Bab II

Tinjauan Pustaka

II.1 Tinjauan Tempat Penelitian

II.1.1 Sejarah Perusahaan

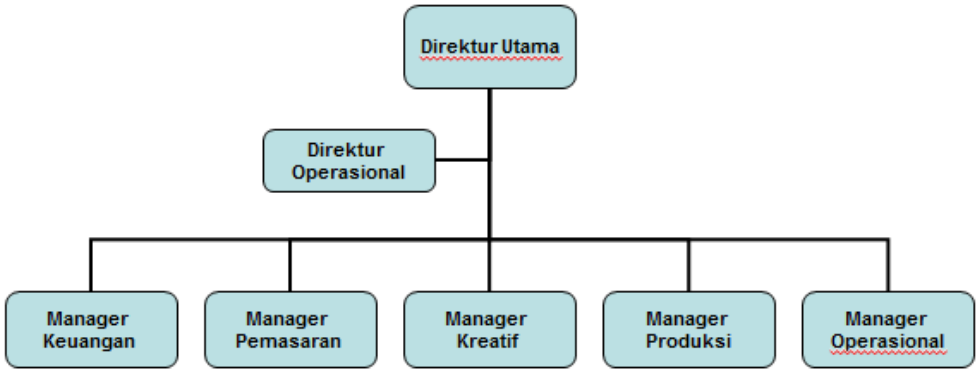
CV. *Cure-Hate Production* merupakan salah satu usaha yang bergerak dalam bidang produksi dan penjualan barang-barang konveksi, kerajinan lukis sepatu, serta alat-alat elektronik. Perusahaan ini berdiri secara resmi sejak tahun 2009, dan telah terdaftar sebagai Perseroan Komanditer yang memiliki kantor pusat di daerah Pemalang, Jawa Tengah. Untuk memperluas jangkauan pemasaran dan distribusi produk, CV. *Cure-Hate Production* telah mendirikan beberapa cabang berupa distro di kota Bandung, Jakarta, dan Lampung. Dalam proses kegiatan usahanya, perusahaan ini dikelola oleh beberapa pengurus usaha yang dibagi menjadi ke dalam beberapa divisi diantaranya adalah; divisi desain, divisi produksi, divisi pemasaran, dan divisi penjualan. Kantor pusat usaha ini terletak di Jl. Nusabarung IV No. 32, Perumnas Bojong Bata, Pemalang Jawa Tengah.

II.1.2 Visi dan Misi Perusahaan

Visi dari CV. *Cure-Hate Production* adalah "Menjadi perusahaan konveksi terdepan dan terpandang yang menghasilkan barang-barang konveksi yang unggul dalam kualitas bagi konsumen, yang berhubungan dengan bidang promosi"

Misi dari CV. *Cure-Hate Production* adalah "Berjuang dalam mengembangkan dan meningkatkan sumber daya yang tepat guna untuk dapat menghasilkan produk konveksi yang bermutu tinggi dan konsisten serta harga yang bersaing di pasar".

II.1.3 Struktur Organisasi



Gambar II.1 Struktur Organisasi Perusahaan CV. *Cure-Hate Production*

Dengan melihat struktur organisasi perusahaan pada CV. *Cure-Hate Production* pada gambar II.1, setiap jabatan memiliki beberapa deskripsi tugas yang harus dilakukan dan dipertanggungjawabkan yaitu:

1. Direktur Utama yang juga berperan sebagai Direktur Perusahaan dan juga pemilik usaha PT. CV *Cure-Hate Production*
   a. Memimpin keseluruhan dewan atau komite ekeskutif.
   b. Menawarkan visi dan imajinasi di tingkat tertinggi.
   c. Memimpin rapat umum dalam hal menentukan agenda, menjelaskan dan menyimpulkan tindakan dan kebijakan.
   d. Bertindak sebagai perwakilan organisasi dalam hubungannya dengan dunia luar.

e. Menjalankan tanggung jawab sesuai dengan standar etika, hukum, dan dokumen kebijakan direktur.
f. Mengumpulkan data permintaan dari pihak luar yang akan dikerjakan.
g. Mengangkat karyawan atau tenaga kerja demi kepentingan perusahaan.

2. Direktur Operasional
   a. Melakukan perencanaan, pengelolaan, pengawasan, dan pengendalian dalam bidang operasional secara sistematis.
   b. Menyusun kebijakan dan strategi pencapaian target perusahaan.
3. Manajer Keuangan
   a. Melakukan perencanaan, pengelolaan, pengawasan, dan pengendalian dalam bidang keuangan dan akuntansi secara sistematis.
   b. Membangun sinergi dan berusaha mencapai hasil bisnis yang optimal dari pelaksanaan seluruh kegiatan perusahaan.
   c. Memastikan ketersediaan dana operasional yang dibutuhkan oleh perusahaan untuk kegiatan operasional sehari-hari.
   d. Memastikan konsolidasi keuangan yang akurat dan tepat waktu untuk keperluan pelaporan keuangan kepada Direktur Perusahaan.
   e. Mengkoordinir perumusan strategi jangka panjang sebagai dasar perumusan Rencana Kerja dan Anggaran Perusahaan (RKAP).
   f. Memberlakukan dan mengambil keputusan yang dapat mengurangi dan menanggulangi berbagai resiko finansial yang dapat dihadapi oleh perusahaan dengan berkoordinasi dengan Direksi lainnya.

4. Manajer *Marketing*
    a. Menentukan harga jual dari produk yang dihasilkan untuk memastikan tercapainya target penjualan.
    b. Menganalisa dan mengembangkan strategi pemasaran untuk meningkatkan jumlah pelanggan dan area sesuai dengan target yang ditentukan.
    c. Melakukan evaluasi kepuasan pelanggan dari hasil survei, untuk memastikan tercapainya target kepuasan pelanggan yang ditentukan.
    d. Menganalisa dan memberikan arahan pengembangan desain dan warna, untuk memastikan pengembangan produk sesuai dengan kebutuhan pasar.
    e. Mengawasi jumlah stok yang dipasarkan untuk memastikan umur dari stok produk tidak melebihi dari target yang telah ditentukan.
5. Manajer Kreatif
    a. Memimpin divisi kreatif yang melakukan kegiatan desain produk untuk memastikan desain yang dihasilkan sesuai dengan permintaan pasar.
    b. Memberikan merek dari desain produk yang dihasilkan.
    c. Menjaga integritas merek produk yang dihasilkan oleh perusahaan
    d. Memastikan bahwa aspek visual setiap produk baru harus konsisten dan sesuai dengan standar merek dari perusahaan.
    e. Memperbaharui merek produk yang dihasilkan sesuai dengan strategi dan teknologi baru yang digunakan.

6. Manajer Produksi
    a. Menyusun rencana dan jadwal produksi.
    b. Mengkoordinir dan mengawasi, serta memberikan pengarahan kerja kepada setiap divisi di bawahnya untuk menjamin terlaksananya kesinambungan dalam proses produksi.
    c. Bertanggungjawab atas pengendalian bahan baku, dan efisiensi penggunaan tenaga kerja, mesin, dan peralatan.
    d. Membuat laporan harian dan berkala mengenai kegiatan produksi sesuai dengan sistem pelaporan yang berlaku.
    e. Berusaha mencari strategi penekanan biaya produksi dan metode perbaikan kerja yang efisien.
    f. Bertanggungjawab dalam pemenuhan standar kualitas produksi yang sesuai dengan kebutuhan pasar.
    g. Bertanggungjawab terhadap keselamatan kerja, dan standar kebersihan lingkungan kerja.
7. Manajer Operator
    a. Bertanggungjawab atas pemantauan produktivitas di bagian penjualan.
    b. Membuat laporan harian secara tertulis kepada Direktur Utama mengenai seluruh kegiatan operasional.
    c. Seluruh keluhan dari pelanggan adalah tanggung jawab dari Manajer Operator.
    d. Bertanggungjawab dalam pelatihan berkelanjutan untuk karyawan baru.

II.2 Sistem

II.2.1 Konsep Dasar Sistem

Kata sistem mempunyai beberapa pengertian, tergantung dari sudut pandang mana kata tersebut didefinisikan. Secara garis besar ada dua kelompok pendekatan sistem, yaitu: pendekatan sistem yang lebih menekankan pada elemen-elemen atau kelompoknya didefinisikan sebagai "Suatu jaringan kerja dari prosedur prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau menyelesaikan suatu aturan tertentu.

Pendekatan sistem sebagai jaringan kerja dari prosedur, yang lebih menekankan urutan operasi didalam sistem. didefinisikan sebagai "urutan operasi kerja *(tulis-menulis)*, yang biasanya melibatkan beberapa orang di dalam satu atau lebih departemen, yang diterapkan untuk menjamin penanganan yang seragam dari transaksi bisnis yang terjadi. Suatu sistem mempunyai karakteristik atau sifat-sifat yang tertentu, yaitu mempunyai komponen sistem *(component)*, batasan sistem *(boundary)*, lingkungan luar sistem *(environment)*, penghubung sistem *(interface)*, masukan sistem *(input)*, keluaran sistem *(output)*, pengolahan sistem *(process)* dan sasaran *(objective)* atau tujuan *(goal)*.

II.2.1.1 Pengertian Sistem

Sistem adalah sekumpulan unsur / elemen yang saling berkaitan dan saling mempengaruhi dalam melakukan kegiatan bersama untuk mencapai suatu tujuan. Dari pengertian diatas dapat diambil suatu kesimpulan bahwa Sistem adalah setiap

kesatuan secara konseptual atau fisik yang terdiri dari bagian-bagian dalam keadaan saling tergantung satusama lainnya.

II.2.1.2 Karakteristik Sistem

Suatu sistem memiliki karakteristik atau sifat-sifat tertentu, yaitu memiliki komponen-komponen *(components)*, batas sistem *(boundary)*, lingkungan luar sistem *(environment)*, penghubung *(interprest)*, masukan *(input)*, keluaran *(output)*, pengolah *(process)* dan sasaran *(objective)* dan tujuan *(goal)*.

1. Komponen Sistem (*System Components*)

   Komponen sistem atau elemen-elemen sistem dapat berupa suatu sub sistem atau bagian-bagian dari sistem. Setiap sistem baik besar maupun kecil, selalu mengandung komponen-komponen atau subsistem-subsistem. Setiap subsistem mempunyai sifat-sifat dari sistem untuk menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan. Suatu sistem dapat mempunyai suatu yang lebih besar yang disebut *supra system.*

2. Batas Sistem (*System Boundary*)

   Batas sistem merupakan daerah-daerah yang membatasi antara satu sistem dengan sistem lainnya dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan. Batas suatu sistem menujukkan ruang lingkup (*scope*) dari sistem tersebut.

3. Lingkungan Luar Sistem (*System Environment*)

   Lingkungan luar sistem dari suatu sistem adalah apapun di luar batas dari sistem yang mempengaruhi operasi sistem yang dapat bersifat

menguntungkan dan dapat pula merugikan sistem tersebut. Lingkungan luar yang menguntungkan merupakan energi dari sistem yang harus dijaga dan dipelihara. Sedangkan yang merugikan harus ditahan dan dikendalikan, karena akan mengganggu kelangsungan hidup sistem.

4. Penghubung Sistem

   Penghubung sistem merupakan media penghubung antara satu sub sistem dengan subsistem yang lainnya. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem yang lainnya. Dengan penghubung akan terjadi interaksi antar subsistem, sehingga membentuk satu kesatuan.

5. Masukan Sistem *(System Input)*

   Masukan adalah suatu energi yang dimasukkan ke dalam sistem. Masukan dapat berupa masukan perawatan (*maintenance input*) dan masukan sinyal (*signal input*). *Maintenance input* adalah energi yang dimasukkan supaya sistem tersebut dapat beroperasi. Contoh *maintenance input* di dalam sistem komputer adalah program, yang digunakan untuk mengoperasikan komputer. Sedangkan *signal input* adalah energi yang diproses untuk mendapatkan keluaran. Contoh *signal input* di dalam sistem komputer adalah data, yang dapat diolah menjadi Informasi.

6. Keluaran Sistem *(System Output)*

   Keluaran (*Output*) merupakan hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna.

7. Pengolah Sistem (*System Process*)

   Suatu sistem dapat mempunyai suatu bagian pengolah atau sistem itu sendiri sebagai pengolahnya, yang bertugas untuk merubah masukan menjadi keluaran.

8. Sasaran Sistem (*System Objective*)

   Suatu sistem pasti memiliki tujuan (*goal*) atau sasaran (*objective*). Suatu operasi sistem akan berguna dan berhasil apabila mencapai sasaran atau tujuannya. Sasaran sistem sangat menentukan masukan yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem.

II.3 Data

II.3.1 Pengertian Data

Data adalah kumpulan dari fakta-fakta, kejadian-kejadian yang dapat berupa simbol, angka, huruf, dan lain-lain yang berguna bagi suatu pengolahan data (process) atau sebagai masukan (*input*) bagi suatu proses.

II.3.2 Model Data

Data yang disimpan menggambarkan beberapa aspek dari suatu organisasi. Model data adalah himpunan deksripsi data level tinggi yang dikonstruksi untuk menyembunyikan beberapa detail dari penyimpanan level rendah. Beberapa manajemen basis data didasarkan pada model data relasional, model data hirarkis, atau model data jaringan.

II.3.2.1 Model Data Hirarkis

Model hirarkis biasa disebut model pohon, karena menyerupai pohon yang dibalik. Model ini menggunakan pola hubungan orang tua-anak. Setiap simpul (biasa dinyatakan dengan lingkaran atau kotak) menyatakan sekumpulan medan. Simpul yang terhubung ke simpul pada level di bawahnya disebut orang tua. Setiap orang tua bisa memiliki satu (hubungan 1:1) atau beberapa anak (hubungan 1:M), tetapi setiap anak hanya memiliki satu orang tua. Simpul – simpul yang dibawahi oleh simpul orang tua disebua anak. Simpul orang tua yang tidak memiliki orang tua disebut akar. Simpul yang tidak mempunyi anak disebut daun. Adapun hubungan antara anak dan orang tua disebut cabang.

II.3.2.2 Model Data Jaringan

Model jaringan distandarisasi pada tahun 1971 oleh *Data Base Task Group* (DBTG). Itulah sebabnya disebut model DBTG. Model ini juga disebut model CODASYL (*Conference on Data System Languages*), karena DBTG adalah bagian dari CODASYL.

Model ini menyerupai model hirarkis, dengan perbedaan suatu simpul anak bisa memilki lebih dari satu orang tua. Oleh karena sifatnya demikian, model ini menyatakan hubungan 1:1 (satu arang tua punya satu anak), 1:N (satu orang tua punya banyak anak), maupun N:N (beberapa anak bisa mempunyai beberapa orangtua). Pada model jaringan, orang tua diseut pemilik dan anak disebut anggota.

II.3.2.3 Model Data Relasional

Model relasional adalah model data yang paling banyak digunakan saat ini. Pembahasan pokok pada model ini adalah relasi, yang dimisalkan sebagai himpunan dari record. Deskripsi data dalam istilah model data disebut skema. Pada model relasional, skema untuk relasi ditentukan oleh nama, nama dari tiap *field* (atau atribut atau kolom), dan tipe dari tiap *field*.

II.4 Konsep Dasar Informasi

Informasi *(information)* dapat didefinisikan sebagai berikut. Informasi adalah data yang sudah diolah menjadi sebuah bentuk yang lebih berguna dan lebih berarti bagi yang menerimanya atau pengguna. Sumber dari informasi adalah data. Data merupakan bentuk jamak dari bentuk tunggal datum atau item-item. Data adalah kenyataan yang menggambarkan suatu kejadian-kejadian dan kesatuan nyata. Kejadian-kejadian *(event)* adalah sesuatu yang terjadi pada saat yang tertentu.

II.4.1 Kualitas Informasi

Kualitas dari suatu informasi tergantung pada tiga hal pokok yaitu:

1. Akurat (*accurate*)

   Informasi harus bebas dari kesalahan-kesalahan dan tidak menyesatkan, dalam hal ini informasi harus jelas mencerminkan maksudnya.

2. Tepat waktu (*time lines*)

   Informasi yang datang pada penerima tidak boleh terlambat. Informasi yang sudah usang tidak akan mempunyai nilai lagi karena informasi

merupakan suatu landasan dalam mengambil sebuah keputusan dimana bila pengambilan keputusan terlambat maka akan berakibat fatal untuk organisasi.

3. Relevan (*relevance*)

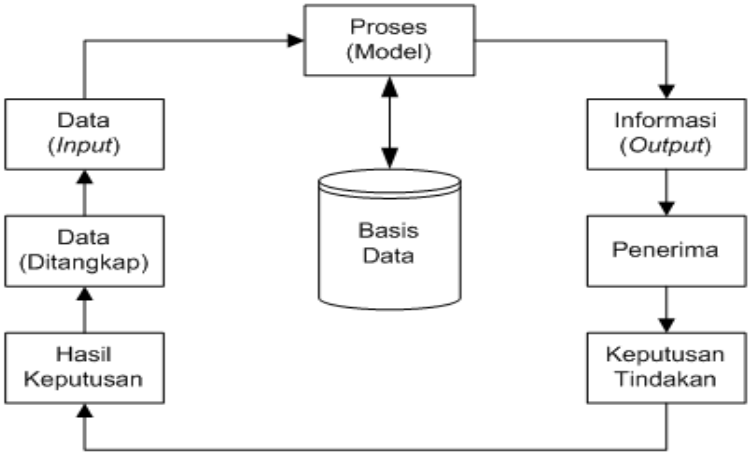
   Informasi harus mempunyai manfaat untuk pemakainya, dimana relevansi informasi untuk tiap-tiap individu berbeda tergantung pada yang menerima dan yang membutuhkan.

II.4.2 Nilai Informasi

Nilai Informasi (*value of information*) ditentukan oleh dua hal yaitu manfaat dan biaya. Suatu informasi dikatakan bernilai bila manfaatnya lebih efekif dibandingkan dengan biaya mendapatkannya.

II.4.3 Siklus Informasi

Data agar menjadi lebih berarti dan berguna dalam bentuk Informasi, maka perlu diolah melalui suatu model tertentu. Data yang telah diolah tersebut kemudian diterima oleh penerima, lalu penerima membuat suatu keputusan dan melakukan tindakan, yang berarti menghasilkan suatu tindakan yang lain yang akan membuat sejumlah data kembali. Data tersebut akan ditangkap sebagai *input*, dan diproses kembali lewat suatu model dan seterusnya sehingga membentuk suatu siklus. Siklus ini disebut dengan siklus Informasi (*information cycle*) atau disebut pula siklus pengolahan data (*processing cycles*) yang dapat dilihat pada gambar II.2.



Gambar II.2 Siklus Informasi

II.5 Konsep Dasar Sistem Informasi

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

Dan Sistem Informasi dapat diartikan juga sebagai kumpulan dari sub-sub sistem komponen baik phisik maupun non phisik yang saling berhubungan satu sama lain dan bekerjasama secara harmonis untuk mencapai satu tujuan yaitu mengolah data menjadi informasi yang berguna

Dari definisi diatas dapat diambil kesimpulan bahwa sistem informasi merupakan perpaduan antara manusia, alat teknologi, media, prosedur dan pengendalian yang bertujuan untuk menata jaringan komunikasi sehingga dapat membantu dalam pengambilan keputusan yang tepat. Kegiatan yang terdapat pada sistem informasi antara lain :

1. *Input*, menggambarkan suatu kegiatan untuk menyediakan data yang akan diproses.
2. Proses, menggambarkan bagaimana suatu data diproses untuk menghasilkan suatu informasi yang bernilai tambah.
3. *Output*, suatu kegiatan untuk menghasilkan laporan dari proses diatas.
4. Penyimpanan, suatu kegiatan untuk memelihara dan menyimpan data.
5. Kontrol, suatu aktifitas untuk menjamin bahwa sistem informasi tersebut berjalan sesuai dengan yang diharapkan.

Sistem informasi dalam sebuah sistem meliputi pemasukan data (*input*) kemudian diolah melalui suatu model dalam pemrosesan data, dan hasil informasi akan ditangkap kembali sebagai suatu input dan seterusnya sehingga membentuk siklus informasi yang dapat diperoleh dari sistem informasi sebagai sistem khusus dalam organisasi untuk mengolah informasi tersebut.

II.5.1 Komponen Sistem Informasi

Sistem informasi terdiri dari komponen-komponen yang disebut dengan istilah blok bangunan (*building block*) yaitu:

1. *Hardware* yaitu suatu perangkat keras dalam komputer yang kita bisa sentuh dan rasakan.
2. *Software* yaitu suatu perangkat lunak di dalam komputer yang berfungsi untuk mengoperasikan suatu aplikasi di dalam sistem komputer.
3. Data yaitu sekumpulan karakter yang diterima sebagai masukan (*input*) untuk sistem informasi dan disimpan serta diolah.

4. Prosedur yaitu suatu urutan pekerjaan tata usaha yang biasanya melibatkan beberapa orang dalam satu bagian atau lebih, dan disusun untuk menjamin adanya perlakuan yang seragam terhadap transaksi yang terjadi.
5. *User* yaitu orang yang terlibat dalam sistem informasi seperti operator, pemimpin sistem informasi, dan sebagainya.

II.5.2 Tujuan Sistem Informasi

Sistem Informasi memiliki beberapa tujuan, yaitu:

1. Integrasi sistem
   a. Menghubungkan sistem individu/kelompok
   b. Pengkolektifan data dan penyambungan secara otomatis
   c. Peningkatan koordinasi dan pencapaian sinergi
2. Efisiensi pengelolaan
   a. Penggunaan basis data dalam upaya kesamaan pengadministrasian data
   b. Pengelolaan data berkaitan dengan karakteristik Informasi
   c. Penggunaan dan pengambilan Informasi
3. Dukungan keputusan untuk manajemen
   a. Melengkapi Informasi guna kebutuhan proses pengambilan kebutuhan
   b. Akuisisi Informasi eksternal melalui jaringan komunikasi
   c. Ekstraksi dari Informasi internal yang terpadu.

II.5.3 Manfaat Sistem Informasi

Sistem Informasi memiliki beberapa manfaat, yaitu:

1. Menghemat tenaga kerja.
2. Peningkatan efisiensi.
3. Mempercepat proses.
4. Perbaikan dokumentasi.
5. Pencapaian standar.
6. Perbaikan keputusan.

II.6 Electronic Commerce

Electronic Commerce (e-commerce) merupakan konsep baru yang biasa digambarkan sebagai proses jual beli barang atau jasa pada *World Wide Web* Internet atau proses jual beli atau pertukaran produk, jasa dan informasi melalui jaringan informasi termasuk internet. *E-Commerce* merupakan kegiatan bisnis yang dijalankan secara elektronik melalui suatu jaringan internet atau kegiatan jual beli barang atau jasa melalui jalur komunikasi digital.

II.6.1 Sejarah Perkembangan e-Commerce

Istilah *e-commerce* telah berubah sejalan dengan waktu. Awalnya, perdagangan elektronik berarti pemanfaatan transaksi komersial, seperti penggunaan EDI untuk mengirim dokumen komersial seperti pesanan pembelian atau *invoice* secara elektronik.

Kemudian dia berkembang menjadi suatu aktivitas yang mempunya istilah yang lebih tepat "perdagangan web" pembelian barang dan jasa melalui

WorldWide Web melalui *server* aman (HTTPS), protokol *server* khusus yang menggunakan enkripsi untuk merahasiakan data penting pelanggan.

Pada awalnya ketika web mulai terkenal di masyarakat pada 1994, banyak jurnalis memperkirakan bahwa *e-commerce* akan menjadi sebuah sektor ekonomi baru. Namun, baru sekitar empat tahun kemudian protokol aman seperti HTTPS memasuki tahap matang dan banyak digunakan. Antara 1998 dan 2000 banyak bisnis di AS dan Eropa mengembangkan situs web perdagangan ini.

II.6.2 Kelebihan e-Commerce

Secara sederhana, perbedaan antara proses perdagangan secara manual dengan menggunakan *e-commerce* dapat jelas terlihat, dimana pada proses dengan *e-commerce* terjadi efisiensi pada penggunaan fax, pencetakan dokumen, entry ulang dokumen, serta jasa kurir. Efisiensi tersebut akan menunjukkan pengurangan biaya dan waktu/kecepatan proses. Kualitas transfer data pun lebih baik, karena tidak dilakukan entry ulang yang memungkinkan terjadinya *human error*.

Secara ringkas *e-commerce* mampu menangani masalah berikut :

1. Otomatisasi, proses otomatisasi yang menggantikan proses manual. *(enerprise resource planning concept)*
2. Integrasi, proses yang terintegrasi yang akan meningkatkan efisiensi dan efektivitas proses. *(just in time concept)*
3. Publikasi, memberikan jasa promosi dan komunikasi atas produk dan jasa yang dipasarkan. *(electronic cataloging concept)*

4. Interaksi, pertukaran data atau informasi antar berbagai pihak yang akan meminimalkan *human error (electronic data interchange/EDI concept)*
5. Transaksi, kesepakatan antara 2 pihak untuk melakukan transaksi yang melibatkan institusi lainnya sebagai pihak yang menangani pembayaran. *(electronic payment concept)*

II.6.3 Kekurangan e-Commerce

Walaupun adanya e-commerce memberi banyak keuntungan, masih terdapat berbegai kekurangan dari e-commerce antara lain:

1. Bagi organisasi / perusahaan
   a. Keamanan sistem rentan diserang

      Terdapat sejumlah laporan mengenai website dan basis data yang dihack, dan berbagai lubang kelemahan keamanan dalam software. Hal ini dialami oleh sejumlah perusahaan besar seperti Microsoft dan lembaga perbankan. Masalah keamanan ini menjadi sangat pnting karena bila pihak lain yang tidak berwenang bisa menembus sistem maka dapat menghancurkan bisnis yang telah berjalan.

   b. Persaingan tidak sehat

      Di bawah tekanan untuk berinovasi dan membangun bisnis untuk memanfaatkan kesempatan yang ada dapat memicu terjadinya tindakan ilegal yaitu penjiplakan ide dan perang harga.

   c. Masalah kompabilitas teknologi lama dengan yang lebih baru

      Dengan perkembangan dan inovasi yang melahirkan teknologi baru, sering muncul masalah yaitu sistem bisnis yang lama tidak dapat

berkomunikasi dengan infrastruktur berbasis web dan internet. Hal ini memaksa perusahaan untuk menjalankan dua sistem independen yang tidak dapat saling berbagi, hal ini dapat mengakibatkan pembengkakan biaya.

2. Bagi konsumen
   a. Perlunya keahlian komputer

      Tanpa menguasai keahlian *computer*, mustahil konsumen dapat berpartisipasi dalam e-*commerce*. Pengetahuan dasar *computer* diperlukan, antara lain pengetahuan mengenai internet dan *web*.

   b. Biaya tambahan untuk mengakses internet

      Untuk ikut serta dalam e-commerce dibutuhkan koneksi internet yang tentu saja menambah pos pengeluaran bagi konsumen.

   c. Biaya peralatan komputer

      Komputer diperlukan untuk mengakses internet, tentu saja dibutuhkan biaya untuk mendapatkannya. Perkembangan komputer yang sangat pesat menyarankan konsumen untuk juga mengupdate peralatannya apabila tidak ingin ketinggalan teknologi.

   d. Resiko bocornya privasi dan data pribadi

      Segala hal mungkin terjadi saat konsumen mangakses internet untuk menjalankan e-*commerce*, termasuk risiko bocornya data pribadi karena ulah orang lain yang ingin membobol sistem.

   e. Berkurangnya waktu untuk berinteraksi secara langsung dengan orang lain.

Transaksi e-*commerce* yang berlangsung secara on-line telah mengurangi waktu konsumen untuk dapat melakukan proses sosial dengan orang lain. Hal ini tidak baik karena dikhawatirkan akan dapat mengurangi rasa kepedulian terhadap lingkungan sekitarnya.

f. Berkurangnya rasa kepercayaan karena konsumen berinteraksi hanya dengan komputer.

3. Bagi masyarakat

a. Berkurangnya interaksi antar manusia

Karena masyarakat lebih sering berinteraksi secara elektronik, dimungkinkan terjadi berkurangnya kemampuan sosial dan personal manusia untuk bersosialisasi dengan orang lain secara langsung.

b. Kesenjangan sosial

Terdapat bahaya potensial karena dapat terjadi kesenjangan sosial antara orang-orang yang memiliki kemampuan teknis dalam e-commerce dengan yang tidak, yang memiliki keahlian digaji lebih tinggi daripada yang tidak.

c. Adanya sumber daya yang terbuang

Munculnya teknologi baru akan membuat teknologi lama tidak dimanfaatkan lagi. Misalnya dengan komputer model lama atau *software* model lama yang sudah tidak relevan untuk digunakan.

d. Sulitnya mengatur internet

Sejumlah kriminalitas telah terjadi di internet dan banyak yang tidak terdeteksi. Karena jumlah jaringan yang terus berkembang semakin

luas dan jumlah pengguna yang semakin banyak, seringkali membuat pihak berwenang kesulitan dalam membuat peraturan untuk internet.

II.6.4 Manfaat e-Commerce

Manfaat yang diperoleh bagi organisasi pemilik *E-Commerce* :

1. Memperluas *market place* hingga kepasar nasional dan internasional.
2. Dengan capital outplay yang minim, sebuah perusahaaan dapat dengan mudah menemukan lebih banyak pelanggan, supplier yang lebih baik dan partner bisnis yang paling cocok dari seluruh dunia.
3. *E-Commerce* menurunkan biaya pembuatan, pemrosesan, pendistribusian, penyimpanan, dan pencarian informasi yang menggunakan kertas.
4. *E-Commerce* memungkinkan pengurangan *inventory* dan *overhead* dengan menyederhanakan *supply chain management* tipe "pull". Dalam *supply chain management* tipe "pull", prses dimulai dari pesana pelanggan serta digunakan *manufacturing just-in-time.*
5. *E-Commerce* mengurangi waktu antara *outlay modal* dan penerimaan produk dan jasa.
6. *E-Commerce* mendukung upaya-upaya *business process reengineering*. Dengan mengubah prosesnya, maka produktivitas *sales-people*, pegawai yang berpengetahuan, dan administrator bias meningkat 100% atau lebih.
7. *E-Commerce* memperkecil biaya telekomunikasi. Karena internet lebih murah dibandingkan VAN.
8. Akses informasi menjadi lebih cepat.
9. Biaya transportasi dan fleksibilitas bertambah.

Manfaat *E-Commerce* bagi konsumen antara lain:

1. *E-Commerce* memungkinkan pelanggan untuk berbelanja atau melakukan transaksi selama 24 jam sehari sepanjang tahun dari hamper setiap lokasi.
2. *E-Commerce* memberikan lebih banyak pilihan kepada pelanggan, mereka bias memilih berbagai produk dari banyak vendor.
3. *E-Commerce* menyediakan produk dan jasa yang tidak mahal kepada pelanggan dengan cara mengunjungi banyak tempat dan melakukan perbandingan secara cepat.
4. Pelanggan bias menerima informasi yang relevan secara detil dalam hitungan detik, bukan lagi hari atau minggu.
5. *E-Commerce* memberi tempat kepada para pelanggan untuk berinteraksi dengan pelanggan lain di *electronic community* dan bertukar pikiran serta pengalaman.

Manfaat *E-Commerce* bagi masyarakat antara lain:

1. *E-Commerce* memungkinkan orang untuk bekerja didalam rumah dan tidak harus keluar rumah untuk berbelanja. Ini berakibat menurunkan arus kepadatan lalu lintas dijalan serta mengurangi polusi udara.
2. *E-Commerce* memungkinkan seluruh masyarakat di berbagai wilayah perkotaan maupun pedesaan untuk menikmati aneka produk dan jasa yang mungkin susah untuk mereka dapatkan tanpa *E-Commerce.*

II.6.5 Klasifikasi e-Commerce

Penggolongan *E-Commerce* yang lazim dilakukan orang ialah berdasarkan sifat transaksinya, antara lain:

1. *Business to Business* (B2B)

   Jenis transaksi dimana pembeli biasanya membeli dalam jumlah besar karena akan dijual kembali. Contoh penjualan grosir.

2. *Business to Consumer* (B2C)

   Jenis transaksi dimana pembelinya perorangan dan tidak punya tujuan untuk menjualnya kembali biasanya semacam toko *online* yang menjual berbagai macam barang.

3. *Consumer to Consumer* (C2C)

   Jenis transaksi dimana pembelinya perorangan yang tidak mempunyai tujuan untuk dijual kembali dan penjualnya juga perorangan yang tidak menyediakan bermacam-macam barang melainkan hanya beberapa barang saja. Contoh: *online* advertising.

4. *Consumer to Business* (C2B)

   Termasuk kedalam kategori ini adalah perseorangan yang menjual produk atau layanan kepada organisasi, dan perseorangan yang mencari penjual, berinteraksi dengan mereka dan menyepakati suatu transaksi.

II.6.6 Komponen Utama e-Commerce

II.6.6.1 Electronic Data Interchange (EDI)

Electronic Data Interchange (EDI) didefinisikan sebagai pertukaran data komputer antar berbagai bidang organisasi atas suatu informasi terstruktur dalam format yang standar dan bisa diolah oleh komputer. EDI merupakan bentuk *e-commerce* sesuai definisinya, dan telah ada bentuk yang sama selama lebih dari 20 tahun. Saat ini teknologi dan implementasi EDI sudah sangat berkembang.

Tujuan EDI adalah untuk memfasilitasi perdagangan dengan cara mengikat bisnis antar partner dagang, EDI meningkatkan proses manual untuk mempertukarkan informasi dengan bidang bisnis lainnya dalam berbagai cara, misalnya data hanya perlu untuk dimasukkan satu kali saja, kemudian data tersebut bisa digunakan oleh pihak pengirim barang, manager kantor, dan lain-lainnya. Hal ini akan menurangi tenaga entry data. Pada dasarnya, data bisa dikirimkan dengan lebih efisien dengan menggunakan EDI.

Komponen utama dari EDI standar adalah sebagai berikut :

1. *Data Element*

   Merupakan potongan data seperti tanggal, harga atau nama organisasi,. Setiap data element diidentifikasikan dengan nomor referensi tertentu yang berisi judul, keterangan, jenis, nomor, dan panjang minimum/maximum.

2. *Data Segment*

   Dalam suatu baris data disebut dengan segment dan setiap item di dalam segmen mewakili satu elemen. Misalnya segmen baris pesanan pembelian

terdiri atas nomor barang, keterangan, jumlah, unit pengukuran, dan harga barang. Setiap segmen memiliki satu identifier, satu data elemet delimiter, element diagrams, data segmen terminator dan notes.

3. *Transaction Set*

   Suatu *transation set* merupakan dokumen khusus seperti dokumen pesanan pembelian. Di dalam transaction set, ada 3 area utama yaitu *area header*, *area detail* dan *area summary*.

4. *Functional Group*

   Merupakan sekelompok *transaction set* yang sejenis. *Transation set* di dalam *functional group* dikelompokkan berdasarkan *functional identifier* yang sama. Untuk mengirimkan transaksi EDI pada konsumen, diperlukan 4 fungsi dasar yaitu *Mapping* elemen dalam suatu database, *Extraction* atas data yang belum diidentifikasi dari database, *Transalation* atas data yang sudah diekstrak ke format EDI, dan Transmisi pesan dalam format EDI melalui media komunikasi.

   Berikut ini adalah penjelasan singkat dari masing-masing fungsi tersebut:

   a. *Mapping*

      Merupakan proses identifikasi elemen di dalam database yang diperlukan untuk membuat pesan dalam format EDI. *Mapping* adalah pekerjaan yang hanya satu kali dilakukan pada saat diperlukan transaksi EDI baru. Software EDI tidak bisa melaksanakan pekerjaan ini.

b. *Extraction*

Merupakan proses pengumpulan data yang belum diidentifikasi dan menempatkannya ke dalam format tertentu. Secara umum, data extract dari database dan dijadikan dalam bentuk flat file. Struktur dari flat file biasanya ditentukan oleh pembuat translation software.

c. *Translation*

Untuk mengirimkan pesan keluar, ketika data yang diperlukan masih dalam bentuk flat file, pembentukan pesan EDI bisa dilakukan menggunakan software translasi atau formatting. Software translasi akan mengatur data menjadi struktur tertentu yang sesuai dengan kebutuhan transaksi EDI.

d. *Communication*

Pengiriman/transmisi atas pesan EDI dikendalikan oleh software komunikasi, yang akan mengatur dan memelihara: nomor telepon partner dagang, menjalankan automatic dialing dan *up/downloading*, juga membuat *activity log*. Setiap pesan EDI dibungkus dengan amplop khusus yang bertuliskan alamat tujuan, serta jenis transaksi EDI sebagai *header* dan *error checking codes* sebagai tambahan di bawahnya. Untuk keperluan penerimaan pesan EDI, proses tersebut tinggal dibalik.

II.6.6.2 Digital Currency

Digital currency dimaksudkan untuk memungkinkan user untuk memindahkan dananya secara elektronik dalam lingkungan kerja tertentu. Saat ini, *digital currency* dirancang untuk versi elektronik dari uang kertas, dimana memiliki atribut yang sama dengan media fisik sebenarnya baik secara anatomis maupun dari segi likuiditasnya.

Karakteristik *digital currency* adalah sebagai berikut:

1. Mewakili suatu nilai moneter tertentu
2. Bisa ditukarkan sebagai alat pembayaran untuk barang dan jasa, mata uang dan koin serta token lainnya
3. Bias disimpan dan diambil lagi.
4. Sulit diduplikasi atau dipalsukan.

Jenis-jenis digital currency antara lain:

1. *Electronic Cash*

   Sistem *electronic cash* telah terintegrasi sepenuhnya dengan *software web browser* untuk memudahkan pembelian barang melalui internet. Sistem *electronic cash* bisa menunjukkan saldo terakhir pada *user* tertentu sesuai permintaan. *Electronic cash* pada umumnya memerlukan infrastruktur *public key* dan mekanisme enkripsi tertentu. Saat ini *electronic cash* belum sepopuler pengunaan *smart card* atau model pembayaran lainnya.

2. *Micropayments*

   Micropayments adalah pembayaran untuk item dengan nilai relatif rendah, misalnya informasi atau hiburan *on-line* yang biayanya bervariasi antara 1

cent sampai 10 *cent*. Sedangkan *Minipayment* adalah pembayaran untuk item dengan nilai antara $ 0,25 sampai $ 10. Ada beberapa skema yang mampu menangani micropayments yaitu: *Milicent*, *eCash*, *CyberCoin*, *Mondex*, *VisaCash* dan *NetBill*.

II.6.6.3 Electronic Catalog

Electronic Catalogs (e-catalogs) telah berada pada aplikasi komersil yang dirancang untuk internet dan merupakan komponen utama dari sistem *e-commerce. E-catalogs* merupakan antar muka grafis (*Graphical User Interface*) yang umumnya berbentuk halaman WWW dimana menyediakan informasi tentang penwaran produk dan jasa. *E-catalog* umumnya mendukung on line shopping dan kemampuan pemesanan dan pembayaran barang. Aplikasi *e-catalog* sebaiknya memiliki karakteristik seperti bersifat interaktif, mampu diperbaharui secara dinamis, *hypertextuality* dan *global presence*.

II.6.6.4 Intranet Dan Extranet

Umumnya *intranet* digambarkan hanya sebagai *web server* di dalam perusahaan (internal), padahal sebenarnya intranet hanyalah kumpulan web site yang dimiliki oleh suatu kelompok (biasanya perusahaan) yang bisa diakses hanya oleh anggota kelompok tersebut. Sedangkan *extranet* merupakan area tertentu dari *intranet* yang bisa diakses oleh kelompok di luar anggota kelompok *intranet*, tapi dengan otorisasi tertentu. Fitur *intranet* standar dalam suatu organisasi memiliki 4 kemamapuan dasar yaitu *e-mail, on line publishing, on line searches* dan

application distribution. Sedangkan *extranet* memperluas fitur ini ke partner bisnis jika dimungkinkan.

Keuntungan menggunalkan intranet di dalam suatu organisasi :

1. Mempercepat proses bisnis.
2. Memfasilitasi pertukaran informas.
3. Meningkatkan komunikasi dan kolaborasi.

II.6.7 Security e-Commerce

Secara umum, keamanan merupakan salah satu komponen atau servis yang dibutuhkan untuk menjalankan *e-commerce*. Beberapa topik yang harus dikuasai antara lain akan didaftar di bawah ini.

1. Teknologi *Kriptografi*

   Teknologi kriptografi menjelaskan bagaimana mengamankan data dengan menggunakan enkripsi. Berbagai sistem sudah dikembangkan seperti sistem *private key* dan *public key*. Penguasaan algoritma-algoritma populer digunakan untuk mengamankan data juga sangat penting. Contoh algoritma ini antara lain DES, IDEA, RC5, RSA dan ECC (*Ellliptic Curve Cryptography*). Salah satu masalah dalam mengamankan enkripsi adalah bagaimana memastikan bahwa hanya sang penerima yang dapat mengakses data.

2. Konsultan Keamanan

   Konsultan, organisasi, dan institusi yang bergerak di bidang keamanan dapat membantu meningkatkan dan menjaga keamanan. Contoh organisasi yang bergerak di bidang ini adalah IDCERT.

II.7 Paypal

II.7.1 Definisi Paypal

PayPal adalah salah satu alat pembayaran (*Payment procesors*) menggunakan internet yang terbanyak digunakan didunia dan teraman. Pengguna internet dapat membeli barang di *ebay*, lisensi *software original*, keanggotaan situs, urusan bisnis, mengirim dan menerima donasi/sumbangan, mengirim uang ke pengguna *PayPal* lain di seluruh dunia dan banyak fungsi lainnya dengan mudah dan otomatis menggunakan internet atau mobile, *PayPal* mengatasi kekurangan dalam pengiriman uang tradisional seperti Cek atau *Money order* yang prosesnya dapat memakan waktu *PayPal* seperti rekening bank, pertama anda membuat *account*, lalu dana akan ditarik langsung dari kartu kredit anda ke *account* tersebut atau dengan dana dari transferan account *PayPal* orang lain ke Saldo / *Balance PayPal* anda, dan anda sudah dapat menggunakan *account PayPal* untuk bertransaksi.

Februari 2010 *PayPal* sudah menerima 190 negara dan 23 mata uang, pengguna *PayPal* di Indonesia masih harus menggunakan hitungan *US dollar* karena rupiah belum ada di *PayPal*, dana yang ditarik langsung dari kartu kredit akan dikonversi ke *US dollar* otomatis.

eBay adalah tempat lelang/jual-beli/pasar *online* terbesar didunia, dengan pengguna lebih dari 200 juta orang di seluruh dunia (data juni 2006), anda dapat mencari dan membeli apapun, mulai dari barang elektronik, kendaraan, hingga barang-barang langka dari seluruh pengguna ebay didunia, cara pembayaran utama di *eBay* adalah dengan *PayPal* karena lebih aman dan otomatis, meskipun ada penjual yang menerima pembayaran dengan cara lain.

II.7.2 Keuntungan PayPal

Pembeli / *Buyer* yang merasa dirugikan atau tidak sesuai deskripsi yang diberitahukan bisa meminta uang kembali kepada Penjual / *Seller* dalam waktu maksimal 45 hari setelah terjadi pembayaran. Jika pembeli menggunakan kartu kredit akan mendapatkan pengembalian uang *chargeback* dari perusahaan kartu kreditnya.

PayPal juga melindungi penjual dari pengembalian uang atau komplain yang tidak benar dari pembeli sesuai pembuktian. Perlindungan untuk penjual dirancang untuk melindungi penjual dari klaim pembeli yang mengaku telah mengirim uang yang tidak mempunyai catatan bukti pembayaran dan deskripsi transaksi, setiap pembelian dan pembayaran menggunakan *PayPal* selalu ada catatan bukti pembayarannya di account *PayPal* pengirim dan penerima uang, sebagai bukti jika benar telah terjadi pengiriman uang antara keduanya

Dari sini bisa diambil kesimpulan menggunakan *PayPal* lebih aman dari alat pembayaran *online* lain bagi pengirim dan penerima uang karena ada bukti

pembayaran dan catatan deskripsi, serta dapat melakukan komplain yang benar jika terjadi sesuatu atau terhindar dari komplain yang salah.

II.7.3 Sejarah PayPal

Paypal (*Paypal corp*.) seperti yang dikenal sekarang adalah penggabungan antara perusahaan *Confinity* dan X.com pada tahun 2000 *Confinity* didirikan pada desember 1998 oleh *Peter Thiel* dan *Max Levchin*, awalnya sebagai alat pembayaran Palm Pilot dan sebagai perusahaan kriptograpi (ilmu pembacaan sandi, tulisan-tulisan atau angka-angka rahasia). Sedangkan X.com didirikan oleh Elon Musk pada maret 1999, yaitu perusahaan penyedia jasa perencanaan keuangan. Kedua perusahaan lokasi pertamanya adalah kantor di 165 *University Avenue* di *Palo Alto*, *California*, rumah dari beberapa pemula di *Silicon Valley*.

Ebay melihat perkembangan penggunaan *Paypal* dan menyimpulkan *Paypal* cocok sebagai salah satu alat pembayaran di *ebay*. Pada mei 1999 *ebay* membeli Billpoint dan menjadikan *Billpoint* sebagai alat pembayaran utama *ebay* saat itu dan membuat Billpoint khusus hanya untuk alat pembayaran di ebay saja, karenanya *Paypal* hanya tampil beberapa kali sebagai pilihan alat pembayaran di ebay dibandingkan Billpoint. Tetapi karena masyarakat lebih menyukai *Paypal* karena mudah dan lebih aman pada Februari 2000 rata-rata ada 200.000 penawaran barang perhari yang menggunakan *Paypal* sedangkan *Billpoint* hanya 4.000 penawaran. Pada april 2000 lebih dari 1 juta penawaran menggunakan *Paypal*.

II.8 SSL (Secure Socket Layer)

II.8.1 Definisi SSL

Secure Socket Layer (SSL) adalah protokol yang digunakan untuk browsing *web* secara aman. SSL bertindak sebagai protokol yang mengamankan komunikasi antara *client* dan *server*. Protokol ini memfasilitasi penggunaan enkripsi untuk data yang rahasia dan membantu menjamin integritas informasi yang dipertukarkan antara website dan *web browser*.

II.9 SEO

SEO (*Search Engine Optimization*) adalah serangkaian proses yang dilakukan secara sistematis yang bertujuan untuk meningkatkan volume dan kualitas trafik dari mesin pencari menuju alamat situs web tertentu dengan memanfaatkan mekanisme kerja alami algoritma mesin pencari tersebut.

Praktisi SEO terkenal seperti *Rand Fishkin, Barry Schwartz, Aaron Wall and Jill Whalen* mencoba mempelajari beberapa pendekatan berbeda untuk *Search Engine Optimization*, dan telah menerbitkan opini mereka di forum *online* dan blog. Praktisi SEO mencoba mempelajari pola yang diterapkan oleh beragam *Search Engine* untuk melihat lebih dalam algoritma tersebut.

Berdasarkan *Jerri L. Ledford* (*Search Engine Oprimization*), SEO adalah teknik pencarian yang menggunakan kata kunci atau frase yang mengandung indikator yang terkandung dalam halaman halaman web, informasi tersebut yang akan di index oleh mesin pencari.

Berdasarkan *David Viney* (*Get to the top on Google*), SEO adalah sebuah teknik pengoptimasian sebuah halam yang mengandung kata kunci atau frase yang bersangkutan yang akan di index oleh mesin pencari sebagai kata kunci pecarian.

Berdasarkan *Peter Kent* (*Search Engine Oprimization for Dummies*), SEO adalah teknik yang di gunakan untuk mendapatkan posisi yang menguntungkan di mesin pencari yang sesui dengan kata kunci yang berada di halaman *website*.

Tujuan dari SEO adalah agar web/blog selalu berada di halaman terdepan dan teratas dari suatu *Search Engine*, sehingga selalu berada di posisi tearatas, maka besar kemungkinan web/blog sering dikunjungi.

II.9.1 Sejarah SEO

Pada pertengahan tahun 1990-an ketika para webmaster dan penyedia konten mulai mengoptimalkan situs mereka di *Search Engine* yang mulai menggunakan katalog. Cara optimasi situs (SEO) saat itu adalah dengan mengirimkan alamat sebuah situs atau halaman situs dalam bentuk URL ke berbagai *search*.

Pada periode tersebut algoritma *search engine* belum terlalu kompleks sehingga mudah dimanipulasi. Versi awal algoritma pencarian didasarkan sepenuhnya pada informasi yang diperoleh pada saat "*crawling*" oleh "*spider*" pada bagian "*meta tag*" pada kode html situs web. "*Meta tag*" menyediakan informasi tentang konten yang terkandung pada suatu halaman web dengan serangkaian ***keyword***. Bagian inilah yang kemudian dimanipulasi dengan cara

memasukkan *keyword* yang tidak relevan, tidak lengkap, dan tidak konsisten dengan konten situs, sehingga *Search Engine* salah menempatkan dan memeringkat situs tersebut. Hal ini menyebabkan hasil pencarian menjadi tidak akurat dan menimbulkan kerugian baik bagi *Search Engine* maupun bagi pengguna internet yang mengharapkan informasi yang relevan dan berkualitas.

Search Engine yang yang dikembangkan oleh *Larry Page* dan *Sergey Brin*, dua orang mahasiswa doktoral ilmu komputer *Universitas Stanford*, cukup berhasil mengatasi permasalahan tersebut dengan mengembangkan *Backrub*, sebuah *Search Engine* yang sepenuhnya mengandalkan algoritma matematis untuk memeringkat halaman web.

Bisnis dan layanan SEO berkembang pesat seiring dengan pertumbuhan web, yang menyebabkan suatu situs web harus berjuang sekuat tenaga agar urlnya mudah ditemukan calon pelanggan di antara jutaan alamat situs lain dari seluruh dunia yang menjadi kompetitornya. *Search Engine* merupakan pintu masuk utama, karena pengguna internet tidak lagi sanggup (dan tidak ingin) menghafalkan alamat-alamat situs web, dan sebagai gantinya mereka mengandalkan indeks yang dibuat oleh *Search Engine*.

II.9.2 Fungsi SEO

Fungsi dari SEO antara lain :

1. Memberikan posisi terbaik di mesin pencari.
2. Memberikan trafic sesui dengan target pengunjung.
3. Mempopulerkan situs web site dengan cepat.

II.9.3 Kelebihan SEO

Dengan adanya mesin pencari saat ini, maka user akan di permudah dalam mencari dan mengetahui situs – situs yang akurat sesuai dengan informasi yang di butuhkan oleh user.

SEO dapat memberikan pengunjung informasi yang lebih akurat yang sesuai dengan isi *content* situs serta mempermudah user untuk mendapatkan informasi yang akurat dan sesuai dengan informasi yang di butuhkan.

II.9.4 Kekurangan SEO

Banyaknya situs web di seluruh dunia membuat mesin pencari harus semakin selektif dalam proses *indexing* sebuah halaman situs, maka mesin pencari membuat standarisasi dari konten situs web.

Maka memerlukan waktu yang cukup lama untuk mendapatkan pengunjung pada situs web yang di rekomendasikan oleh mesin pencari sesuai sesuai dengan isi konten situs web.

II.9.5 Strategi Penerapan SEO

Strategi penerapan SEO merupakan teknik yang paling efektif, untuk memaksimalkan kinerja situs web *hosting* sehingga situs *website* yang menggunakan teknik ini akan lebih maksimal, berikut stategi yang di terapkan.

1. Usia Domain

   Penerapan teknik SEO ini akan berjalan dengan baik jika usia domain sudah memiliki usia minimal 6 bulan dari tanggal registrasi kepemilikan nama domain.

2. *Unique Page Title*

   Page Title adalah pemberian judul halaman web yang akan menjadi informasi pertama yang akan direkam oleh mesin, sehingga bisa di mengeti oleh pengunjung situs *website*.

3. *Meta Description*

   Meta Description adalah *tag* untuk memberikan penjelasan mengenai halaman tersebut, ini akan menjadi info penting yang akan di simpan oleh mesin pencari yang akan di berikan pada user yang mencari sesuai dengan kategori yang di maksud.

4. *URL*

   URL adalah alamat sebuah halaman website yang akan memudahkan mensin pencari untuk membaca halaman yang sesui dengan kata kunci yang dimaksud.

5. *Site Navigation*

   Site Navigation adalah untuk memberikan petunjuk kepada pengunjung bahwa alur *link* di dalam situs tersebut.

6. *Anchot Text*

   Anchot Text berfungsi untuk memberikan tanda bahwa kategori tersebut berhubungan dengan halaman yang lainnya.

7. *Tags Appropriatly*

   Tags Appropriatly berfungsi untuk menampilkan struktur halaman kepada pengguna.

8. *Images*

   Images berfungsi untuk memudahkan pengunjung jika suatu file gambar tidak bisa di akses maka akan ada keterangan mengenai gambar tersebut.

9. *Robots*

   Robots berfungsi untuk memberitahukan pada robot google bahwa halaman atau direktori mana saja yang tidak boleh di akses.

II.10 Konsep Dasar Analisis Sistem

II.10.1 Flowmap

Bagan alir atau *flow map* adalah bagan yang menunjukan alir didalam program atau prosedur sistem secara logika. Bagan alir digunakan terutama untuk alat bantu komunikasi dan dokumentasi. Bagan alir dokumen atau sering disebut juga *Flow Map* merupakan bagan alir yang menunjukan arus dari aliran formulir termasuk tembusan-tembusannya.

II.10.2 ERD (Entity Relationship Diagram)

Entity Relationship Diagram atau biasa dikenal dengan diagram E-R secara grafis menggambarkan isi sebuah database. Diagram ini memiliki dua komponen utama yaitu entity dan relasi. Untuk melambangkan fungsi diatas maka digunakan simbol-simbol yang bisa dilihat pada daftar simbol.

Elemen-elemen *Entity Relationship Diagram* adalah sebagai berikut:

1. *Entity* (Entitas)

   Pada E-R diagram, *entity* digambarkan dengan sebuah bentuk persegi panjang. Entity adalah sesuatu apa saja yang ada didalam sistem, nyata maupun abstrak dimana data tersimpan. Entitas diberi nama dengan kata benda dan dapat dikelompokan dalam empat jenis nama, yaitu : orang, benda, lokasi kejadian (terdapat unsur waktu didalamnya). Simbol entitas pada ERD dapat dilihat pada gambar II.3.



Gambar II.3 Simbol Entitas Pada ER Diagram

2. *Relationship* (Relasi)

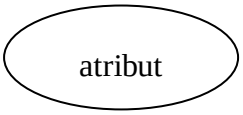
   Pada E-R diagram, *relationship* dapat digambarkan dengan sebuah bentuk belah ketupat. Realationship adalah hubungan alamiah yang terjadi antara entitas. Pada umumnya relationship diberi nama dengan kata kerja dasar, sehinga memudahkan untuk melakukan pembacaan relasinya. Simbol relasi pada ERD dapat dilihat pada gambar II.4.

Gambar II.4 Simbol Relasi Pada ER Diagram

3. Atribut

   Seacara umum atribut adalah sifat atau karakteristik dari tiap entitas maupun tiap *relationship*. Maksudnya adalah sesutau yang menjelaskan

apa sebenarnya yang dimaksud entitas maupun *relationship*, sehingga sering dikatakan bahwa atribut adalah elemen dari setiap entitas dan *relationship*. Simbol attribut pada ERD dapat dilihat pada gambar II.5.



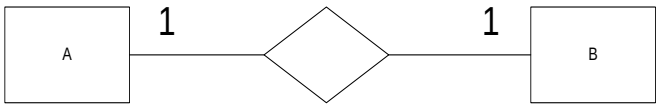
Gambar II.5 Simbol Atribut Pada ER Diagram

4. Kardinalitas

Kardinalitas relasi menunjukan jumlah maksimum tupel yang dapat berelasi dengan entitas yang lainnya. Dari sejumlah kemungkinan banyaknya hubungan yang terjadi dari entitas, kardinalitas relasi merujuk kepada hubungan maksimum yang terjadi dari entitas yang satu ke entitas yang lainnya dan begitu juga sebaliknya. Macam-macam kardinalitas relasi, yaitu :

a. *One to one Relationship*

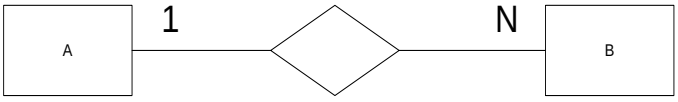
Tingkat hubungan satu ke satu, dinyatakan dengan satu kejadian pada entitas pertama, hanya mempunyai satu hubungan dengan satu kejadian pada entitas yang kedua dan sebaliknya. Simbol kardinalitas relasi *one to one* dapat dilihat pada gambar II.6.



Gambar II.6 Kardinalitas Relasi *One to One*

b. *One to many Relationship*

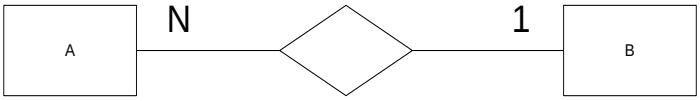
Tingkat hubungan satu ke banyak adalah untuk satu kejadian pada entitas yang pertama dapat mempunyai banyak hubungan dengan kejadian pada entitas yang kedua. Simbol kardinalitas relasi *one to many* dapat dilihat pada gambar II.7.



Gambar II.7 Kardinalitas Relasi *One to* Many

c. *Many to one Relationship*

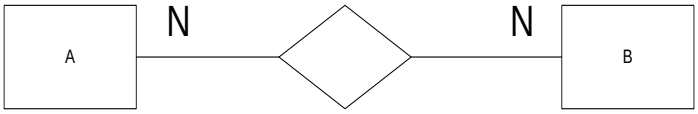
Untuk banyak kejadian pada entitas yang pertama hanya dapat mempunyai satu hubungan dengan kejadian pada entitas yang kedua. Simbol kardinalitas relasi *many to one* dapat dilihat pada gambar II.8.



Gambar II.8 Kardinalitas Relasi *Many to One*

d. *Many to many Relationship*

Tingkat hubungan banyak ke banyak terjadi jika tiap kejadian pada sebuah entitas akan mempunyai banyak hubungan dengan kejadian pada entitas lainnya, baik dilihat dari sisi entitas yang pertama maupun dilihat dari sisi yang kedua. Simbol kardinalitas relasi *many to many* dapat dilihat pada gambar II.9.



Gambar II.9 Kardinalitas Relasi *Many to Many*

5. *Key* (Kunci)

   Sebuah atribut atau set atribut yang nilainya mengidentifikasikan entitas secara unik dalam suatu entitas. *Key* memiliki beberapa jenis sesuai dengan kegunaannya masing-masing, yaitu *primary key* (kunci utama), *foreign key* (kunci tamu).

II.11 Analisis dan Perancangan Berorientasi Objek

II.11.1 Konsep Dasar Pendekatan Objek

Konsep dasar pendekatan objek adalah suatu teknik atau cara pendekatan baru dalam melihat permasalahan dari sistem yang akan dikembangkan sebagai suatu kumpulan objek yang berkorespondensi dengan objek-objek dunia nyata. Ada banyak cara untuk mengabstraksikan dan memodelkan objek-objek tersebut, mulai dari abstraksi objek, kelas, hubungan antar kelas sampai abstraksi sistem.

II.11.2 Objek dan Kelas

Objek adalah abstraksi dari sesuatu yang mewakili dunia nyata seperti benda, manusia, satuan, organisasi, tempat, kejadian, struktur, status, dan hal-hal lainnya yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan) yang dapat diterapkan atau dapat berperngaruh pada status objeknya. Dalam konteks OOP (*Object Oriented Programming*), objek adalah instansi yang dibentuk secara

seketika dari kelas pada saat dieksekusi atau dapat dikatakan objek adalah instansi dari kelas. Objek memiliki siklus hidup mulai dari diciptakan, dimanipulasi, dan dihancurkan.

Kelas adalah kumpulan dari objek-objek yang memiliki karakteristik yang sama. Sebuah kelas akan mempunyai sifat (attribut), kelakuan (operasi), hubungan (*relationship*), dan arti. Suatu kelas dapat diturunkan dari kelas lain, dimana atribut dari kelas semula dapat diwariskan ke kelas yang baru.

II.11.3 Sistem Berorientasi Objek

Sistem berorientasi objek adalah sebuah sistem yang komponennya dibungkus (dienkapsulasi) menjadi kelompok data dan fungsi. Ada beberapa karakteristik atau sifat-sifat yang dimiliki oleh sebuah sistem berorientasi objek yaitu:

1. Abstraksi

   Prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana, dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.

2. Enkapsulasi

   Pembungkusan atribut data dan layanan (operasi-operasi) yang dimiliki oleh objek, untuk menyembunyikan implementasi dari objek sehingga objek lain tidak mengetahui cara kerjanya. Konsep enkapsulasi ini mendukung *information hiding*. Untuk mengakses nilai atribut kelas harus melalui sebuah operasi. Detail implementasi dari data dan prosedur

disembunyikan dari dunia luar, sehingga mengurangi efek samping ketika terjadi perubahan dalam kelas.

3. Pewarisan (*Inheritance*)

   Inheritance merupakan pewarisan sifat dari sebuah *class* ke *class* yang baru. *Subclass* Y merupakan pewaris dari *superclass* X, maka *subclass* Y mewarisi semua atribut dan operasi yang dimiliki oleh *superclass* X. Hal ini mendukung konsep *reuse*. Pada setiap level hirarki *class*, atribut dan operasi baru dapat ditambahkan ke *class* yang telah diwarisi dari level yang lebih tinggi dalam hirarki. Pada *inheritance* juga memungkinkan terjadinya *overriding*. *Overriding* terjadi ketika atribut dan operasi yang diwarisi, dimodifikasi untuk kebutuhan spesifik dari *class* yang baru

4. *Reusability*

   Reusability adalah pemanfaatan kembali objek yang sudah didefinisikan untuk suatu permasalahan pada permasalahan lainnya yang melibatkan objek tersebut.

5. Generalisasi dan Spesialisasi

   Generalisasi dan spesialisasi menunjukkan hubungan antar kelas dan objek yang umum dengan kelas dan objek yang khusus.

6. Komunikasi antar Objek

   Komunikisi antar objek dilakukan lewat pesan (*message*) yang dikirim dari satu objek ke objek lainnya.

7. *Polymorphism*

   Polymorphism adalah kemampuan suatu objek untuk digunakan di banyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program. Hal ini membuat objek saling terpisah dari objek lainnya dan membuat setiap objek lebih independen.

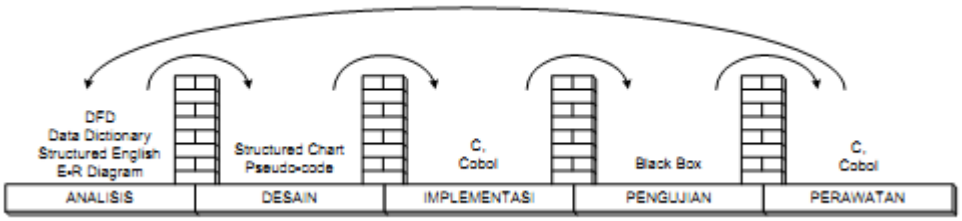
II.11.4 Metodologi Berorientasi Objek

Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya dan dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek meliputi aktivitas analisis berorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek.

Strategi utama untuk menangani kompleksitas pembangunan perangkat lunak adalah dekomposisi permasalahan menjadi bagian-bagian kecil yang dapat dikelola. Pada metode berorientasi fungsi atau aliran data (DFD), dekomposisi permasalahan berdasarkan fungsi atau proses secara hirarki, mulai dari konteks sampai proses-proses yang paling kecil. Sementara pada metode berorientasi objek, dekomposisi permasalahan dilakukan berdasarkan objek-objek yang ada dalam sistem. Ada beberapa faktor mengapa metode berorientasi objek banyak dipilih diantaranya adalah sebagai berikut:

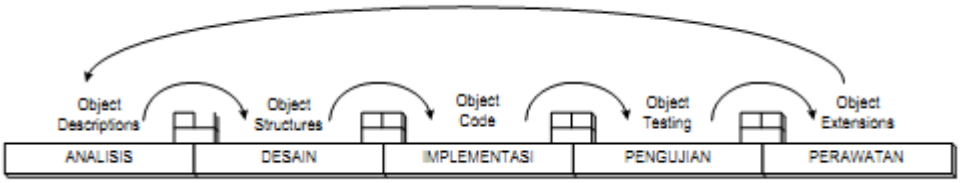
1. Metode lama banyak menimbulkan masalah.

   Adanya kesulitan pada saat mentransformasi hasil dari suatu tahapa pengembangan ke tahap berikutnya, misalnya pada metode *Structured Analysis and Design* yang dapat dilihat pada gambar II.10.



Gambar II.10 Tahapan dalam Metode *Structure Analysis and Design*

Bandingkan tahapan pengembangan dengan metode berorientasi objek terhadap metode terstruktur yang dapat dilihat pada gambar II.11.



Gambar II.11 Tahapan dalam Metode Berorientasi Objek

2. Jenis aplikasi yang dikembangkan pada saat ini berbeda dengan masa lalu.

   Aplikasi yang dikembangkan pada saat ini sangat beragam (aplikasi bisnis, realtime, *utility* dan sebagainya) dengan platform yang berbeda-beda, sehingga menimbulkan tuntutan kebutuhan metodologi pengembangan yang dapat mengakomodasi kesemua jenis aplikasi tersebut.

II.11.4.1 Keuntungan Metodologi Berorientasi Objek

Pengembangan sistem dengan metode berorientasi objek dapat memberikan keuntungan-keuntungan sebagai berikut:

1. Meningkatkan produktivitas.

   Karena kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai ulang untuk masalah lainnya yang melibatkan objek tersebut (*reusable*).

2. Kecepatan pengembangan.

   Karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan menyebabkan berkurangnya kesalahan pada saat pengkodean.

3. Kemudahan pemeliharaan.

   Karena dengan objek, pola-pola yang cenderung tetap dan stabil dapat dipisahkan dari pola-pola yang mungkin sering berubah.

4. Adanya konsistensi.

   Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analisis, perancangan, maupun pengkodean.

5. Meningkatkan kualitas perangkat lunak.

   Karena pendekatan pengembangan lebih dekat dengan dunia nyata dan adanya konsistensi pada saat pengembangannya, perangkat lunak yang

dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan.

II.11.4.2 Beberapa Metode Berorientasi Objek

Ada beberapa metode pengembangan perangkat lunak berorientasi objek yang sudah dikenal dan diantaranya adalah:

1. *Object Oriented Analysis* (OOA) dan *Object Oriented Design* (OOD) dari Peter Coad dan Edward Yourdan.
2. *Object Modelling Technique* (OMT) dari James Rumbaugh, Michael Blaha, Wiliam Premerlan, Frederick Eddy, dan William Lorensen.
3. *Object Oriented Software Engineering* (OOSE) dari Ivar Jacobson.
4. *Booch Method* dari Grady Booch.
5. *Syntropy* dari Steve Cook dan John Daniels.
6. *Unified Modelling Language* (UML) dari James Rumbaugh, Grady Booch, dan Ivar Jacobson.

II.12 Unified Modelling Language (UML*)*

II.12.1 Sejarah Unified Modelling Language (UML)

Tahun 1994, Grady Boch dan James Rumbaugh bergabung untuk menggunakan metode berorientasi objek. Ivan Jacobson bergabung pada tahun 1995, dan mereka bertiga fokus membuat suatu bahasa pemodelan objek standar sebagai ganti dari pendekatan atau metode objek standar. Berdasarkan kerja mereka dan hasil kerja lainnya pada industri, *Unified Modeling Language* **(**UML**)** versi 1.0 dirilis pada tahun 1997.

Unified Modeling Language **(**UML**)** tidak menentukan metode untuk sistem-sistem pengembangan, tetapi sudah diterima luas sebagai standar untuk pemodelan objek. *Object Management Group* (OMG), badan standar industri, mengadopsi UML pada bulan November 1997 dan terus bekerja sama untuk meningkatkannya berdasarkan kebutuhan industri. Pada saat ini, salah satu industri telah merilis sebuah *sofware* yang mendukung UML yaitu *Visual Paradigm* 6.4 *Interprise edition*. Berbagai industri juga bermunculan dan mendukung penggunaan UML dengan berbagai produk, diantaranya *Rational Rose*, *SmartDraw*, *StarUML*, dan lain-lain.

II.12.2 Definisi Unified Modelling Language (UML)

Berikut ini definisi *Unified Modeling Language* (UML) menurut para ahli:

1. Menurut (Henderi, 2006) "*Unified Modeling Language* (UML) adalah bahasa yang telah menjadi standard untuk visualisasi, menetapkan, membangun dan mendokumentasikan artifak suatu sistem perangkat lunak".
2. Menurut (Adi Nugroho : 2005). "*Unified Modeling Language* (UML) adalah alat bantu analisis serta perancangan perangkat lunak berbasis objek".
3. Menurut (Joomla dari http://soetrasoft.com : 2007). *"Unified Modeling Language* (UML) merupakan standard *modeling language* yang terdiri dari kumpulan-kumpulan diagram, dikembangkan untuk membantu para pengembang sistem dan *software* agar bisa menyelesaikan tugas-tugas

seperti: Spesifikasi, Visualisasi, Desain Arsitektur, Konstruksi, Simulasi dan testing serta Dokumentasi".

Berdasarkan beberapa pendapat yang dikemukakan diatas dapat ditarik kesimpulan bahwa "*Unified Modeling Language* (UML) adalah sebuah bahasa yang berdasarkan grafik atau gambar untuk menvisualisasikan, menspesifikasikan, membangun dan pendokumentasian dari sebuah sistem pengembangan perangkat lunak berbasis OO (*Object Oriented*)".

II.12.3 Konsep Pemodelan Menggunakan UML

Pemodelan menggunakan Unified Modeling Language merupakan metode pemodelan berorientasi objek dan berbasis visual. Karenanya pemodelan menggunakan UML merupakan pemodelan objek yang fokus pada pendefinisian struktur statis dan model sistem informasi yang dinamis daripada mendefinisikan data dan model proses yang tujuannya adalah pengembangan tradisional. UML menawarkandiagram yang dikelompokan menjadi lima perspektif berbeda untuk memodelkan suatu sistem. Seperti satu set *blue print* yang digunakan untuk membangun sebuah rumah.

UML merupakan salah satu alat bantu yang sangat handal dalam bidang pengembangan sistem berorientasi objek karena UML menyediakan bahasa pemodelan visual yang memungkinkan pengembang sistem membuat *blue print* atas visinya dalam bentuk yang baku. UML berfungsi sebagai jembatan dalam mengkomunikasikan beberapa aspek dalam sistem melalui sejumlah elemen grafis yang bisa dikombinasikan menjadi diagram. UML mempunyai banyak diagram

yang dapat mengakomodasi berbagai sudut pandang dari suatu perangkat lunak yang akan dibangun. Diagram-diagram tersebut digunakan untuk :

1. Mengkomunikasikan ide.
2. Melahirkan ide-ide baru dan peluang-peluang baru.
3. Menguji ide dan membuat prediksiMemahami struktur dan relasi-relasinya.

II.12.4 Tujuan Penggunaan UML

Adapun tujuan dari penggunaan *Unifed Modelling Language* (UML) adalah :

1. Memberikan bahasa pemodelan yang bebas dari berbagai bahas pemrograman dan proses rekayasa.
2. Menyatukan praktek-praktek terbaik yang terdapat dalam pemodelan.
3. Memberikan model yang siap pakai, bahsa pemodelan visual yang ekspresif untuk mengembangkan dan saling menukar model dengan mudah dan dimengerti secara umum.
4. UML bisa juga berfungsi sebagai sebuah (*blue print*) cetak biru karena sangat lengkap dan detail. Dengan cetak biru ini maka akan bias diketahui informasi secara detail tentang coding program atau bahkan membaca program dan menginterpretasikan kembali ke dalam bentuk diagram (*reserve enginering*).

II.12.5 Fokus Unified Modeling Language (UML)

Menurut (Adi Nugroho : 2005) dalam kerangka spesifikasi, *Unified Modeling Language* (UML) menyediakan model-model yang tepat, tidak mendua arti *(ambigu)* serta lengkap. Secara khusus, *Unified Modeling Language (uml* menspesifikasikan langkah-langkah penting dalam pengambilan keputusan analisis, perancangan serta implementasi dalam sistem yang sangat bernuansa perangkat lunak *(software intensive system).*

Dalam hal ini, *Unified Modeling Language* (UML) bukanlah merupakan bahasa pemprograman tetapi model-model yang tercipta berhubungan langsung dengan berbagai macam bahasa pemprograman, sehingga adalah mungkin melakukan pemetaan *(mapping)* langsung dari model-model yang dibuat dengan *Unified Modeling Language* (UML) dengan bahasa-bahasa pemprograman berorientasi obyek, seperti *Java, Borland Delphi, Visual Basic, C++,* dan lain-lain. Pemetaan *(mapping) Unified Modeling Language* (UML) bersifat dua arah yaitu :

1. Generasi kode bahasa pemprograman tertentu dari *Unified Modeling Language* (UML) *forward engineering.*
2. Generasi kode belum sesuai dengan kebutuhan dan harapan pengguna, pengembang dapat melakukan langkah balik bersifat *iterative* dari implementasi ke *Unified Modeling Language* (UML) hingga didapat sistem/peranti lunak yang sesuai dengan harapan pengguna dan pengembang.

II.12.6 Bangunan Dasar Metodologi UML

Menurut (Adi Nugroho : 2005) bangunan dasar metodologi *Unified Modeling Language* (UML) menggunakan tiga bangunan dasar untuk mendeskripsikan sistem/perangkat lunak yang akan dikembangkan yaitu :

1. Sesuatu *(things)*

   Ada 4 (empat) *things* dalam *Unified Modeling Language* (UML), yaitu:

   a. *Structural things*

      Merupakan bagian yang relatif statis dalam model *Unified Modeling Language* (UML). Bagian yang relatif statis dapat berupa elemen-*elemen* yang bersifat fisik maupun konseptual. Jenis-jenis structural things adalah *class*, *interface*, *collaboration*, *use case*, *active class*, *component*, *artifact*, dan *node* (Booch, et al., 2005).

   b. *Behavioral things*

      Merupakan bagian yang dinamis pada model *Unified Modeling Language* (UML), biasanya merupakan kata kerja dari model *Unified Modeling Language* (UML), yang mencerminkan perilaku sepanjang ruang dan waktu. Jenis *behavorial things* adalah *interaction*, *state machine*, dan *activity* (*Booch, et al*., 2005)

   c. *Grouping things*

      Merupakan bagian pengorganisasi dalam *Unified Modeling Language* (UML). Dalam penggambaran model yang rumit kadang *diperlukan* penggambaran paket yang menyederhanakan model. Paket-paket ini

kemudian dapat didekomposisi lebih lanjut. Paket berguna bagi pengelompokkan sesuatu, misalnya model-model dan subsistem-subsistem. Satu-satunya jenis *grouping things* adalah *package*. *Package* merupakan mekanisme untuk mengorganisasikan *design* itu sendiri. *Structural things*, *behavorial things*, dan bahkan *grouping things* yang lain dapat ditempatkan dalam sebuah *package* (*Booch, et al*., 2005).

d. *Annotational things*

Annotational things adalah bagian penjelas dari model UML. Jenis *annotational things* hanya satu, yaitu *note* yang digunakan untuk memberikan *constraint* atau komentar yang dilekatkan pada sebuah atau sekumpulan elemen.

2. Relasi *(Relationship)*

Relationship merupakan *building block* UML yang berfungsi sebagai penghubung antar-things. Jenis relationship yang terdapat pada UML adalah *dependency*, *association*, *generalization*, dan *realization* (*Booch, et al*., 2005).

3. Diagram

Diagram adalah presentasi grafis dari sekumpulan elemen. Diagram merupakan kombinasi antara things dan relationship. Diagram dibuat untuk memvisualisasikan sistem dari sejumlah perspektif yang berbeda, sehingga diagram merupakan proyeksi terhadap sistem. Pada UML versi 2

terdapat 13 diagram, yaitu: *class*, *object*, *component*, *composite stucture*, *use case*, *sequence*, *communication*, *state*, *activity*, *deployment*, *package*, *timing*, dan *interaction diagram* (*Booch, et al*., 2005). Penjelasan sejumlah diagram yang umum digunakan adalah sebagai berikut:

a. *Use Case Diagram*

   Diagram ini memperihatkan himpunan use case dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini terutama sangat penting untuk mengorganisasi dan memodelkan perilaku dari suatu sistem yang dibutuhkan serta diharapkan pengguna.

b. *Class Diagram*

   Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi dan relasi-relasi antar objek.

c. *Sequence Diagram*

   Diagram ini memperlihatkan interaksi yang menekankan pada pengiriman pesan *(message)* dalam suatu waktu tertentu.

d. *Collaboration Diagram*

   Diagram ini merupakan diagram interaksi yang menekankan organisasi struktur dari objek-objek yang mengirim dan menerima pesan.

e. *State Chart Diagram*

   Diagram ini memperlihatkan state-state pada sistem, memuat state, transisi, event, dan aktifitas. Diagram ini terutama penting untuk

memperlihatkan sifat dinamis dari antarmuka, kelas, kolaborasi dan terutama penting pada pemodelan sistem-sistem yang reaktif.

f. *Activity Diagram*

Diagram ini memperlihatkan aliaran dari suatu aktifitas ke aktifitas lainnya dalam suatu sistem. Diagram ini terutama penting dalam pemodelan fungsi-fungsi dalam suatu sistem dan memberi tekanan pada aliran kendali antar objek.

II.12.7 Diagram Dasar dalam UML

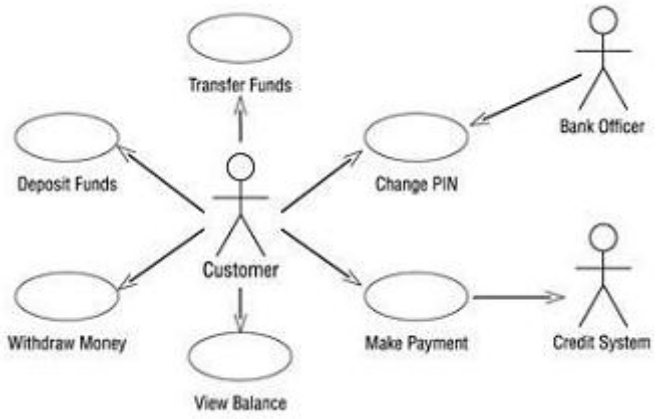
Berikut ini adalah penjelasan mengenai berbagai diagram UML serta tujuannya:

1. Model *Use Case* Diagram

   Use case diagram menunjukkan sekumpulan use case, aktor, dan relationship yang terjadi antara use case dan aktor tersebut. Use case menggambarkan bagaimana aktor berinteraksi dengan sistem. Aktor adalah entitas eksternal yang memainkan peranan (role) ketika berinteraksi dengan sistem. Aktor dapat berupa user atau benda seperti dumb terminal, sensor, atau sistem komputer lainnya. *Use Case* secara naratif digunakan untuk secara tekstual menggambarkan sekuensi langkah-langkah dari setiap interaksi. Jenis-jenis relationship pada use case dapat dilihat pada tabel II.1, dan contoh sebuah model *use case* diagram dapat dilihat pada gambar II.12.

Tabel II.1 Jenis Relationship pada Use Case Diagram

Jenis	Deskripsi	Gambar
Association	*Association* merupakan komunikasi antar *actor* dan *use case* yang terlibat	
Generalization	*Generalization* merupakan hubungan generalisasi/spesialisasi. *Generalization* dapat terjadi antar *use case* atau *actor*	
Extend	Relasi yang menspesifikasikan bahwa target *use case* meng-*extend behavior* dari *source use case*	<<extends>>
Include	X *include* Y berarti *use case* X menggunakan *use case* Y sepenuhnya.	<<include>>

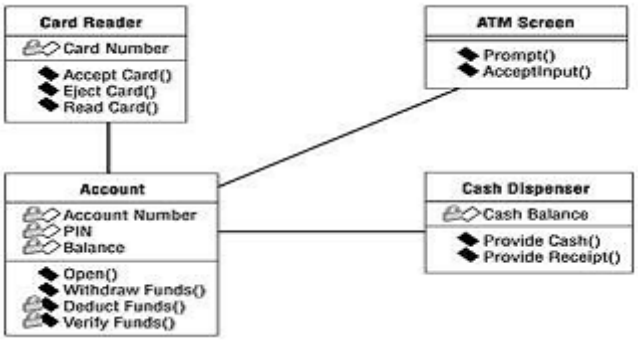


Gambar II.12 Contoh *Use Case Diagram* Sistem ATM

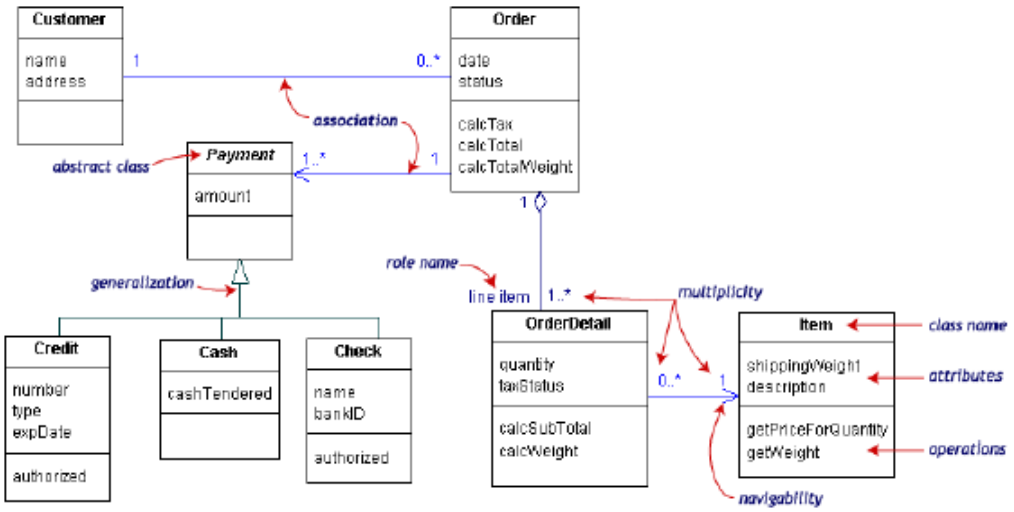
2. Diagram Struktur Statis

UML menawarkan dua diagram untuk memodelkan struktur statis sistem informasi, yaitu:

a. *Class Diagram* : menggambarkan struktur object sistem. Diagram ini menunjukan *class object* yang menyusun sistem dan juga hubungan antara *class object* tersebut. Sebagai contoh dapat dilihat pada gambar II.13 dan gambar II.14.



Gambar II.13 Contoh *Class Diagram* Sistem ATM



Gambar II.14 Contoh Sederhana *Class Diagram* dan Hubungan Antar Kelas

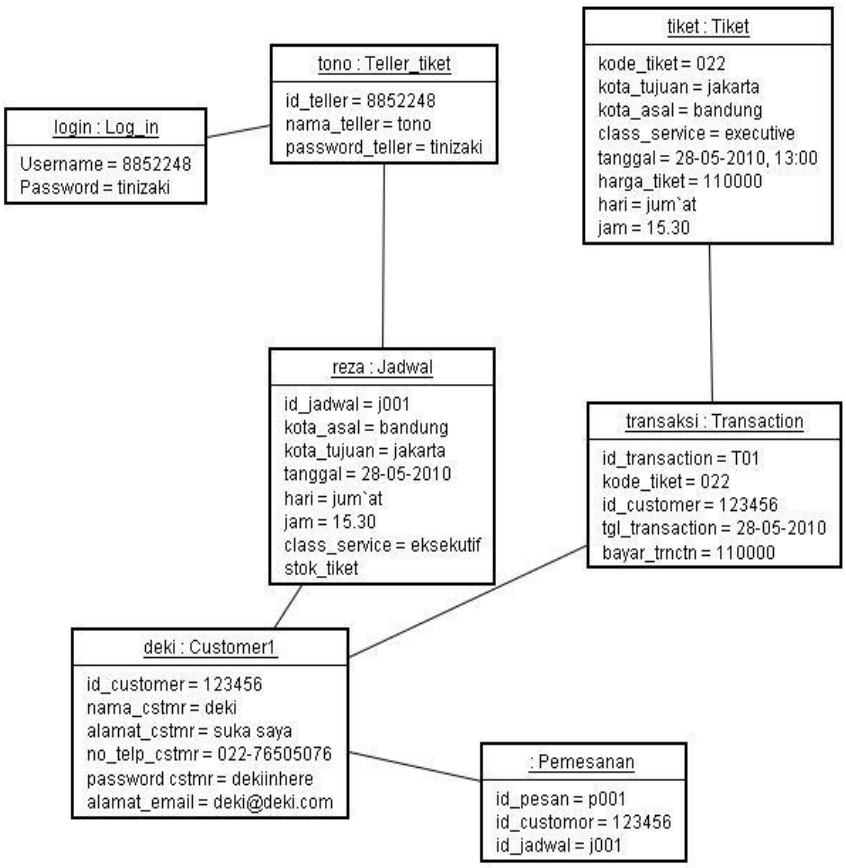
Dua element utama dari *class diagram* adalah *class,* dan *relationship.* Notasi *class* terdiri atas 3 bagian, yaitu nama, atribut dan operasi. Nama suatu *class* digunakan sebagai pengenal. Atribut merupakan informasi yang dimiliki oleh suatu *class,* sedangkan operasi merupakan tingkah laku yang didefinisikan suatu *class. Class* dapat berkolaborasi dengan *class* lainnya dalam berbagai cara. Jenis *relationship* antar *class* dapat dilihat pada tabel II.2.

Tabel II.2 Jenis Relationship pada *Class Diagram*

Jenis	Deskripsi	Gambar
Association	*Association* menghubungkan dua *class* dan menunjukkan hubungan semantik. *Assoctiation* dapat mempunyai arah yang menunjukkan *navigability*.	
Depedency	*Dependency* adalah relasi yang menunjukkan bahwa sebuah *class* menggunakan informasi atau layanan dari *class* lain.	
Generalization	*Generalization* menunjukkan hubungan generalisasi/spesialisasi.	
Aggregation	*Aggregation* menunjukkan hubungan “bagian dari” atau *whole*/*part hierarchy*.	
Composition	*Composition* menunjukkan jenis khusus dari *aggregation* dengan bagian bawah *multiplicity* selalu bernilai 1.	

Dalam class diagram dikenal adanya *multiplicity*, yaitu angka yang menunjukkan jumlah kemungkinan instantiasi dari sebuah *class* berhubungan dengan instantiasi dari *class* yang lain. *Multiplicity* dapat dilihat dari angka yang terdapat pada hubungan antar-class, misalnya 1, 0..1, 0..*, dan 1..*. Pada umumnya *multiplicity* didefinisikan dengan batas bawah dan batas atas. Batas bawah dapat berupa angka positif atau nol, sedang batas atas dapat berupa angka positif atau * (untuk nilai tak terhingga). Jika batas bawah dan batas atas bernilai sama maka hanya perlu menuliskan sekali.

b. *Object Diagram* : serupa dengan class diagram, tetapi *object diagram* memodelkan *instance object actual* dengan menunjukan nilai-nilai saat ini dari atribut *instance*. *Object Diagram* menyajikan "*snapshot*/potret" tentang objek sistem pada point waktu tertentu. Diagram ini tidak digunakan sesering *Class Diagram*, tetapi saat digunakan dapat membantu seorang *developer* memahami struktur sistem secara lebih baik. Contoh model *object diagram* dapat dilihat pada gambar II.15.



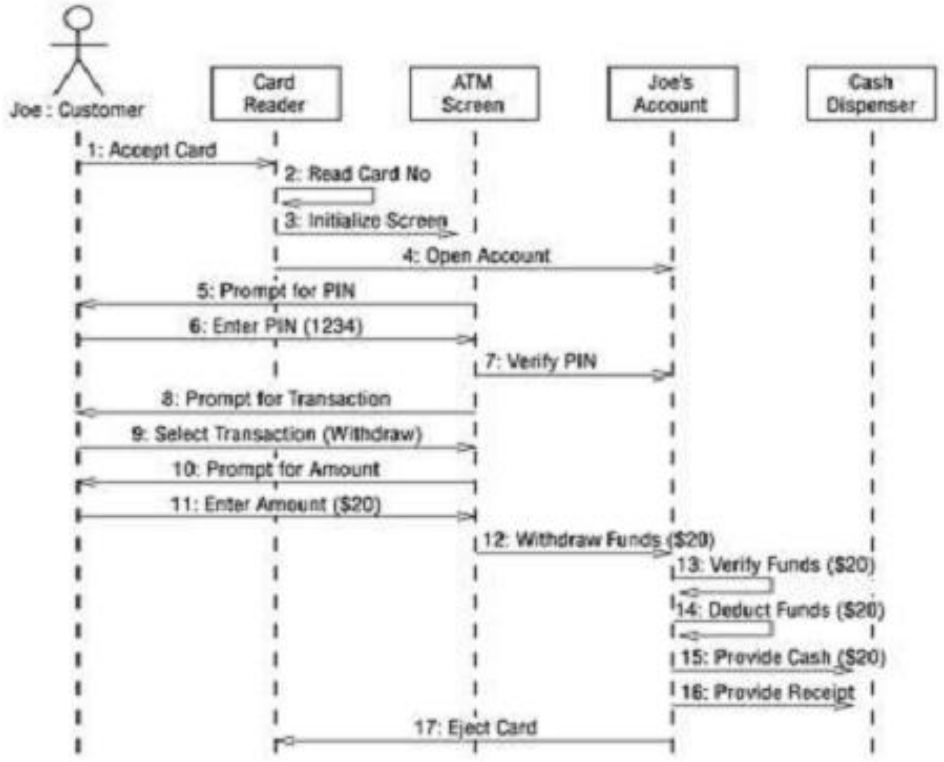
Gambar II.15 Contoh *Object Diagram* Pada Sistem Pemesanan Tiket

3. Diagram Interaksi

Diagram interaksi memodelkan sebuah interaksi, terdiri dari satu set objek, hubungan-hubungannya, dan pesan yang terkirim di antara objek. Model diagram ini memodelkan *behavior* (kelakuan) sistem yang dinamis dan UML memiliki dua diagram untuk tujuan ini, yaitu:

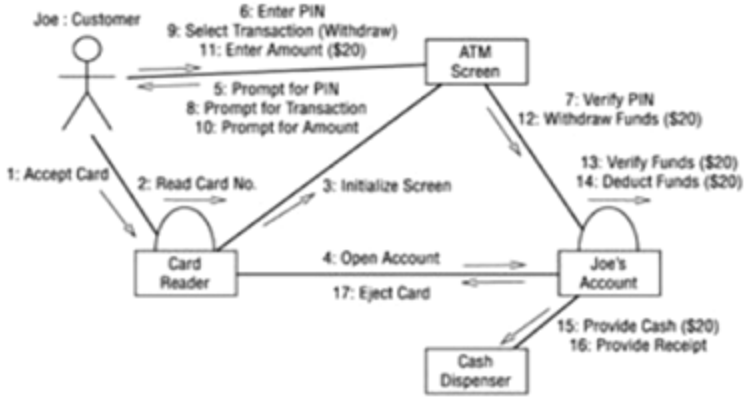
a. Diagram Rangkaian/*Sequence Diagram*: secara grafis menggambarkan bagaimana objek berinteraksi dengan satu sama lain melalui pesan pada sekuensi sebuah *use case* atau operasi. Diagram ini mengilustrasikan bagaimana pesan terkirim dan diterima di antara

objek dan dalam sekuensi atau *timing* apa. Contoh model *sequence diagram* dapat dilihat pada gambar II.16.



Gambar II.16 Contoh *Sequence Diagram* Pada Transaksi Sistem ATM

b. Diagram Kolaborasi/*Collaboration Diagram* : serupa dengan diagram rangkaian/sekuensi, tetapi tidak fokus pada *timing* atau sekuensi pesan. Diagram ini justru menggambarkan interaksi (atau kolaborasi) antara objek dalam sebuah format jaringan. Contoh model *collaboration diagram* dapat dilihat pada gambar II.17.



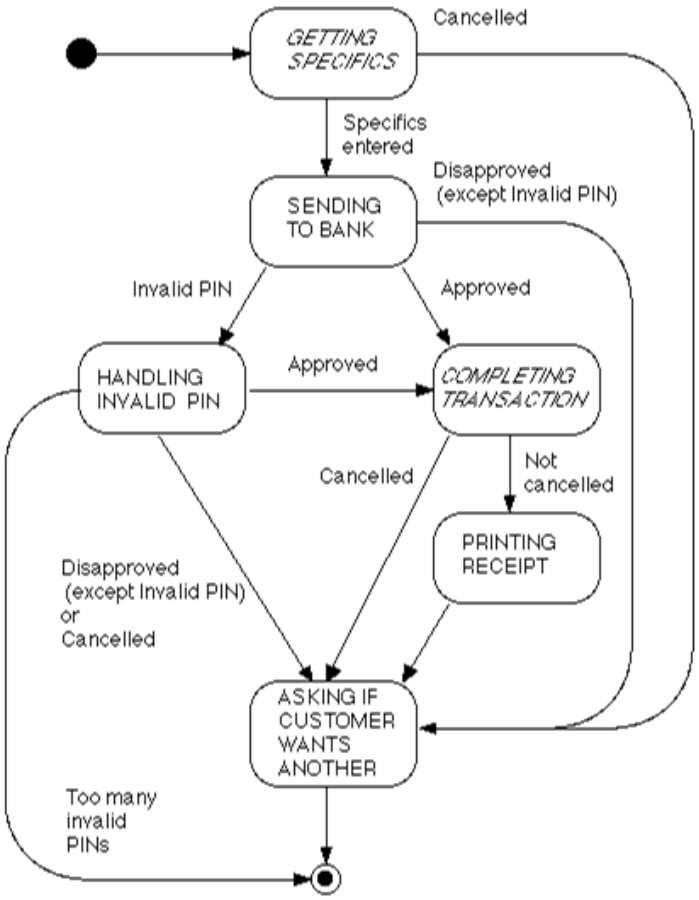
Gambar II.17 Contoh *Collaboration Diagram* Pada Transaksi Sistem ATM

Diagram rangkaian maupun diagram kolaborasi merupakan *isomorphic* artinya kita dapat mengubah dari satu diagram ke diagram lain.

4. Diagram State/*State Diagram*

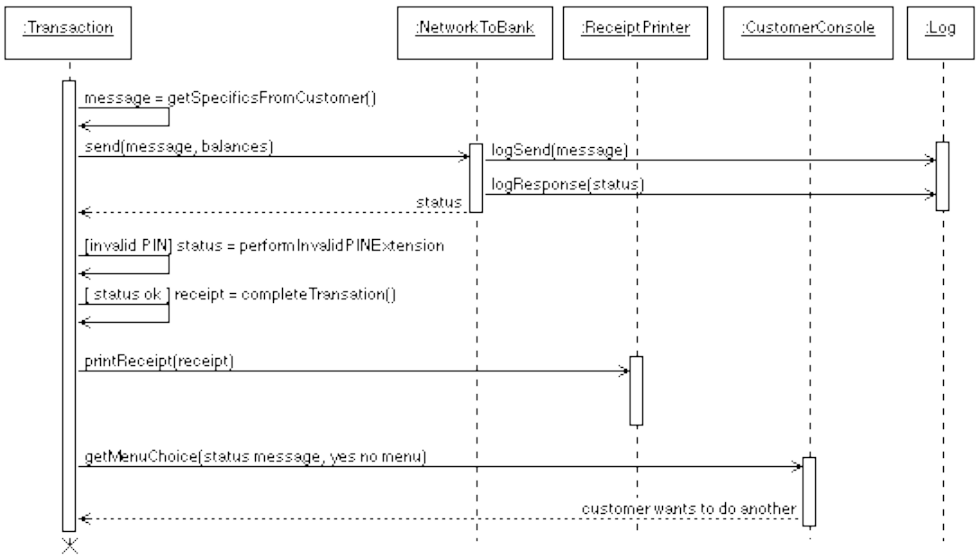
   UML memiliki sebuah diagram untuk memodelkan *behavior* objek khusus yang kompleks (*statechart*) dan sebuah diagram untuk memodelkan *behavior* dari sebuah *use case* atau sebuah metode, yaitu:

   a. *Diagram Statechart* : digunakan untuk memodelkan *behavior* objek khusus yang dinamis. Diagram ini mengilustrasikan siklus hidup objek-berbagai keadaan yang dapat diasumsikan oleh objek dan *event-event* (kejadian) yang menyebabkan objek beralih dari satu *state* ke *state* lain. Contoh model *statechart diagram* dapat dilihat pada gambar II.18.



Gambar II.18 Contoh *Statechart Diagram* Pada Transaksi Sistem ATM

b. Diagram aktivitas/*Activity Diagram* : secara grafis digunakan untuk menggambarkan rangkaian aliran aktivitas baik proses bisnis maupun *use case*. *Activity diagram* dapat juga digunakan untuk memodelkan *action* yang akan dilakukan saat sebuah operasi dieksekusi, dan memodelkan hasil dari *action* tersebut. Contoh model activity diagram dapat dilihat pada gambar II.19.



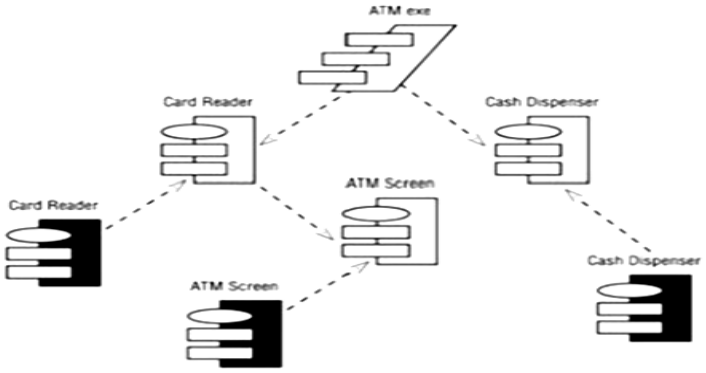
Gambar II.19 Contoh *Activity Diagram* PadaTransaksi Sistem ATM

5. Diagram Implementasi

Diagram implementasi juga memodelkan struktur sistem informasi, yaitu:

1. Diagram komponen/*Component Diagram*

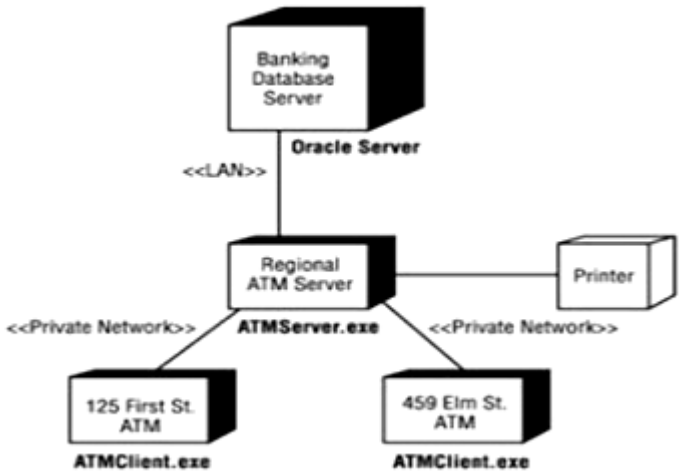
   Digunakan untuk menggambarkan organisasi dan ketergantungan komponen-komponen *software* sistem. Komponen diagram dapat digunakan untuk menunjukan bagaimana kode pemrograman dibagi menjadi modul-modul (atau komponen). Contoh model *component diagram* dapat dilihat pada gambar II.20.



Gambar II.20 Contoh *Component Diagram* Pada Sistem ATM

2. Diagram penguraian/*Deployment Diagram*

   Digunakan untuk mendeskripsikan arsitektur fisik dalam istilah ”*node*” untuk *hardware* dan *software* dalam sistem. Diagram ini menggambarkan konfigurasi komponen-komponen software *real-time*, prosesor, dan peralatan yang membentuk arsitektur sistem. Contoh model *deployment diagram* dapat dilihat pada gambar II.21.



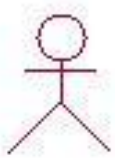
Gambar II.21 Contoh *Deployment Diagram* Pada Sistem ATM

II.12.8 Notasi Umum dalam UML

Dalam *UML* terdapat notasi-notasi yang sering digunakan, antara lain :

A. *Actor*

Actor adalah segala sesuatu yang berinteraksi dengan komputer. *Actor* bisa merupakan sebuah perangkat keras / komputer, orang maupun obyek lain dalam system yang sama. Biasanya yang dilakukan oleh *actor* adalah memberikan informasi pada sistem atau memerintah system untuk melakukan sesuatu. Notasi *actor* pada diagram UML dapat dilihat pada gambar II.22.



Gambar II.22 Notasi *Actor*

B. *Class*

Class merupakan pembentuk utama dari system berorientasi obyek Karen *class* menunjukan kumpulan obyek yang memiliki attribut dan operasi yang sama. *Class* digunakan untuk mengimplementasikan *interface. Class* digunakan untuk mengabstraksikan elemen-elemen dari sistem yang sedang dibangun. *Class* juga bias digunakan untuk mengabstraksikan elemen-elemen dari sistem yang sedang dibangun. *Class* bisa untuk mempresentasikan baik perangkat lunak maupun perangkat keras, baik konsep maupun bendanyata. Notasi *class* berbentuk persegi panjang yang

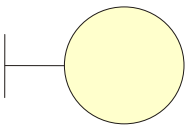
berisi 3 bagian :persegi panjang paling atas untuk nama *class*, persegi panjang paling bawah untuk operasi dan persegi panjang ditengah untuk attribute. Attribut digunakan untuk menyimpan informasi. Nama attribute menggunakan kata benda yang bias dengan jelas merepresentasikan informasi yang disimpan didalamnya. Operasi menunjukan sesuatu yang bias dilakukan oleh obyek, dan menggunakan kata kerja. Notasi *class* pada diagram UML dapat dilihat pada gambar II.23.



Gambar II.23 Notasi *Class*

1. *Boundary Class*

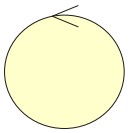
   Boundary class adalah *class* yang menghubungkan user dengan sistem. Oleh karena itu sering disebut juga sebagai user interface class. Biasanya dimulai dengan menetapkan 1 boundary class untuk setiap pasangan actor-use case. User interface class ini sering disamakan dengan form yang digunakan sebagai interface antara system dengan user. Notasi *boundary class* dapat dilihat pada gambar II.24.



Gambar II.24 Notasi *Boundary Class*

2. *Control Class*

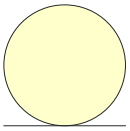
   Control class adalah class yang mengkoordinasi aktivitas dalam sistem. *Class* ini menghubungkan *boundary class* dengan *entity class.* Jadi mengkoordinasi *entity class* mana yang perlu dikunjungi, kapan dan apa yang ingin didapatkan dari entity class tersebut. Notasi *control class* dapat dilihat pada gambar II.25.



Gambar II.25 Notasi *Control Class*

3. Entity Class

   Entity Class adalah class yang behubungan dengan data atau informasi yang digunakan oleh sistem. Entity class ini adalah class yang menyimpan dan mengolah data. Notasi entity class dapat dilihat pada gambar II.26.

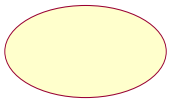


Gambar II.26 Notasi *Entity Class*

C. *Use Case*

Use case menjelaskan kegiatan yang dilakukan *actor* dan system untuk mencapai suatu tujuan tertentu. Walau menjelaskan kegiatan namun *use case* hanya menjelaskan apa yang dilakukan oleh *actor* dan sistem, bukan

bagaimana *actor* dan sistem melakukan kegiatan tersebut. Notasi *use case* pada diagram UML dapat dilihat pada gambar II.27.



Gambar II.27 Notasi *Use Case*

D. *Interaction*

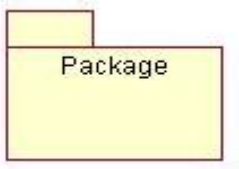
Interaction digunakan untuk menunjukan baik aliran pesan atau informasi antar obyek maupun hubungan antar obyek. Biasanya *interaction* ini dilengkapi juga dengan teks bernama "*operation signature*" yang tersusun dari nama operasi, paramater yang dikirim dan tipe yang dikembalikan. Notasi *interaction* pada diagram UML dapat dilihat pada gambar II.28.



Gambar II.28 Notasi *Interaction*

E. *Package*

Package adalah container atau wadah konseptual yang digunakan untuk mengelompokan elemen-elemen dari sistem yang sedang dibangun, sehingga bias dibuat model yang lebih sederhana. Tujuannya adalah untuk mempermudah pengelihatan dari model yang sedang dibagun. Notasi *package* pada diagram UML dapat dilihat pada gambar II.29.



Gambar II.29 Notasi *Package*

F. *Note*

Note digunakan untuk memberikan keterangan dan komentar dari suatu elemen sehingga bias langsung terlampir dalam model. *Note* ini bias ditempelkan kesemua elemen notasi yang lain. Notasi note pada diagram UML dapat dilihat pada gambar II.30.



Gambar II.30 Notasi *Note*

G. *Dependency*

Dependency merupakan relasi yang menunjukan bahwa perubahan pada salah satu elemen member pengaruh padaelemen yang lain. Elemen yang ada dibagian tanda panah adalah elemen yang tergantung pada elemen yang ada dibagian tanpa tanda panah.

Terdapat 2 *stereotype* dari *dependency*, yaitu *: include dan extend. Include* menunjukan bahwa suatu bagian dari suatu elemen (yang ada digaris tanpa panah) memicu eksekusi bagian dari elemen yang lain (yang ada digaris panah), misalnya untuk notasi A → B operasi yang ada di class A memicu dieksekusinya yang ada di *class* B.

Extend menunjukan bahwa suatu bagian dari elemen di garis tanpa panah bias disisipkan kedalam elemen yang ada digaris dengan panah; missal notasi A → B suatu fungsi dari use case A bias disisipkan kedalam *use case* B atau dengan kata lain A optional untuk B. Notasi *dependency* pada diagram UML dapat dilihat pada gambar II.31.



Gambar II.31 Notasi *Dependency*

H. *Generalization*

Generalization menunjukan hubungan antara elemen yang lebih umum ke elemen yang lebih spesifik. Dengan *generalization*, *class* yang lebih spesifik *(subclass)* akan menurunkan attribute dan operasi dari *class* yang lebih umum *(superclass) atau "subclass is a superclass".* Dengan menggunakan notasi *generalization* ini konsep *inheritance* dari prinsip hirarki dimodelkan. Notasi yang menggambarkan *generalization* dapat dilihat pada gambar II.32.



Gambar II.32 Notasi *Generalization*

I. *Realization*

Realization menunjukan hubungan bahwa elemen yang ada dibagian tanpa panah akan merealisasikan apa yang dinyatakan oleh elemen yang ada dibagian dengan panah. Misalnya class merealisasikan *package*, *component* merealisasikan *class* atau *interface.* Notasi *realization* pada diagram UML dapat dilihat pada gambar II.33.



Gambar II.33 Notasi *Realization*

II.12.9 Langkah-langkah Penggunaan UML

Menurut (Afif Amrullah:2002) langkah-langkah penggunaan *Unified Modeling Language* (UML) sebagai berikut:

1. Buatlah daftar *business process* dari *level* tertinggi untuk mendefinisikan aktivitas dan proses yang mungkin muncul.
2. Petakan *use case* untuk setiap *business process* untuk mendefinisikan dengan tepat fungsional yang harus disediakan oleh sistem, kemudian perhalus *use case diagram* dan lengkapi dengan *requirement, constraints* dan catatan-catatan lain.
3. Buatlah *deployment diagram* secara kasar untuk mendefinisikan arsitektur fisik sistem.
4. Definisikan *requirement* lain *non fungsional, security* dan sebagainya yang juga harus disediakan oleh sistem.
5. Berdasarkan *use case diagram,* mulailah membuat *activity diagram.*
6. Definisikan obyek-obyek level atas *package* atau *domain* dan buatlah *sequence* dan/atau *collaboration* utuk tiap alir pekerjaan, jika sebuah *use case* memiliki kemungkinan alir normal dan *error,* buat lagi satu diagram untuk masing-masing alir.
7. Buatlah rancangan *user interface model* yang menyediakan antamuka bagi pengguna untuk menjalankan *skenario use case.*
8. Berdasarkan model-model yang sudah ada, buatlah *class diagram*. Setiap *package* atau *domian* dipecah menjadi *hirarki class* lengkap dengan

atribut dan metodenya. Akan lebih baik jika untuk setiap *class* dibuat *unit test* untuk menguji *fungsionalitas class* dan interaksi dengan *class* lain.

9. Setelah *class diagram* dibuat, kita dapat melihat kemungkinan pengelompokkan *class* menjadi komponen-komponen karena itu buatlah *component diagram* pada tahap ini. Juga, definisikan *test integrasi* untuk setiap komponen meyakinkan ia bereaksi dengan baik.
10. Perhalus *deployment diagram* yang sudah dibuat. Detilkan kemampuan dan *requirement* piranti lunak, sistem operasi, jaringan dan sebagainya. Petakan komponen ke dalam node.
11. Mulailah membangun sistem. Ada dua pendekatan yang tepat digunakan:
12. Pendekatan *use case* dengan meng*assign* setiap *use case* kepada tim pengembang tertentu untuk mengembangkan unit kode yang lengkap dengan *test.*
13. Pendekatan *komponen* yaitu *mengassign* setiap komponen kepada tim pengembang tertentu.
14. Lakukan uji modul dan uji integrasi serta perbaiki model beserta *code*nya. Model harus selalu sesuai dengan *code* yang aktual.
15. Perangkat lunak siap dirilis.

II.13 Analisis dan Pemodelan Use Case

Pemodelan persyaratan sistem menggunakan *use case* bertujuan untuk men-*capture* dan mendokumentasikan persyaratan sistem dari sudut pandang pengguna dengan cara yang mudah dipahami. Konsep analisis *use case* adalah sebuah proses pengembangan sistem yang didasarkan pada pemahaman akan

kebutuhan *stakeholder* dan alasan mengapa sistem itu harus dikembangkan. Ada beberapa sarana kebutuhan dalam proses pemodelan *use case* yaitu:

a. Proses pemodelan fungsi-fungsi sistem dalam konteks peristiwa-peristiwa bisnis, siapa yang mengawalinya, dan bagaimana sistem itu merespon.
b. Pendekatan yang memfasillitasi pengembangan berpusatkan pada kegunaan atau fungsionalitas.
c. Menentukan, mendokumentasikan, dan memahami persyaratan fungsional sistem informasi.

II.13.1 Manfaat Pemodelan Sistem Mengggunakan Use Case

Ada beberapa manfaat pemodelan persyaratan sistem menggunakan *use case* diantaranya adalah sebagai berikut:

1. Menyediakan *tools* untuk meng-*capture* persyaratan fungsional.
2. Membantu menyusun ulang lingkup sistem menjadi bagian-bagian yang lebih dapat dikelola.
3. Menyediakan alat komunikasi dengan para pengguna dan stakeholder yang berhubungan dengan fungsionalitas sistem.
4. Memberikan cara bagaimana mengidentifikasi, menetapkan, melacak, mengontrol, dan mengelola kegiatan pengembangan sistem, terutama pengembangan *incremental* dan *iteratif*.
5. Menyajikan panduan untuk mengestimasi lingkup usaha dan jadwal proyek.
6. Menyajikan garis pokok pengujian, khususnya rencana test, dan *test case*.

7. Menyajikan garis pokok bagi *help system* dan manual pengguna, dan juga dokumentasi pengembangan sistem.
8. Menyajikan tools untuk melacak persyaratan.
9. Menyajikan titik mulai/awal untuk identifikasi objek data atau entitas.
10. Menyajikan spesifikasi fungsional untuk mendesain antarmuka pengguna dan sistem.
11. Menyajikan alat untuk menentukan persyaratan akses database dalam menambah, mengubah, menghapus, dan membaca.
12. Menyajikan kerangka kerja untuk mengarahkan proyek pengembangan sistem.

II.13.2 Alat Utama yang Digunakan dalam Pemodelan Use Case

Adapun alat utama yang digunakan dalam pemodelan *use case* adalah :

a. *Use Case* Diagram

Diagram yang menggambarkan interaksi antara sistem dengan sistem eksternal dan pengguna. Dengan kata lain secara grafis menggambarkan siapa yang akan menggunakan sistem dengan dengan cara apa pengguna mengharapkan untuk berinteraksi dengan sistem.

b. *Use Case Narative*

Use Case Narative adalah deskripsi tekstual kegiatan bisnis dan bagaimana pengguna akan berinteraksi dengan sistem untuk menyelesaikan suatu tugas.

II.13.3 Komponen Utama dalam Pemodelan Use Case

II.13.3.1 Use Case

Use case adalah urutan langkah-langkah yang secara tindakan saling terkait (skenario), baik terotomatisasi, maupun secara manual, untuk tujuan melengkapi satu tugas bisnis tunggal.

II.13.3.2 Actor

Actor (pelaku) adalah segala sesuatu yang perlu berinteraksi dengan sistem untuk pertukaran informasi. Pada pemodelan *use case* terdapat 4 buah tipe *actor* yaitu :

1. *Primary business actor* adalah *stakeholder* yang terutama mendapatkan keuntungan dari pelaksanaan *use case* (contoh: karyawan dengan menerima gaji untuk periode tertentu).
2. *Primary system actor* adalah *stakeholder* yang secara langsung berhadapan dengan sistem untuk menginisiasi atau memicu kegiatan atau sistem. (contoh: operator telepon yang memberikan bantuan kepada pelanggan, kasir bank yang memproses transaksi).
3. *External server actor* adalah *stakeholder* yang melayani kebutuhan pengguna *use case* (contoh: biro kredit yang memiliki kuasa atas perubahan kartu kredit).
4. *External receiver actor* adalah *stakeholder* yang bukan pelaku utama, tapi menerima nilai yang terukur atau teramati (*output*) dari *use case* (contoh:

gudang menerima paket permintaan untuk menyiapkan pengiriman sesudah seorang pelanggan memesannya).

5. *Temporal event* adalah kejadian sistem yang dipicu dengan waktu (contoh: sistem billing untuk perusahaan kredit secara otomatis mencetak tagihan pada hari kelima dalam bulan itu, billing PLN, billing PAM, bank merekonsiliasi transaksi tiap hari pada jam 5 sore). Yang menjadi *actor/*pelaku disini adalah waktu.

Ada beberapa langkah yang digunakan dalam mengidentifkasikan aktor binis yaitu:

a. Menggunakan referensi dokumen berikut:
   - Diagram konteks yang mengidentifikasi ruang lingkup dan batasan sistem
   - Dokumentasi sistem dan manual pengguna yang ada.
   - Waktu pertemuan proyek dan lokakarya.
   - Dokumen persyaratan, perjanjian proyek, atau pernyataan kerja yang ada.

b. Saat mencari actor, ajukan pertanyaan berikut:
   - Siapa atau apa yang menyediakan input ke dalam sistem ?
   - Siapa atau apa yang menerima output dari sistem ?
   - Interface apa yang dibutuhkan bagi sistem yang lain ?
   - Apakah ada kejadian yang dipicu secara otomatis pada waktu yang telah ditentukan sebelumnya ?
   - Siapa yang akan mengurusi informasi dalam sistem ?

c. Buatlah definisi tekstual aktor menggunakan tabel dengan memuat istilah, sinonim dan deskripsi.

II.13.3.3 Use Case dan Actor Relationship

Relationship adalah hubungan yang terjadi antar simbol dalam *use case. Relationship* digambarkan dengan menggunakan garis dan tipe simbol yang digunakan untuk menghubungkan antara *actor → actor, actor → use case, use case → use case.*

II.13.4 Langkah-langkah Analisis dan Pemodelan Use Case

Terdapat beberapa langkah dalam menyusun use case diagram secara efektif dan benar yaitu :

a. Memilih sistem boundary (batasan, ruang lingkup sistem, paket)
b. Menemukan dan mengidentifikasi *primary actor* dan *user goal*.
c. Mendeskripsikan use case dan spesifikasinya.

II.14 Perangkat Lunak yang Digunakan

II.14.1 PHP

PHP diciptakan oleh *Rasmus Lerdorf* pada tahun 1994 dan bersifat *open source.* Sampai bulan januari 2007, PHP sudah digunakan oleh kurang lebih 20 juta *domain* dan terus berkembang sampai saat ini.

PHP merupakan singkatan dari *Hypertext Preprocessor*, adalah sebuah bahasa scripting yang terpasang pada HTML. Sebagian besar sintaks mirip dengan bahasa C, Java dan Perl, ditambah beberapa fungsi PHP yang spesifik. Tujuan

utama bahasa ini adalah untuk memungkinkan perancang *web* menulis halaman *web* dinamis dengan cepat.

PHP merupakan bahasa pemograman *web* yang bersifat *server-side HTML (embedded scripting)*, di mana script-nya menyatu dengan HTML dan berada si *server*, artinya adalah sintaks dan perintah-perintah yang kita berikan akan sepenuhnya dijalankan di *server* tetapi disertakan HTML biasa. PHP dikenal sebgai bahasa scripting yang menyatu dengan *tag* HTML, dieksekusi di *server* dan digunakan untuk membuat halaman *web* yang dinamis seperti ASP (*Active Server Pages*) dan JSP (*Java Server Pages*).

Seluruh aplikasi berbasis *web* dapat dibuat dengan PHP. Namun kekuatan yang paling utama PHP adalah pada konektivitasnya dengan system *database* di dalam *web*. Sistem *database* yang dapat didukung oleh PHP adalah :

1. Oracle
2. MySQL
3. Sybase
4. PostgreSQL

PHP dapat berjalan di berbagai system operasi seperti *windows* 98/NT, UNIX/LINUX, solaris maupun macintosh. PHP merupakan *software* yang *open source* yang dapat anda download secara gratis dari situs resminya yaitu http://www.php.net, ataupun dari situs-situs yang menyediakan *software* tersebut seperti di ftp://gerbang.che.itb.ac.id.

Software ini juga dapat berjalan pada *web server* seperti PWS (*Personal Web Server*), Apache, IIS, AOL*Server*, fhttpd, phttpd dan sebagainya. PHP juga merupakan bahasa pemograman yang dapat kita kembangkan sendiri seperti menambah fungsi-fungsi baru. Keunggulan lainnya dari PHP adalah PHP juga mendukung komunikasi dengan layanan seperti protocol IMAP, SNMP, NNTP, POP3 bahkan HTTP. PHP dapat diinstal sebagai bagian atau modul dari apache web *server* atau sebagai CGI script yang mandiri.

Banyak keuntungan yang dapat diperoleh jika menggunakan PHP sebagai modul dari apache di antaranya adalah :

1. Tingkat keamanan yang cukup tinggi
2. Waktu eksekusi yang lebih cepat dibandingkan dengan bahasa pemograman web lainnya yang berorientasi pada *server-side scripting*.
3. Akses ke sistem *database* yang lebih fleksibel seperti MySQL.

II.14.2 MySQL

MySQL adalah sebuah aplikasi *Relational Database Management Server* (RDBMS) bersifat *open source* yang memungkinkan data diakses dengan cepat oleh banyak pemakai secara bersamaan dan juga memungkinkan pembatasan akses pemakai berdasarkan *privilege* (hak akses) yang diberikan. *MySQL* menggunakan bahasa SQL (*structured query language)* yang merupakan bahasa standar pemograman *database*.

MySQL dipublikasikan sejak tahun 1996, akan tetapi sebenarnya sudah dikembangkan sejak tahun 1979. *MySQL* telah memenangakan penghargaan *Linux*

Journal Reader's Choice Award selama tiga tahun. MySQL sekarang tersedia di bawah lisensi *open source*, tapi ada juga lisensi utuk menggunakan *MySQL* yang bersifat komersial.

Keunggulan dari *MySQL* adalah :

1. Bersifat open source.
2. Sistem software-nya tidak memberatkan kerja *server* atau komputer karena dapat bekerja di background.

II.14.2.1 Perintah-perintah MySQL

Pada *MySQL* terdapat beberapa perintah. Perintah-perintah pada *MySQL* ini hampir sama dengan perintah-perintah pada *database server* lainnya. Perintah-perintah *MySQL* itu antara lain adalah sebagai berikut :

1. *Create database*, digunakan untuk membuat *database* pada *database server*.

   Sintaksnya adalah :

   ```
   Create database database_name
   ```

   Database_name adalah nama *database* yang akan dibuat.
2. *Use database*, digunakan untuk menunjuk *database* yang akan digunakan.

   Sintaksnya adalah :

   ```
   Use database_name
   ```

   Database_name adalah nama *database* yang akan digunakan.

3. *Create table*, digunakan untuk membuat tabel pada *database* yang digunakan.

   Sintaksnya adalah :

```
Create table table_name
(
Column1 column_type column_attributes,
Column2 column_type column_attributes,
Primary_key (column_name)
);
```

   Table_name adalah nama tabel yang akan dibuat. *Column1* adalah nama kolom yang akan dibuat pada tabel. *Column_type* adalah tipe dari kolom tersebut , dapat berupa *char, varchar, tinytext, mediumtext, longtext, enum, int, tinyint, mediumint, bigint, float, decimal, time, date, datetime, timestamp, year.*

4. *Insert*, digunakan untuk menambahkan record pada tabel.

   Sintaksnya adalah :

```
Insert into table_name(column1, column2,..) values
value1,value2,..)
```

   Table_name adalah nama tabel yang akan ditambahkan *record*-nya. *Column1, column2* adalah kolom yang akan ditambahkan data. *Value1,value2* adalah data yang akan ditambahkan.

5. *Update*, digunakan untuk mengubah record yang sudah ada pada tabel.

   Sintaksnya adalah :

   ```
   Update table_name set column1=value1, column2=value2 where column=value
   ```

   Table_name adalah nama tabel yang akan dirubah *record*-nya. *Column1*, *column2* adalah kolom yang akan dirubah data. *Value1*,*value2* adalah data yang akan digantikan.

6. *Drop table*, digunakan untuk menghapus tabel.

   Sintaksnya adalah :

   ```
   Drop table table_name
   ```

   Table_name adalah nama tabel yang akan dihapus.

7. *Show tables*, digunakan untuk menampilkan tabel-tabel yang telah dibuat dalam *database* yang aktif.

   Sintaksnya adalah :

   ```
   Show tables
   ```

8. *Show field*, digunakan untuk menampilkan seluruh *field* dalam suatu tabel.

   Sintaksnya adalah :

   ```
   Show field from table_name
   ```

   Table_name adalah nama tabel yang akan ditampilkan *field*-nya.

9. *Alter table*, digunakan untuk menambah,merubah, dan menghapus *field* dalam suatu tabel.

   Sintaksnya adalah :

   a. Untuk menambahkan

      ```
      Alter table_name add column column1 column_type column_attributes
      ```

      Table_name adalah nama tabel yang akan ditambahkan *field*-nya. *Column1* adalah nama *field* baru, *column_type* adalah tipe kolom dan *column_attributes* adalah atribut kolom yang akan ditambahkan.

   b. Untuk mengubah

      ```
      Alter table_name change column1 column2 column_type column_attributes
      ```

      column1 adalah mana *field* yang akan dirubah, *column2* adalah nama *field* baru, *column_type* adalah tipe kolom dan *column_attributes* adalah atribut kolom.

   c. Untuk menghapus

      ```
      Alter table table_name drop column column1
      ```

      Table_name adalah tabel yang akan dihapus *field*-nya. *Column1* adalah nama *field* yang akan dihapus.

II.14.2.2 Koneksi PHP dengan MySQL

Untuk menggabungkan bahasa pemograman PHP dan MySQL dibutuhkan beberapa perintah khusus, yaitu :

1. Pembuatan koneksi antara *server* dari MySQL dengan *web server* tempat menyimpan halaman *web*, perintahnya :

```
<?php
Mysql_connet("host name","username","password");
?>
```

2. Setelah terbentuk koneksi selanjutnya dilakukan pemilihan terhadap *database* yang akan digunakan, perintahnya :

```
<?php
Mysql_select_db("nama database");
?>
```

3. Baru kemudian dapat dilakukan perintah-perintah MySQL lainya seperti *select, update, insert,* dan lainnya.

II.14.3 NetBeans 7.0

NetBeans adalah *Integrated Development Environment* (IDE) berbasiskan Java dari Sun Microsystems yang berjalan di atas Swing. Swing sebuah teknologi Java untuk pengembangan aplikasi *desktop* yang dapat berjalan di berbagai macam platforms seperti Windows, Linux, Mac OS X and Solaris.

Suatu IDE adalah lingkup pemrograman yang diintegrasikan kedalam suatu aplikasi perangkat lunak yang menyediakan pembangun *Graphic User Interface* (GUI), suatu *text* atau kode editor, suatu *compiler* atau *interpreter* dan suatu *debugger*.

Netbeans merupakan *software development* yang *Open Source*, dengan kata lain software ini di bawah pengembangan bersama, bebas biaya.

NetBeans mengacu pada kedua platform kerangka untuk aplikasi *desktop* Java, dan sebuah lingkungan pengembangan terpadu (IDE) untuk pengembangan dengan Java, JavaScript, PHP, Python, Ruby, Groovy, C, C++, Scala, Clojure, dan lain-lain.

NetBeans IDE ditulis dalam Java dan berjalan di mana-mana mana JVM diinstal, termasuk Windows, Mac OS, Linux, dan Solaris.Sebuah JDK diperlukan untuk pengembangan fungsionalitas Java, tetapi tidak diperlukan untuk pembangunan di bahasa pemrograman lain.

Platform *NetBeans* memungkinkan aplikasi untuk dikembangkan dari satu set modular komponen *software* yang disebut modul. Aplikasi berbasis *platform NetBeans* (termasuk IDE *NetBeans*) dapat diperpanjang oleh pengembang pihak ketiga .

NetBeans membantu dalam membangun aplikasi *web database* dinamis dengan menggunakan bahasa *server* seperti *Markup Language* (CFML), JSP, dan PHP. Selain itu, *NetBeans* memberikan dukungan terhadap layanan *code folding, safe delete, rename refactoring,* HTML 5.0, *PHPdoc generate.*

II.14.4 Apache

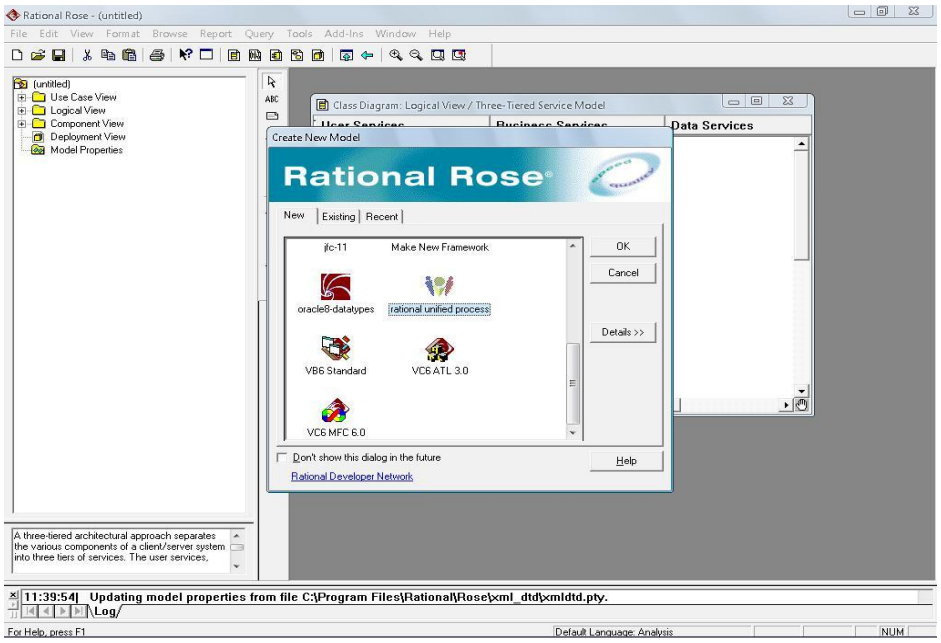
Server HTTP *Apache* atau *Server Web*/WWW *Apache* adalah *server web* yang dapat dijalankan dibanyak sistem operasi (Unix, BSD, Linux, Windows, Novotel Netware dan lainnya) yang berguna untuk melayani dan memfungsikan situs web. Protokol yang digunakan untuk melayani fasilitas web/www ini menggunakan HTTP.

Apache memiliki fitur-fitur yang canggih seperti pesan kesalahan yang dapat dikonfigur, autentikasi berbasis basis data dan lain-lain. *Apache* juga didukung oleh sejumlah antar muka pengguna berbasis grafik (GUI) yang memungkinkan penanganan *server* menjadi mudah.

Apache merupakan *software open source* dikembangkan oleh komunitas terbuka yang terdiri dari pengembang-pengembang dibawah naungan *Apache Software Foundation*

II.14.5 Rational Rose

Software Rational Rose merupakan salah satu tools untuk membuat suatu U*ML*. Banyak *tools selain Rational Rose* dari yang gratis sampai yang berbayar, namun untuk Rational Rose merupakan software yang berbayar. *Rational Rose merupakan software buatan IBM*. Lingkungan kerja *Rational Rose* dapat dilihat pada *Gambar* II.34.

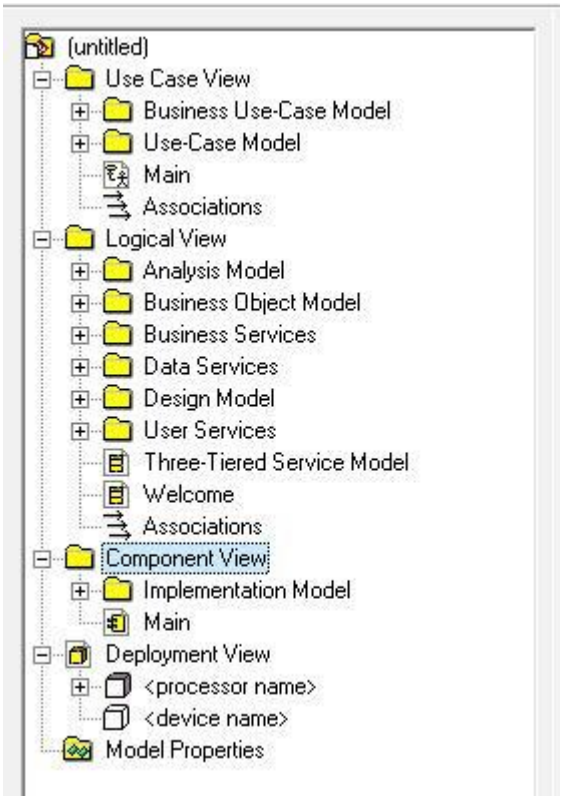


Gambar II.34 Jendela Ruang Kerja Rational Rose

Kerangka kerja yang akan digunakan membuat suatu diagram UML adalah berada dalam Use Case View (untuk membuat business use case diagram dan use case diagram), Logical View (untuk membuat activity diagram, class diagram, sequence diagram dan statate chart serta collaboration diagram), Component View

(untuk membuat component diagram) dan Deployment View (untuk membuat deployment diagram).

Pada gambar II.35 memperlihat area pemodelan di *rational rose* yang digunakan sebagai tempat dimana desain analis akan memodelkan suatu UML :



Gambar II.35 Area Pemodelan UML Pada Rational Rose

II.15 Internet

Internet merupakan jaringan global yang terdiri dari berbagai komputer yang saling berhubungan dan bekerjasama dengan cara berbagai informasi dan data. Media penghubung tersebut bisa melalui kabel, kanal satelit maupun frekuensi radio.

Setiap komputer yang terhubung dengan jaringan tersebut, diberikan sebuah nomor yang unik, dan berkomunikasi satu sama lainnya dengan bahasa

komunikasi yang sama. Bahasa komunikasi yang sama ini disebut protokol. Protokol yang digunakan di internet adalah TCP/IP (*Transmission Control Protocol* / *Internet Protocol*).

II.15.1 Sejarah Internet

Internet pertama kali dikembangkan oleh ARPANet, suatu proyek yang dimulai dari Departemen Pertahanan Amerika Serikat (*US Departement of Defense–DOD*) pada tahun 1969, suatu percobaan dalam *reliable networking* (jaringan secara terpercaya) untuk menghubungkan antara DOD dengan kontraktor penelitian militer, termasuk sejumlah besar universitas yang melakukan penelitian dengan dana militer. Pada konsepnya sebenarnya internet merupakan suatu jaringan, yang mana suatu paket informasi dapat dikirim dari suatu komputer ke komputer yang lain. Hal ini dimaksudkan untuk mempermudah pemindahan berbagai data penting apabila terjadi perang

ARPA merupakan singkatan dari *Advanced Research Projects Agency*, cabang dari *Defence* (Pertahanan) yang mempunyai kewajiban membagi-bagikan uang *grant* sehingga menjadi DARPA (*Defence-ARPA*). ARPANet mulai dengan 3 komputer kecil yang dikoneksi di California digabungkan dengan satu di Utah, tetapi secara cepat berkembang di seluruh kontinen. Internet ini mulai tumbuh pesat pada dekade 1990.

II.15.2 Kegunaan Internet

Kegunaan internet yang utama antara lain :

a. Fungsi komunikasi

Internet adalah alat komunikasi, kegunaan yang sangat penting dari internet adalah pertukaran pesan dengan menggunakan *electronic mail* (*e-mail*).

b. Fungsi *Resource Sharing*

Dengan internet, kita dapat mencari *software*, essay, data dan program dari ribuan titik distribusi di seluruh dunia.

c. Fungsi *Resource Discovery*

Navigasi untuk mencari file tertentu, dokumen, *host* atau orang diantara jutaan host.

d. Fungsi Komunitas

Masyarakat pengguna internet dapat berhubungan dan membuat perkumpulan/komunitas tertentu.

II.15.3 Perkembangan Internet

Jumlah pengguna internet yang besar dan semakin berkembang, telah mewujudkan budaya internet. Internet juga mempunyai pengaruh yang besar atas ilmu, dan pandangan dunia. Dengan hanya menggunakan mesin pencari seperti *Google*, pengguna di seluruh dunia mempunyai akses yang mudah atas bermacam-macam informasi. Dibanding dengan buku dan perpustakaan, internet melambangkan penyebaran (*decentralization*) informasi dan data secara ekstrim.

Perkembangan internet juga telah mempengaruhi perkembangan ekonomi. Berbagai transaksi jual beli yang sebelumnya hanya bisa dilakukan dengan cara tatap muka (dan sebagian sangat kecil melalui pos atau telepon), kini sangat

mudah dan sering dilakukan melalui internet. Transaksi melalui internet ini dikenal dengan nama *e-commerce*.

Terkait dengan teknologi pembuatan website sebagai konten dari teknologi internet itu sendiri memicu berkembangan teknik pembuatan website yang semakin canggih dengan fitur-fitur yang sangat beragam tidak hanya terbatas pada teks dan gambar saja, akan tetapi sudah merambah kedalam dunia multimedia dan telkomunikasi visual. Bahkan dengan munculnya teknologi konten manajemen sistem pengguna yang awam dengan kemampuan teknis yang sedikipun bisa membuat website sesuai dengan keinginannya tanpa harus berhubungan dengan *webmaster* dengan konten yang dinamis dan selalu *uptodate.*

II.15.4 Web Server

Web server menurut Minoli(1998,p33) berpendapat bahwa sebuah *web server* adalah seuatu program untuk menawarkan pelayanan yang bisa diperoleh seluruh jaringan. *Web server* merupakan suatu tipe *server* khusus yang dapat berkomunikasi langsung dengan *client* menggunakan HTTP, *web server* menerima permintaan dari client dan meresponnya, biasanya dengan mengembalikan sebuah dokumen atau gambar.

II.15.5 Web Browser

Dalam dunia web, perangkat lunak *client*, yaitu *browser web* mempunyai tugas yang sama yaitu menterjemahkan informasi yang diterima oleh *server web* dan menampilkannya pada layer computer pengguna, oleh karena HTTP memungkinkan *server web* mengirimkan beragam data, seperti teks atau gambar,

browser harus bisa mengenali berbagai macam data yang akan diterimanya, dan selanjutnya harus tahu cara untuk menampilkanya dengan benar. Teks ditampilkan sebagai teks dan gambar ditampilkan sebagai gambar. Umumnya *browser web* menerima data dalam bentuk HTML. File HTML sebenarnya adalah file teks biasa yang selain berisi informasi yang hendak ditampilkan kepada pengguna, juga mempunyai perintah-perintah untuk mengatur tampilan data tersebut. Browserlah yang memiliki kuasa penuh dalam menterjemahkan perintah-perintah tadi. Meskipun sudah dibuat consensus untuk menstandarkan format dan elemen-elemen HTML, setiap jenis browser bisa menterjemahkan file HTML secara berbeda.

Beberapa *server web* memiliki feature seperti *server side programming*, *sevurity control* dan lain sebagainya. Meskipun beragam macamnya, secara fungsional semua jenis *server web* adalah sama saja, yaitu berfungsi melayani permintaan-permintaan dari *browser web*.

Banyak web browser yang bisa digunakan untuk mengakses web, diantaranya internet explorer, mozilla firefox, opera, safari, dan masih banyak lagi

II.15.6 Website

Website merupakan *webpage* beserta *homepage*, yang merupakan system yang luas dari *server* yang menawarkan informasi untuk semua orang melalui jaringan. Informasi tersebut tersebut berupa teks, gambar, suara, dan tipe data link, yang merupakan navigasi kehalaman lainnya (*hyperlink*). *Homepage*

merupakan halaman yang pertama kali muncul ketika pertama kali dibuka,sedang *webpage* adalah halaman yang ada pada *website*.

II.15.7 World Wide Web (WWW)

Sebuah situs web adalah sebutan bagi sekelompok halaman web (*web page*), yang umumnya merupakan bagian dari suatu nama domain (*domain name*) atau subdomain di *World Wide Web* (WWW) di Internet. WWW terdiri dari seluruh situs web yang tersedia kepada publik. Halaman-halaman sebuah situs web diakses dari sebuah URL yang menjadi "akar" (*root*), yang disebut *homepage* (halaman induk; sering diterjemahkan menjadi "beranda", "halaman muka"), dan biasanya disimpan dalam *server* yang sama. Tidak semua situs web dapat diakses dengan gratis. Beberapa situs web memerlukan pembayaran agar dapat menjadi pelanggan, misalnya situs-situs yang menampilkan pornografi, situs-situs berita, layanan surat elektronik (*e-mail*), dan lain-lain.

Terminologi website adalah kumpulan dari halaman-halaman situs, yang biasanya terangkum dalam sebuah domain atau subdomain, yang tempatnya berada didalam World Wide Web (WWW) di Internet. Sebuah web page adalah dokumen yang ditulis dalam format HTML (Hyper Text Markup Language), yang hampir selalu bisa diakses melalui HTTP, yaitu protokol yang menyampaikan informasi dari *server* website untuk ditampilkan kepada para pemakai melalui web browser. Semua publikasi dari website-website tersebut dapat membentuk sebuah jaringan informasi yang sangat besar.

II.15.8 Universal Resource Locator (URL)

URL adalah rangkaian karakter menurut suatu format standar tertentu, yang digunakan untuk menunjukkan alamat suatu sumber seperti dokumen dan gambar di internet. URL menunjukkan sumber daya internet alamat sebuah halaman web (homepage) yaitu halaman suatu dokumen atau program yang ingin ditampilkan atau digunakan. Nama file ini tidak hanya menunjukan direktori dan nama filenya saja, tetapi juga nama mesinnya dalam jaringan. URL dapat disediakan (ada atau diakses) dengan berbagai metode dan bisa jadi bukan sekedar file. Karena URL dapat menunjukan queri-queri dokumen dapat disimpan dalam database, hasil dari suatu perintah finger atau achie atau apapun yang berkaitan dengan data hasil proses.

II.15.9 Domain Name Service (DNS)

DNS digunakan untuk mempermudah penggunaan internet, dengan memetakan IP *address* ke nama host. Agar data nama host dapat didistribusikan di banyak *server*, format data yang digunakan harus mencerminkan terdistribusinya data tersebut. Untuk itu, digunakan format tree dengan masing-masing nodenya disebut domain. Penulisan nama host secara lengkap disebut sebagai *Full Qualifield Domain Name (FQDN).*

DNS bekerja secara hirarki dan berbentuk seperti pohon (tree). Bagian atas adalah *Top Level Domain* (TLD) seperti COM, ORG, EDU, MIL dsb. Seperti pohon DNS mempunyai cabang-cabang yang dicari dari pangkal sampai ke ujung. Pada waktu kita mencari alamat misalnya linux.or.id pertama-tama DNS bertanya pada TLD *server* tentang DNS *Server* yang melayani domain .id misalnya dijawab

ns1.id, setelah itu dia bertanya pada ns1.id tentang DNS *Server* yang bertanggung jawab atas .or.id misalnya ns.or.id kemudian dia bertanya pada ns.or.id tentang linux.or.id dan dijawab 64.29.24.175

Sedangkan untuk mengubah IP menjadi nama *host* melibatkan domain in-addr.arpa. Seperti domain lainnya domain in-addr.arpa pun bercabang-cabang. Yang penting diingat adalah alamat IP-nya ditulis dalam urutan terbalik di bawah in-addr.arpa. Misalnya untuk alamat IP 64.29.24.275 prosesnya seperti contoh linux.or.id: cari *server* untuk arpa, cari *server* untuk in-addr.arpa, cari *server* untuk 64.in-addr.arpa, cari *server* 29.64.in-addr.arpa, cari *server* untuk 24.29.64.in-addr.arpa. Dan cari informasi untuk 275.24.29.64.in-addr.arpa. Pembalikan urutan angkanya memang bisa membingungkan.

II.16 Email

Email atau elektronik mail adalah media surat menyurat secara *online*, email memungkinkan kita untuk berkirim surat melalui internet dengan waktu yang cepat. Dalam perkembangannya, email berkembang dari sekedar teks ke media yang bisa mengirimkan berbagai media seperti file dokumen dan gambar.

Email bisa di akses dengan jasa mail service seperti Yahoo dan Google, alamat *email* biasanya akan memakai tanda @, sebagai contoh **namakamu@*yahoo*.com** atau **namakamu@*google*.com**.

II.17 Yahoo Messenger

Yahoo Messenger adalah media *chat* / obrolan. Disini kita dapat mengirim pesan/ ngobrol bersama teman yang sudah menerima kita menjadi temannya.