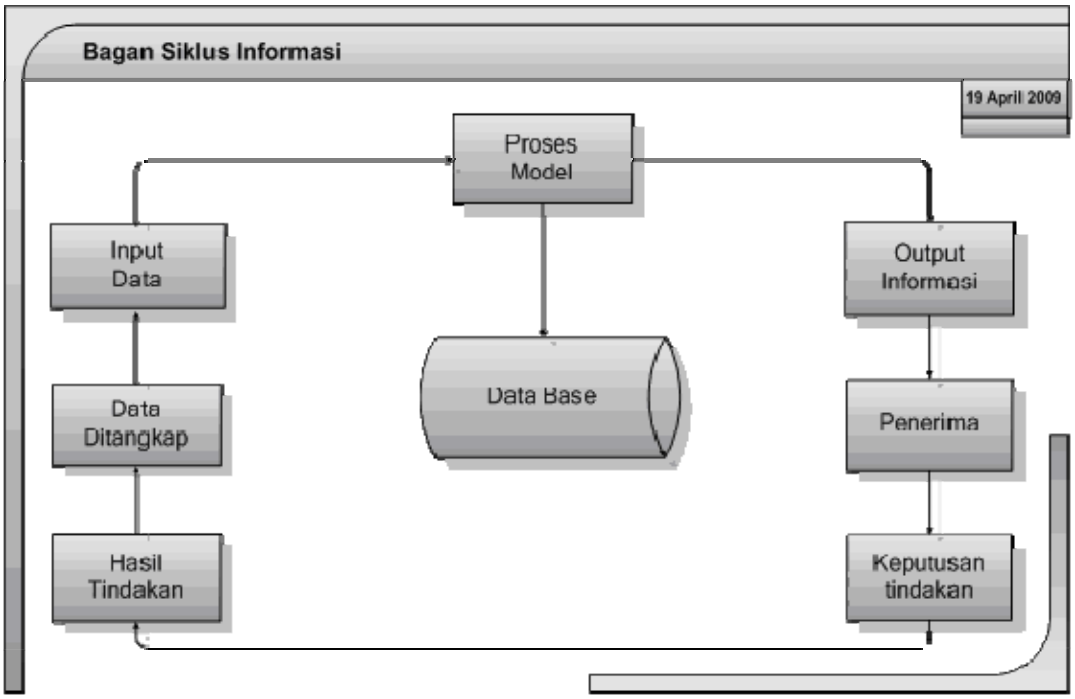
Menurut Murdik (1973, p12) Informasi adalah data yang telah diolah menjadi suatu bentuk yang berarti bagi penerimanya dan bermanfaan dalam pengambilan keputusan saat ini atau mendatang.

Menurut Azmi dalam *website*nya mengatakan bahwa Informasi adalah data yang diproses kedalam bentuk yang lebih berarti bagi penerima dan berguna dalam pengambilan keputusan, sekarang atau untuk masa yang akan datang. Informasi dalam suatu lingkungan sistem informasi memiliki beberapa ciri-ciri yaitu :

1. **Benar atau salah**, Ini dapat berhubungan dengan realitas atau tidak bila penerimaan informasi yang salah dipercayai mengakibatkan sama seperti benar.
2. **Baru**, Informasi dapat sama sekali baru dan segar bagi penerimanya.
3. **Tambahan**, Informasi dapat memperbaharui atau memberikan tambahan baru pada informasi yang talah ada.
4. **Korektif**, Informasi dapat menjadi suatu korektif atas informasi yang salah.

5. **Penegas**, Informasi dapat mempertegas informasi yang telah ada, ini berguna karena meningkatkan persepsi penerimanya atau kebenaran informasi tersebut.

Menurut Jhon Burch (1986, p3) mengemukakan suatu bentuk siklus informasi (*Information Cycle*) seperti terlihat pada gambar berikut :




**Gambar 2.1 Siklus Informasi**

Data yang diolah melalui suatu *model* menjadi suatu informasi, kemudian *user* menerima informasi tersebut, membuat suatu keputusan dan melakukan tindakan yang berarti menghasilkan suatu tindakan yang lain akan membuat sejumlah data kembali, data tersebut akan ditangkap sebagai input untuk diproses selanjutnya.

### 2.1.7 Konsep Analisis dan Perancangan

Menurut kamus komputer dan teknologi informasi, analisis adalah analisa; kupasan.

Menurut McLeod (2001, p190) mengatakan bahwa analisis sistem (*system analysis*) adalah penelitian atas sistem yang telah ada dengan tujuan untuk merancang sistem baru atau diperbaharui. Di dalam tahap analisis sistem, analisis sistem terus bekerja sama dengan manager, dan komite pengarah sistem informasi manajemen yang terlibat dalam titik – titik yang penting.

Tahap – tahapannya antara lain :

a. Mengumpulkan penelitian sistem

b. Mengorganisasikan tim proyek.

c. Mendefinisikan kebutuhan informasi.

d. Mendefinisikan kriteria kinerja sistem.

e. Menyiapkan usulan rancangan.

f. Menyetujui atau menolak rancangan sistem.

Menurut Jugiyanto (1990, p129) mengatakan analisis sistem merupakan penguraian dari suatu sistem informasi yang utuh ke dalam bagian dan komponennya dengan maksud untuk mendefinisikan dan mengevaluasi permasalahan – permasalah, kesempatan – kesempatan, hambatan – hambatan yang terjadi dan kebutuhan – kebutuhan yang diharapkan sehingga dapat di usulkan dapat di usulkan baikan – perbaikannya.

Menurut Jugiyanto (1990, p130) memaparkan langkah – langkah analisis sistem yang antara lain adalah :

a. Mengidentifikasi masalah.

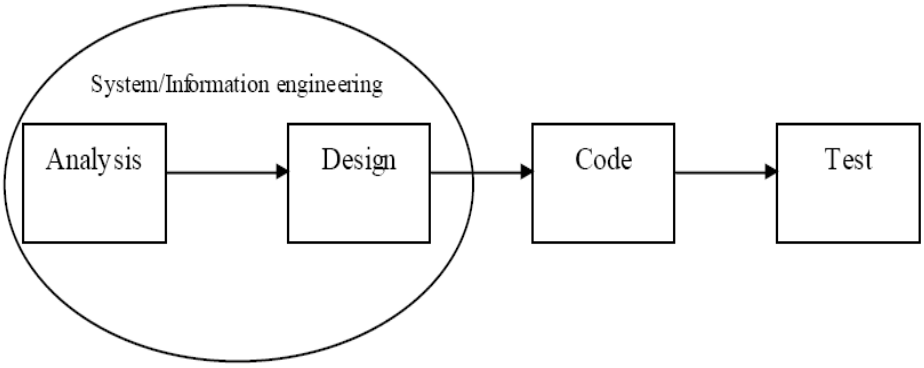
b. Memahami kerja dari sistem yang ada.

c. Menganalisa sistem.

d. Membuat laporan hasil analisis.

### 2.1.8 Pengertian Rekayasa Perangkat Lunak (RPL)

Menurut Pressman (2001, p20), rekayasa piranti lunak adalah penerapan dan penggunaan prinsip-prinsip rekayasa dalam usaha menghasilkan piranti lunak yang ekonomis, dapat dihandalkan dan bekerja secara efisien pada mesin yang sesungguhnya. Paradigma rekayasa piranti lunak yang sering digunakan adalah *the classic life cycle* atau lebih dikenal dengan *waterfall model*.(Pressman,2001,p28 29).




**Gambar 2.2 Life Cycle Waterfall Model**

Tahapan-tahapan yang dilakukan dalam waterfall model ini adalah :

1. Analisis dan perancangan sistem (*System Information dan modelling*)

   Piranti lunak merupakan bagian dari suatu sistem yang lebih besar sehingga langkah pertama yang harus dilakukan adalah menetapkan kebutuhan untuk semua elemen sistem. Hal ini penting karena piranti lunak harus lebih dulu berinteraksi dengan elemen-elemen lainnya seperti perangkat keras, pengguna dan database.

2. Analisis kebutuhan piranti lunak (*Software Requirement Analysis*)

   Proses pengumpulan kebutuhan dilakukan secara intensif dan terfokus khususnya pada piranti lunaknya. Untuk mengerti sistem yang akan dibangun, seorang pembuat sistem harus memahami informasi yang dibutuhkan piranti lunak, fungsi-fungsi, *performance* dan antar muka.

3. Perancangan (*Design*)

   Perancangan dipusatkan pada empat atribut program yaitu struktur data, arsitektur piranti lunak, perincian prosedur, dan karakteristik antar muka. Sama seperti piranti lunak, perancangan ini juga menjadi bagian dari konfigurasi piranti lunak.

4. Pengkodean (*Coding*)

   Di dalam tahapan ini, pengkodean bertujuan untuk menterjemahkan desain ke dalam bentuk yang dapt dimengerti oleh mesin. Jika desain

dilakukan secara terperinci, pengkodean dapat dilakukan secara mekanik seluruhnya.

5. Pengujian (*Testing*)

   Setelah proses pengkodean selesai, dilakukan pengujian sampai semua perintah selesai diuji. Pengujian ini bertujuan untuk menemukan kesalahan dan memastikan keluaran yang dihasilkan sesuai dengan apa yang diharapkan.

6. Dukungan (*Support*)

   Piranti lunak ini akan mengalami perbaikan secara terus menerus setelah dipergunakan oleh pengguna dan apabila terjadi perubahan karena terjadinya kesalahan, hal ini menyebabkan diperlukannya perbaikan fungsional dan unjuk kerja piranti lunak. Pemeliharaan piranti lunak menawarkan setiap langkah daur hidup yang terdahulu menjadi program yang sudah ada daripada membuat suatu program yang baru.

#### 2.1.8.1 Elemen Pokok Rekayasa Piranti Lunak

Elemen pokok rekayasa piranti lunak sebagai berikut :

1. *Process*

   Pada proses RPL merupakan perekat yang mengikat lapisa-lapisan teknologi secara bersama dan memampukan perangkat lunak komputer untuk berkembang secara tepat waktu dan rasional. Proses menggambarkan suatu kerangka kerja

untuk sekumpulan dari *Key Process Areas* (KPAs) yang harus ditetapkan untuk keefektifan pengiriman dari teknologi rekayasa piranti lunak. *Key Process Areas* merupakan bentuk dasar bagi control managemen dari proyek-proyek piranti lunak dan menerapkan konteks dimana metode teknis dapat diterapkan. Produk kerja (model, dokumen, data, laporan, bentuk , dll) yang diproduksi, kualitas yang dipastikan, dan perubahan yang diatur dengan baik.

2. *Methods*

Metode Rekayasa Piranti Lunak menyediakan teknik bagaimana cara untuk membangun piranti lunak. Metode yang meliputi suatu ruang lingkup tugas yang luas yang meliputi analisa kebutuhan, desain, konstruksi program, pengujian, dan dukungan. Metode Rekayasa Piranti Lunak bergantung pada prinsip-prinsip dasar yang memerintah tiap area dari teknologi dan kegiatan modelling dan teknik deskriptif berikutnya.

3. *Tools*

Peranan Rekayasa Piranti Lunak menyediakan dukungan yang otomatis dan semi-otomatis bagi proses dan metode-metodenya. Ketika peralatan itu terintegrasi, maka informasi yang dibentuk oleh suatu alat dapat digunakan oleh yang lainnya. Suatu sistem yang mendukung perkembangan piranti lunak disebut *Computer-Aideds Software* (CASE), telah didirikan. CASE merupakan kombinasi dari *software, hardware*, dan suatu basis data rekayasa piranti lunak (sebuah tempat penyimpanan yang berisi informasi penting tentang analisis,

desain, konstruksi program, dan pengujian) untuk menciptakan suatu lingkungan analogis rekayasa piranti lunak untuk CAD/CAE (*computer-aided design/engineering)* untuk perangkat keras.

### 2.1.9 **Pengertian Internet**

Menurut Hahn (1996, p2) internet adalah jaringan besar yang dibentuk dari interkoneksi jaringan komputer dan komputer tunggal di seluruh dunia, lewat saluran telepon, satelit, dan sistem telekomunikasi lainnya.

Menurut Ellsworth (1995,p3) mengatakan internet(*Interconnection Network*)merupakan suatu jaringan yang luas yang terdiri dari jaringan komputer yang saling berhubungan di seluruh penjuru dunia. Internet terdiri dari ribuan sampai jutaan jaringan komputer yang tersebar di seluruh dunia yang terhubung lewat media seperti : satelit, jalur telepon dan sistem komunikasi lainnya.

Menurut Nugroho (2004,p1) internet adalah suatu media informasi komputerglobal yang dapat dikatakan sebagai teknologi tercanggih abad ini.

Sedangkan menurut Septanto (1998,p1) mendefinisikan internet secara umum yaitu sebuah jaringan *super network* yang terdiri dari kumpulan jaringan yang saling berhubungan satu sama lain dengan menggunakan protokol TCP/IP, dimana jaringa tersebut dapat dengan mudah di akses dari jarak jauh hanya dengan menggunakan saluran telepon lokal.

Menurut Sidharta (1996) : walaupun secara fisik Internet adalah interkoneksi antar jaringan komputer namun secara umum Internet harus dipandang sebagai sumber daya informasi. Isi Internet adalah informasi, dapat

dibayangkan sebagai suatu database atau perpustakaan multimedia yang sangat besar dan lengkap. Bahkan Internet dipandang sebagai dunia dalam bentuk lain (maya) karena hampir seluruh aspek kehidupan di dunia nyata ada di Internet seperti bisnis, hiburan, olah raga, politik dan lain sebagainya.

Heywood (1996) menerangkan : sejarah Internet bermula pada akhir dekade 60-an saat United States Department of Defense (DoD) memerlukan standar baru untuk komunikasi Internetworking. Yaitu standar yang mampu menghubungkan segala jenis komputer di DoD dengan komputer milik kontraktor militer, organisasi penelitian dan ilmiah di universitas. Jaringan ini harus kuat, aman dan tahan kerusakan sehingga mampu beroperasi didalam kondisi minimum akibat bencana atau perang.

Tahun 1969 Advanced Research Project Agency (ARPA) dibentuk tugasnya melakukan penelitian jaringan komputer mempergunakan teknologi packet switching. Jaringan pertama dibangun menghubungkan 4 tempat yaitu : UCLA, UCSB, Utah dan SRI International. Hingga tahun 1972 jaringan ini telah menghubungkan lebih dari 20 host dan disebut sebagai ARPANet. ARPANet kemudian menjadi backbone Internetworking institusi pendidikan, penelitian, industri dan kontraktor terutama yang berkaitan dengan jaringan militer (MILNet).

Tahun 1986 ARPANet mulai dikomersialkan dengan mengisolasikan jaringan militer. National Science Foundation (NFS) kemudian membiayai pembongkaran backbone ARPANet menjadi backbone Internet komersial dan dikelola oleh Advanced Network Service (ANS). Andrew S. Tanenbaum (1996) : andil besar

dalam perwujudan Internet adalah tergabungnya jaringan regional seperti SPAN (jaringan fisika energi tinggi), BITNET (jaringan mainframe IBM), EARN (jaringan akademis Eropa dan digunakan pula di Eropa Timur) dan ditambah dengan sejumlah link transatlantik yang beroperasi pada 64 Kbps - 2 Mbps pada tahun 1988.

Menurut Tung (1997), jaringan pendukung Internet di seluruh dunia adalah :

™ Amerika didorong oleh NFS - ANSNet dan CO+RE (jaringan non profit terbatas)

yang bekerjasama dengan Commercial Internet Exchange (CIX) serta Sprint (perusahaan telekomunikasi umum) tahun 1990. Pengesahaan RUU NREN (National Research and Education Network) oleh Kongres Amerika pada Desember 1991. Ditambah 8 aliansi jaringan regional yang tergabung dalam The Corporation for Regional an Enterprise Networking (CoREN) yaitu : BARRNet, CICNet, MIDNet, EARNet, NorthWestNet, MYSERNet, SURANet dan WestNet. CoREN bekerjasama dengan perusahaan telekomunikasi komersial MCI. ™ Kanada dengan jaringan backbone nasional CA*Net ™ Australian Academic and Research Network (AARNET) ™ The Europe Backbone (EBONE) dan The European UNIX Network (EUNet) dan

RIPE organisasi jaringan e-mail Eropa

™ Jepang memiliki Widely Integrated Distributed Environtment (WIDE), Today International Science Network (TISN), Japan Academic Interuniversity Network (JAIN) dan Japan UNIX Network (JUNET). Kebanyakan bekerjasama dengan

jaringan telekomunikasi komersial AT&T perwakilan Jepang yang disebut dengan SPIN. Pelayanan lain yang bersifat internasional adalah InterCon International KK (IIKK) dan Internet Initiative Japan (IIJ) yang berasosiasi dengan WIDE untuk menyediakan jaringan Internet dikawasan Asia, termasuk jaringan penelitian dan pendidikan untuk kawasan Asia (disponsori oleh NEC, IIJ dan WIDE) yang disebut AI3 (Asia Internet Interconnection Initiative) yang mengembangkan teknologi satelit komunikasi Ku Band

™ Belakangan muncul ABONE (Asia Backbone) yang didirikan oleh konsorsium negara-negara di Asia seperti Jepang, Korea, Thailand, Malaysia, Singapura, Indonesia dan Hongkong. Interkoneksi dunia tersebut memakai jaringan serat optik antar benua berkapasitas + 45 Mbps. (T3 +) dan jaringan satelit telekomunikasi.

Menurut Fahmi (2008) mengatakan Internet adalah suatu jaringan komputer yang satu dengan yang lain saling terhubung untuk keperluan komunikasi dan informasi. Sebuah komputer dalam satu jaringan internet dapat berada di mana saja atau bahkan di seluruh Indonesia. Sering juga internet diartikan sebagai jaringan komputer di seluruh dunia yang berisikan informasi dan sebagai sarana komunikasi data yang berupa suara, gambar, video dan juga teks. Informasi ini dibuat oleh penyelenggara atau pemilik jaringan komputer atau dibuat pemilik informasi yang menitipkan informasinya kepada penyedia layanan internet.

### 2.1.10 Pengertian World Wide Web (WWW)

Menurut McLeod (2001,p75) WWW adalah ruang informasi di internet tempat dokumen – dokumen hypermedia di simpan dan dapt diambil melalui suatu skema alamatyang unik. Internet menyediakan arsitektur jaringan, dan web menyediakan metode untuk menyimpan dan mengambil dokumen – dokumennya.

Menurut Jajang dalam artikelnya pada Kamus Komputer dan Teknologi Informasi (2005) mendefinisikan world wide web terdiri dari :

- Web adalah halaman informasi di internet
- Wide adalah lebar; luas; bermacam-macam; mata terbuka lebar; cerdas; sigap; waspada
- World Wide Web adalah halaman internet yang dapat di akses mendunia

Jadi WWW Adalah fasilitas internet yang menghubungkan dokumen dalam lingkup lokal maupun jarak jauh. Dokumen Web disebut Web Page dan link dalam Web menyebabkan user bisa pindah dari satu page ke page lain (hyper text), baik antar page yang disimpan dalam server yang sama maupun server diseluruh dunia. Pages diakses dan dibaca melalui Web Browser seperti Netscape Navigator atau Internet Explorer.

Web menjadi pusat kegiatan internet karena Web Pages yang berisi text dan grafik mudah diakses melalui Web Browser, Web menyediakan interface bagi

jaringan informasi online terbesar di dunia, dan jumlah informasi ini terus bertambah dengan pesat.

Web juga menjadi sistem pengiriman multimedia, karena fitur browser dan browser plug-in extension yang terus bermunculan menyediakan peluang untuk suara, gambar, telepon, animasi 3D dan videoconferencing melalui Net.

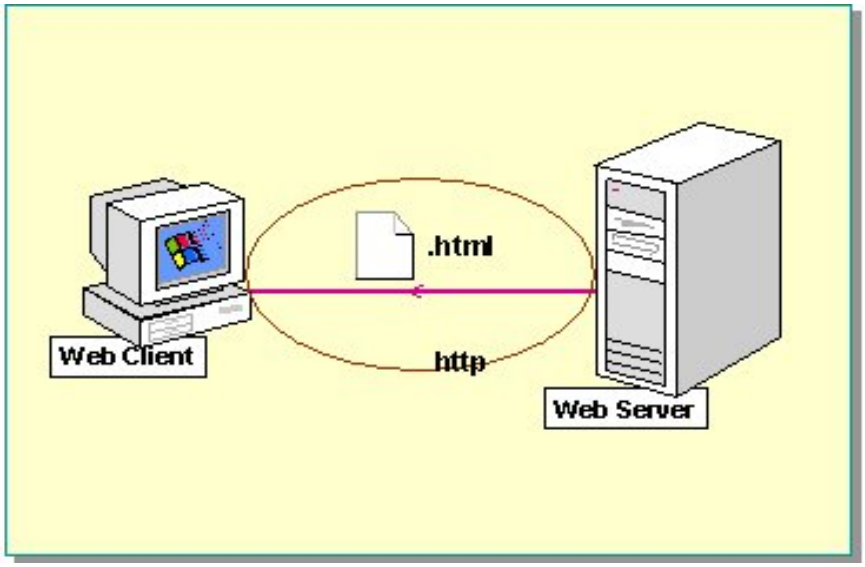
Browser terbaru juga mengerti bahasa Java yang memungkinkan down load semua aplikasi untuk di-run secara lokal. Dasar format Web adalah dokumen text yang digabung dengan HTML yang bisa mengatur format page serta Hypertext Link (URL) ke page lain. Kode HTML yang umum adalah karakter alfanumerik yang dapat diketik dengan text editor atau world processor. Banyak program terbitan Web yang menyertakan interfage grafis untuk kreasi Web Page dan membuat kode dengan otomatis.

Banyak word processor dan program-program yang mengalihkan dokumen ke format HTML. Oleh karena itu Web Pages dapat dibuat oleh user tanpa harus mempelajari sistem pengkodean. Kemudahan kreasi membantu cepatnya pertumbuhan Web.

World Wide Web dirancang oleh tim Berners-Lee dan staf ahli di laboratorium CERN di Jenewa Swiss tahun 1991.

Menurut Geto (2008) dalam *blog*nya mendefinisikan World Wide Web (biasa disingkat WWW) atau web adalah salah satu dari sekian banyak layanan yang ada di internet. Layanan ini paling banyak digunakan di internet untuk menyampaikan informasi karena sifatnya mendukung multimedia. Artinya

informasi tidak hanya disampaikan melalui teks, tapi juga gambar, video dan suara.




**Gambar 2.3 Transfer Data antara Web Client dengan Web Server**

**Web Server** adalah komputer yang tergabung dalam jaringan atau internet yang memberikan informasi.

**Web client** adalah komputer yang tergabung dalam jaringan atau internet yang meminta informasi. Untuk dapat mengakses web server, web client menggunakan aplikasi yang disebut **Web browser**.

Web browser meminta dan menerima data dari web server melalui suatu protokol yang disebut **http (hypertext tranfer protocol)**. Protokol ini bertugas untuk mengirimkan perintah dari web browser ke web server serta mengirimkan file/data dari web server ke web browser.

File yang dikirim dalam layanan web ini berekstensi ***.htm** atau ***.html**. HTML merupakan singkatan dari hypertext markup language, yaitu satu bahasa yang

digunakan untuk mendefinisikan susunan informasi dalam file hypertext. Hypertext sendiri adalah suatu struktur pemyampaian informasi dimana satu atau beberapa kata pada suatu file dapat di-link untuk mengeluarkan file baru yang biasanya berisi informasi detail tentang kata tersebut.

### 2.1.11 Pengertian *Database*

Menurut McLeod(2001,p258) *Database* adalah suatu koleksi data komputer yang terintegrasi, diorganisasikan, dan disimpan dengan suatu cara yang memudahkan pengambilan kembali. Dua tujuan utama dari *database* adalah meminimumkan pengulangan data dan mencapai independensi data.

Menurut Connoly (2002,p14) mengatakan *database* adalah suatu sistem penyimpanan data yang tersusun atas sekumpulan data – data yang secara logika saling terkait yang dirancang untuk memenuhi kebutuhan informasi perusahaan. Model *database* relasional adalah sistem yang paling banyak digunakan karena struktur logikanya yang sederhana. Pada model relasional, seluruh data di susun secara logikal dalam relasi – relasi atau tabel. Setiap relasi terdiri dari baris dan kolom, dan kolom dari relasi yang diberi nama tertentu disebut atribut. Sedangkan baris dari relasi disebut *tuple* dan setiap *tuple* memiliki satu nilai dari setiap *atribute*.

*Database* yang tabel – tabelnya saling terhubung dikatakan memiliki relasi. Karena tidak ada dua relasi yang memiliki dua *tuple* yang sama, maka setiap baris dapat didefinisikan secara unik dengan menggunakan *primary key*.

Munculnya sebuah atribut dalam beberapa relasi dapat mempresentasikan hubungan antar *tuple* dari relasi – relasi tersebut.

Pengertian *Database* dalam Kamus Komputer dan Teknologi Informasi (2005) adalah Basis data. Representasi kumpulan fakta yang saling berhubungan disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redudansi)yang tidak perlu, untuk memenuhi berbagai kebutuhan.

Data perlu disimpan dalam basis data untuk keperluan penyediaan informasi lebih lanjut. Data di dalam basis data perlu diorganisasikan sedemikian rupa, supaya informasi yang dihasilkan berkualitas. Organisasi basis data yang baik juga berguna untuk efisiensi kapasitas penyimpanannya.

Dalam maksud yang sama, bisa juga diartikan sebagai sekumpulan informasi yang disusun sedemikian rupa untuk dapat diakses oleh sebuah software tertentu. Database tersusun atas bagian yang disebut field dan record yang tersimpan dalam sebuah file. Sebuah field merupakan kesatuan terkecil dari informasi dalam sebuah database. Sekumpulan field yang saling berkaitan akan membentuk record.

Menurut Chaely (2008) mengatakan *Database* adalah kumpulan informasi yang disusun berdasarkan cara tertentu dan merupakan suatu kesatuan yang utuh. Dengan sistem tersebut data yang terhimpun dalam suatu *database* dapat menghasilkan informasi yang berguna.

Menurut Aurino (2007) *Database* (basis data) merupakan kumpulan data yang saling berhubungan. Hubungan antar data dapat ditunjukan dengan adanya *field*/kolom kunci dari tiap *file*/tabel yang ada. Dalam satu file atau table terdapat

*record-record* yang sejenis, sama besar, sama bentuk, yang merupakan satu kumpulan entitas yang seragam. Satu *record* (umumnya digambarkan sebagai baris data) terdiri dari *field* yang saling berhubungan menunjukan bahwa *field* tersebut dalam satu pengertian yang lengkap dan disimpan dalam satu *record*. Adapun Struktur *Database* adalah: *Database, File/Table, Record* Elemen data/*Field*Dari pengertian diatas dapat disimpulkan bahwa basis data mempunyai beberapa kriteria penting, yaitu :

1. Bersifat data *oriented* dan bukan program *oriented*.
2. Dapat digunakan oleh beberapa program aplikasi tanpa perlu mengubah basis datanya.
3. Dapat dikembangkan dengan mudah, baik *volume* maupun strukturnya.
4. Dapat memenuhi kebutuhan sistem-sistem baru secara mudah
5. Dapat digunakan dengan cara-cara yang berbeda.

Prinsip utama *Data Base* adalah pengaturan data dengan tujuan utama fleksibelitas dan kecepatan pada saat pengambilan data kembali. Adapun ciri-ciri basis data diantaranya adalah sebagai berikut :

1. Efisiensi meliputi kecepatan, ukuran, dan ketepatan
2. Data dalam jumlah besar.
3. Berbagi Pakai (dipakai bersama sama/*Sharebility*).

4. Mengurangi bahkan menghilangkan terjadinya duplikasi dan ketidak konsistenan data.

### 2.1.12 Pengertian 8 Aturan Emas

Menurut Shneiderman (1998, p74) ada delapan aturan emas dalam perancangan antar muka (*interface*) yang harus dipenuhi yaitu :

a. Berusaha untuk tetap konsisten.

   Urutan kejadian harus konsisten pada situasi – situasi yang serupa. Menu, layar dan perintah harus memiliki terminologi yang identik. Perancang juga harus merancang konsistensi warna, *layout*, kapitalisasi, jenis huruf dan sebagainya.

b. Memungkinkan *frequent user* untuk menggunakan jalan pintas (*shortcut*)

   Umumnya *user* yang sering menggunakan aplikasi ingin kecepatan respon yang tinggi. Untuk itu penggunaan abrasi, kunci spesial dan perintah tersembunyi sangat dibutuhkan.

c. Menawarkan umpan balik yang informatif.

   Untuk setiap aksi dari *user,*sistem harus memberikan umpan balik yang informatif. Untuk aksi minor, respon bersifat lebih umum. Untuk aksi mayor, respon bersifat lebih substansial.

d. Rancangan dialog untuk menghasilkan keadaan akhir

Umpan balik atas akhir dari suatu proses dan aksi akan sangat membantu dan juga *user* akan mendapat sinyal untuk melakukan aksi lainnya.

e. Memberikan pencegahan kesalahan dan penanganan kesalahan yang sederhana

Sedapat mungkin sistem didesain sedemikian rupa agar *user* tidak sering melakukan kesalahan yang serius, jika *user* melakukan kesalahan sistem harus dapat mendeteksi kesalahan dan menawarkan instruksi yang sederhana, konstruktif dan spesifik untuk penyembuhan (*recovery*).

f. Mengizinkan pembalikan keadaan (*undo*) yang mudah.

Dalam suatu waktu *user* mungkin tidak sengaja melakukan aksi yang tidak diinginkan dan ingin melakukan pembatalan. Sistem harus memberikan fungsi pembatalan sebanyak mungkin agar *user* merasa nyaman dan tidak takut dalam mencoba dan memakai sistem.

g. Mendukung tempat pengendali internal (*Internal locus of control*).

Pengguna ingin menjadi pengontrol sistem dan sistem akan merespon tindakan yang dilakukan pengguna daripada pengguna merasa bahwa

sistem mengontrol pengguna. Sebaiknya sistem dirancang sedemikan rupa sehingga pengguna menjadi inisiator daripada responden.

h. Mengurangi beban memori jangka pendek.

Keterbatasan memori pada manusia harus ditanggulangi dengan meminimalisasi proses penyimpanan memori dengan tampilan yang sederhana, konsolidasi tampilan halaman jamak dan pengurangan frekuensi perubahan *windows*.

## 2.2 Teori – teori Khusus

### 2.2.1 Pengertian UML

*Unified Modelling Language* adalah sebuah bahasa yang telah menjadi standard dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standard untuk merancang sebuah sistem. UML dapat digunakan untuk membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi, dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun.

Sejarah UML cukup panjang. UML adalah suatu kelanjutan dari puluhan metodologi pemodelan berorientasi objek. Masing-masing metodologi tersebut membawa notasi sendiri-sendiri, yang mengakibatkan timbul masalah baru apabila perlu

bekerjasama dengan grup/perusahaan lain yang menggunakan metodologi yang berlainan.

Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya, yaitu: Grady Booch dengan OOD, Jim Rumbaugh dengan OMT, dan Ivar Jacobson dengan OOSE. Usaha penggabungan dimulai sejak tahun 1994, pada tahun 1995 *draft* pertama UML telah dirilis. Konsepsi dasar UML terdiri dari 3 bagian, yaitu *structural classification, dynamic behaviour, dan model management.*

Pada umumnya, UML terdiri dari beberapa diagram antara lain:

a. *Use case diagram*

*Use case diagram* menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah apa yang diperbuat sistem, bukan bagaimana sebuah sistem berjalan. Sebuah usecase merepresentasikan sebuah interaksi antara aktor dengan sistem. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. *Use case diagram* dapat sangat membantu bila kita sedang menyusun kebutuhan sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang skenario tes untuk semua fitur yang ada dalam sistem.

Menurut Jeffrey L.Whitten, Lonnie D.Bentley, dan Kevin C.Dittman (2004, 256), *usecase modeling* adalah proses pemodelan fungsi-fungsi sistem dalam konteks peristiwa-peristiwa bisnis, siapa yang mengawalinya, dan bagaimana sistem itu merespon hal tersebut.

b. *Class diagram*

*Class* adalah sebuah spesifikasi yang jika diintansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan disain berorientasi objek. *Class* menggambarkan keadaan(atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut(metode/fungsi).

*Class diagram* menggambarkan struktur dan deskripsi *class*, *package*, dan objek beserta hubungan satu sama lain, kebijakan, pewarisan, asosiasi, dan lain-lain. *Class* memiliki tiga area pokok:

1) Nama (dan stereotipe)
2) Atribut
3) Metode

Atribut dan metode dapat memiliki salah satu sifat berikut

1) *Private,* tidak dapat dipanggil dari luar *class* yang bersangkutan.
2) *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak-anak yang mewarisinya.
3) *Public*, dapat dipanggil oleh siapa saja.

Menurut Whitten, Bentley, dan Dittman (2004, p418), diagram ini menggambarkan struktur objek sistem yang menunjukkan kelas objek yang menyusun sistem dan juga hubungan antara kelas objek tersebut.

*Class* dapat merupakan implementasi dari sebuah *interface*, yaitu *class* abstrak yang hanya memiliki metode. *Interface* tidak dapat langsung diinstansikan, tetapi harus

diimplementasikan dahulu menjadi sebuah *class*. Dengan demikian, *interface* mendukung resolusi metode pada saat *run time*.

Dalam *class diagram*, terdapat beberapa jenis hubungan antar *class*. Hubungan-hubungan tersebut adalah sebagai berikut:

1) Asosiasi, yaitu hubungan statis antar *class*. Umumnya menggambarkan *class* yang memiliki atribut berupa *class* lain, atau *class* yang harus mengetahui eksistensi *class* lain. Panah menunjukkan arah *query* antar *class*.
2) Aggregasi, yaitu hubungan yang menyatakan bahwa suatu *class* terdiri dari *class* lain.
3) Pewarisan, yaitu hubungan hirarkis antar *class. Class* dapat diturunkan dari *class* lain dan mewarisi semua atribut dan metode *class* asalnya dan menambahkan fungsionalitas baru, sehingga ia disebut anak dari *class* yang diwarisinya. Kebalikan dari pewarisan adalah generalisasi.
4) Hubungan dinamis, yaitu rangkaian pesan(*message*) yang dikirim dari satu *class* ke *class* lain. Hubungan dinamis dapat degambarkan juga dengan menggunakan *sequence diagram.*

c. *Statechart Diagram*

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

Menurut Whitten, Bentley, dan Dittman (2004, p419), diagram ini digunakan untuk memodelkan behavior objek khusus yang dinamis. Diagram ini mengilustrasikan siklus hidup objek-berbagai-keadaan yang dapat diasumsikan oleh objek dan *event-event* yang menyebabkan objek beralih dari satu *state* ke *state* lain.

Dalam UML, *state* digambarkan berbentuk segiempat dengan sudut membulat dan memiliki nama sesuai dengan kondisinya saat itu. Transisi antar *state* umumnya memiliki kondisi *guard* yang merupakan syarat terjadinya transisi yang bersangkutan dituliskan dalam kurung siku. Pekerjaan yang dilakukan sebagai akibat dari *event* tertentu dituliskan dengan diawali garis miring. Titik awal dan akhir digambarkan berbentuk lingkaran berwarna penuh dan berwarna setengah.

d. *Activity diagram*

*Activity diagram* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

*Activity diagram* merupakan *state diagram* khusus, dimana sebagian besar *state* adalah *action* dan sebagian besar transisi dipicu oleh selesainya *state* sebelumnya *(internal processing).* Oleh karena itu, *statechart diagram* tidak menggambarkan kelakuan internal sebuah sistem(dan interaksi antar subsistem) secara eksak, tapi lebih menggambarkan proses-proses, dan jalur-jalur aktivitas dan dari level atas secara umum.

Menurut Whitten, Bentley, dan Dittman (2004,p419), diagram ini secara grafis digunakan untuk menggambarkan rangkaian aliran aktivitas baik proses bisnis maupun usecase. Diagram ini juga dapat digunakan untuk memodelkan action yang akan dilakukan saat sebuah operasi dieksekusi, dan memodelkan hasil dari action tersebut.

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara use case menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktifitas.

Tata cara penggambaran dalam UML hampir sama dengan penggambaran *state diagram*, menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan kelakuan pada kondisi tertentu. Untuk mengilustrasikan proses-proses paralel(*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal.

e. *Sequence diagram*

*Sequence diagram* menggambarkan interaksi antar objek di dalan dan di sekitar sistem(termasuk pengguna, *display*, dansebagainya)berupa pesan (*message*) yang digambarkan terhadap waktu. *Sequence diagram* terdiri atas dimensi vertikal(waktu) dan dimensi horizontal(objek-objek yang terkait).

Menurut Whitten, Bentley, dan Dittman (2004,p419), diagram ini menggambarkan secara grafis, bagaimana objek berinteraksi dengan satu sama lain melalui pesan pada eksekusi sebuah *use case* atau operasi. Diagram ini mengilustrasikan bagaimana pesan terkirim dan diterima di antara objek dan dalam sekuensi apa.

*Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah event untuk menghasilkan output tertentu. Diawali dari apa yang memicu aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal, dan output apa yang dihasilkan.

Masing-masing objek, termasuk aktor, memiliki garis hidup vertikal. Pesan(message) digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase disain berikutnya, pesan akan dipetakan menjadi metode dari *class*. *Activation* baru menunjukkan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah proses.

f. *Collaboration diagram*

Hampir sama dengan *sequence diagram*, *collaboration diagram* juga menggambarkan interaksi antar objek, tetapi lebih menekankan pada peran masing-masing objek dan bukan pada waktu penyampaian pesan. Setiap pesan memiliki nomer urut dimana pesan dari level tertinggi memiliki nomor 1. Pesan dari level yang sama memiliki prefix yang sama.

Menurut Whitten, Bentley, dan Dittman (2004, p419), diagram ini serupa dengan diagram sekuensi, tetapi tidak fokus pada *timing* atau sekuensi pesan. Diagram ini malah menggambarkan interaksi (*collaboration*) antara objek dalam sebuah format jaringan.

g. *Component diagram*

*Component diagram* menggambarkan struktur dan hubungan antar komponen piranti lunak termasuk ketergantungan diantaranya. Komponen piranti lunak adalah modul berisi kode program, baik berisi *source code* maupun *binary code*, baik *library* maupun *executable*, baik yang muncul pada *compile time, link time* maupun *run time*.

Menurut Whitten, Bentley, dan Dittman (2004, p419), diagram ini digunakan untuk menggambarkan organisasi dan ketergantungan komponen-komponen *software system*. Diagram ini digunakan untuk menunjukkan bagaimana kode pemrograman dibagi sebagai modul-modul(komponen).

Umumnya komponen terbentuk dari beberapa *class*, tapi dapat juga dari komponen-komponen yang lebih kecil. Komponen juga dapat berupa *interface*, yaitu kumpulan layanan yang disediakan oleh sebuah komponen untuk komponen lain.

h. *Deployment diagram*

*Deployment/physical* diagram menggambarkan detail bagaimana komponen dipasang dalam infrastruktur sistem, dimana komponen akan terletak pada mesin, *server*, atau piranti keras lainnya, bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi *server*, dan hal-hal lainnya yang bersifat fisik.

Menurut Whitten, Bentley, dan Dittman (2004, p419), diagram ini mendeskripsikan arsitektur fisik dalam istilah node untuk *hardware* dan *software* dalam

sistem. Diagram ini menggambarkan konfigurasi komponen-komponen *software run-time*, *prosessor* dan peralatan yang membentuk arsitektur sistem.

Sebuah node adalah *server, workstation*, atau piranti keras lain yang digunakan untuk memasang komponen dalam lingkungan sebenarnya. Hubungan antar node(misalnya TCP/IP) dan kebutuhan node juga didefinisikan dalam diagram ini.

### 2.2.2 *Structured Query Language (SQL)*

Menurut Wikipedia**,** *Microsoft SQL Server* adalah sebuah sistem manajemen basis data relasional (RDBMS) produk *Microsoft*. Bahasa kueri utamanya adalah *Transact-SQL* yang merupakan implementasi dari SQL standar ANSI/ISO yang digunakan oleh *Microsoft* dan *Sybase*. Umumnya *SQL Server* digunakan di dunia bisnis yang memiliki basis data berskala kecil sampai dengan menengah, tetapi kemudian berkembang dengan digunakannya *SQL Server* pada basis data besar.

Microsoft SQL Server dan *Sybase/ASE* dapat berkomunikasi lewat jaringan dengan menggunakan protokol TDS (*Tabular Data Stream*). Selain dari itu, Microsoft SQL Server juga mendukung ODBC (*Open Database Connectivity*), dan mempunyai driver JDBC untuk bahasa pemrograman Java. Fitur yang lain dari SQL Server ini adalah kemampuannya untuk membuat basis data *mirroring dan clustering*. Pada versi sebelumnya, MS SQL Server 2000 terserang oleh cacing komputer SQL Slammer yang mengakibatkan kelambatan akses Internet pada tanggal 25 Januari 2003.

***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.

***CREATE***

`CREATE` digunakan untuk membuat basis data maupun objek-objek basis data. SQL yang umum digunakan adalah:

```
CREATE DATABASE
nama_basis_data
```

`CREATE DATABASE` membuat sebuah basis data baru.

```
CREATE TABLE nama_tabel
```

`CREATE TABLE` membuat tabel baru pada basis data yang sedang aktif. Secara umum, perintah ini memiliki bentuk

*CREATE TABLE* [*nama_tabel*]

(

nama_field1 tipe_data [constraints][,

nama_field2 tipe_data,

...]

)

atau

```
CREATE TABLE [nama_tabel]
(
nama_field1 tipe_data [,
nama_field2 tipe_data,
...]
[CONSTRAINT nama_field constraints]
)
```

dengan:

*nama_field* adalah nama kolom (*field*) yang akan dibuat. Beberapa sistem manajemen basis data mengizinkan penggunaan spasi dan karakter nonhuruf pada nama kolom.

*tipe_data* tergantung implementasi sistem manajemen basis data. Misalnya, pada MySQL, tipe data dapat berupa `VARCHAR`, `TEXT`, `BLOB`, `ENUM`, dan sebagainya.

*constraints* adalah batasan-batasan yang diberikan untuk tiap kolom. Ini juga tergantung implementasi sistem manajemen basis data, misalnya `NOT NULL`, `UNIQUE`, dan sebagainya. Ini dapat digunakan untuk mendefinisikan kunci primer (*primary key*) dan kunci asing (*foreign key*).

Satu tabel boleh tidak memiliki kunci primer sama sekali, namun sangat disarankan mendefinisikan paling tidak satu kolom sebagai kunci primer.

Contoh:

```
CREATE TABLE user
(
username VARCHAR(30) CONSTRAINT PRIMARY KEY,
passwd VARCHAR(20) NOT NULL,
tanggal_lahir DATETIME
);
```

akan membuat tabel user seperti berikut:

| **username** | **passwd** | **tanggal_lahir** |
|---|---|---|

***Data Manipulation Language***

DML digunakan untuk memanipulasi data yang ada dalam suatu tabel. Perintah yang umum dilakukan adalah:

- `SELECT` untuk menampilkan data
- `INSERT` untuk menambahkan data baru
- `UPDATE` untuk mengubah data yang sudah ada
- `DELETE` untuk menghapus data

***SELECT***

`SELECT` adalah perintah yang paling sering digunakan pada SQL, sehingga terkadang istilah *query* dirujukkan pada perintah `SELECT`. `SELECT` digunakan untuk menampilkan data dari satu atau lebih tabel, biasanya dalam sebuah basis data yang sama. Secara umum, perintah `SELECT` memiliki bentuk lengkap:

```
SELECT [nama_tabel|alias.]nama_field1 [AS alias1] [, nama_field2, ...]
FROM nama_tabel1 [AS alias1] [INNER|LEFT|RIGHT JOIN tabel2 ON kondisi_penghubung]
[, nama_tabel3 [AS alias3], ...]
[WHERE kondisi]
[ORDER BY nama_field1 [ASC|DESC][, nama_field2 [ASC|DESC], ...]]
[GROUP BY nama_field1[, nama_field2, ...]]
[HAVING kondisi_aggregat]
```

dengan:

- *kondisi* adalah syarat yang harus dipenuhi suatu data agar ditampilkan.
- *kondisi_aggregat* adalah syarat khusus untuk fungsi aggregat.

Kondisi dapat dihubungkan dengan operator logika, misalnya AND, OR, dan sebagainya.

Contoh:

Diasumsikan terdapat tabel user yang berisi data sebagai berikut.

| **Username** | **passwd** | **tanggal_lahir** | **jml_transaksi** | **total_transaksi** |
|---|---|---|---|---|
| Aris | 6487AD5EF | 09-09-1987 | 6 | 10.000 |
| Budi | 97AD4erD | 01-01-1994 | 0 | 0 |
| Charlie | 548794654 | 06-12-1965 | 24 | 312.150 |
| Daniel | FLKH947HF | 24-04-1980 | 3 | 0 |
| Erik | 94RER54 | 17-08-1945 | 34 | 50.000 |

Contoh 1: Tampilkan seluruh data.

```
SELECT * FROM user
```

Contoh 2: Tampilkan pengguna yang tidak pernah bertransaksi.

```
SELECT * FROM user
WHERE total_transaksi = 0
```

Contoh 3: Tampilkan username pengguna yang bertransaksi kurang dari 10 dan nilainya lebih dari 1.000.

```
SELECT username FROM user
WHERE jml_transakai < 10 AND
total_transaksi > 1000
```

Contoh 4: Tampilkan total nominal transaksi yang sudah terjadi.

```
SELECT SUM(total_transaksi) AS
total_nominal_transaksi
FROM user
```

Contoh 5: Tampilkan seluruh data diurutkan berdasarkan jumlah transaksi terbesar ke terkecil.

```
SELECT * FROM user
ORDER BY jml_transaksi DESC
```

**Fungsi aggregat**

Beberapa SMBD memiliki fungsi *aggregat*, yaitu fungsi-fungsi khusus yang melibatkan sekelompok data (*aggregat*). Secara umum fungsi *aggregat* adalah:

- `SUM` untuk menghitung total nominal data

- `COUNT` untuk menghitung jumlah kemunculan data
- `AVG` untuk menghitung rata-rata sekelompok data
- `MAX` dan `MIN` untuk mendapatkan nilai maksimum/minimum dari sekelompok data.

Fungsi aggregat digunakan pada bagian `SELECT`. Syarat untuk fungsi aggregat diletakkan pada bagian `HAVING`, bukan `WHERE`.

***Subquery***

Ada kalanya *query* dapat menjadi kompleks, terutama jika melibatkan lebih dari satu tabel dan/atau fungsi aggregat. Beberapa SMBD mengizinkan penggunaan *subquery*. Contoh:

Tampilkan username pengguna yang memiliki jumlah transaksi terbesar.

```
SELECT username FROM user
WHERE jml_transaksi =
(
SELECT MAX(jml_transaksi)
FROM user
)
```

***INSERT***

Untuk menyimpan data dalam tabel dipergunakan sintaks:

```
INSERT INTO [NAMA_TABLE] ([DAFTAR_FIELD]) VALUES ([DAFTAR_NILAI])
```

Contoh:

```
INSERT INTO TEST (NAMA, ALAMAT, PASSWORD) VALUES ('test', 'alamat', 'pass');
```

***UPDATE***

Untuk mengubah data menggunakan sintaks:

```
UPDATE [NAMA_TABLE] SET [NAMA_KOLOM]=[NILAI] WHERE [KONDISI]
```

Contoh:

```
UPDATE Msuser set password="123456" where username="abc"
```

***DELETE***

Untuk menghapus data dipergunakan sintaks:

```
DELETE FROM [NAMA_TABLE] [KONDISI]
```

Contoh:

```
DELETE FROM TEST WHERE NAMA='test';
```

### *2.2.3 Active Server Pages .NET (ASP.NET)*

Menurut Wikipedia, ASP.NET adalah sebuah teknologi layanan *Web* dinamis, aplikasi *web*, dan XML *web service* yang dikembangkan oleh Microsoft sebagai pengganti *Active Server Pages* (*ASP*) yang telah lama. Teknologi ini berbasis *.NET Framework* dan dibangun di atas *Common Language Runtime* (CLR), sehingga para *programmer* dapat menulis kode ASP.NET dengan menggunakan semua bahasa pemrograman .NET, meski yang populer digunakan adalah bahasa C# dan Visual Basic .NET.

- ASP.NET adalah kerangka aplikasi web yang dikembangkan dan dipasarkan oleh Microsoft, yang mana *programmer* dapat mengunakannya untuk membangun situs *web* dinamis, aplikasi *web* dan layanan *web*.
- Pertama dirilis pada Januari 2002 dengan versi 1.0 dari. *NET Framework.*
- Sebagai pengganti teknologi Microsoft *Active Server Pages* (ASP).
- ASP.NET dibangun di atas *Common Language Runtime* (CLR), sehingga memungkinkan para *programmer* untuk menulis kode ASP.NET dengan menggunakan bahasa . NET.
- Sejarah:
    - Pada tahun 1997, penelitian untuk model aplikasi *web* baru untuk memecahkan keluhan umum tentang *Active Server Pages,* terutama yang berkaitan dengan pemisahan presentasi dan isi dan kemampuan untuk menulis kode "bersih”.

- Penelitian dilakukan oleh Mark Anders, seorang manajer di Tim IIS dan Scott Guthrie, lulusan Duke University yang bergabung dengan Microsoft pada tahun 1997, desain awal diselesaikan dalam waktu dua bulan sekitar liburan natal pada tahun 1997.

- Prototipe awal disebut “XSP”, pengembangan awal dilakukan menggunakan *Java*, tetapi tidak lama kemudian memutuskan untuk membangun platform baru di atas *Common Language Runtime* (CLR) sebagai gantinya.

- Dengan pindah ke *Common Language Runtime*, XSP ini kembali dilaksanakan di C# ( "*Project Cool*") dan diubah namanya menjadi ASP+, dengan titik ini *platform* baru dianggap sebagai pengganti *Active Server Pages*, dan maksudnya adalah untuk menyediakan jalur migrasi yang mudah bagi pengembang ASP.

- ASP+ ditunjukkan pertama kali pada *ASP Connections conference* di Phoenix, Arizona pada tanggal 2 Mei 2000. Awal rilis beta dari ASP+ pada 2000 *Professional Developers Conference* pada 11 Juli 2000 di Orlando, Florida.

- Sewaktu presentasi Bill Gates, Fujitsu menunjukkan ASP+ yang digunakan bersama-sama dengan COBOL, dan bahasa lainnya seperti Microsoft Visual Basic. NET yang baru, dan C#, serta *Python* dan *Perl* yang di dukung melalui alat interoperabilitas yang dibuat oleh *ActiveState*.

- Begitu merek ". NET" ditetapkan pada paruh kedua tahun 2000, maka diputuskan untuk mengubah nama ASP+ menjadi ASP.NET.

- .NET History: