BAB II

LANDASAN TEORI

Dalam penulisan tugas akhir ini penulis menggunakan referensi yang berhubungan dengan pembuatan sistem penggajian (payroll) karyawan dan guru, diantaranya :

2.1. Sistem

Suatu sistem informasi mempunyai landasan teori sistem umum (*general system teory*). Menurut Sutanta (2003), sistem dapat didefinisikan sebagai sekumpulan hal atau kegiatan atau elemen atau subsistem yang saling kerjasama atau yang dihubungkan dengan cara-cara tertentu sehingga membentuk satu kesatuan untuk melaksanakan suatu fungsi guna mencapai suatu tujuan. Suatu sistem mempunyai karakteristik sebagai berikut:

a. Mempunyai Komponen (Components)

   Suatu sistem terdiri dari sejumlah komponen yang bekerja sama membentuk suatu kesatuan, komponen, sistem terbagi menjadi suatu bagian-bagian dari sistem atau disebut subsistem, setiap subsistem memiliki sifat-sifat dari sistem utamanya yang akan mempengaruhi proses dari sistem keseluruhan.

b. Mempunyai Batas (Boundary)

   Batasan sistem merupakan hal yang membatasi antara sistem dengan sistem yang lain atau dengan lingkungan luarnya. Batas sistem ini memungkinkan

suatu kesatuan dan batas sistem menunjukkan ruang lingkup dari sistem tersebut.

c. Mempunyai Lingkungan (Environment)

Lingkungan adalah segala sesuatu yang berada di luar sistem yang mempengaruhi kinerja sistem.

d. Mempunyai Penghubung / antar muka (Interface) antar komponen

Penghubung (interface) adalah segala sesuatu yang bertugas untuk menjembatani hubungan antara komponen dalam sistem. Dengan penghubung ini setiap komponen dapat saling berinteraksi dan berkomunikasi dalam rangka menjalankan fungsi masing-masing.

e. Mempunyai Masukan (Input)

Masukkan adalah segala sesuatu yang perlu dimasukkan ke dalam sistem sebagai bahan yang akan diolah untuk menghasilkan keluaran (output) yang berguna.

f. Mempunyai Pengolahan (Processing)

Pengolahan merupakan komponen sistem yang mempunyai peran utama mengolah masukkan agar dapat menghasilkan suatu keluaran yang berguna bagi pemakainya. Dalam hal ini pengolahan adalah berupa program aplikasi.

g. Mempunyai Keluaran (Output)

Keluaran merupakan komponen sistem yang berupa berbagai macam bentuk keluaran yang dihasilkan oleh komponen pengolahan. Dalam hal ini keluaran adalah informasi yang dihasilkan oleh program aplikasi.

h. Mempunyai Sasaran (Objectives) dan Tujuan (Goal)

Sasaran merupakan segala sesuatu yang ingin dicapai oleh sistem dalam jangka waktu yang relatif pendek, sedangkan tujuan adalah hasil akhir yang ingin dicapai oleh sistem dalam jangka waktu yang relatif lebih panjang.

i. Mempunyai Kendali (Control)

Supaya setiap komponen yang berada pada sistem tetap bekerja sesuai dengan peran dan tugasnya masing-masing, maka dibutuhkan pengkontrol/pengendali sebagai penjaga setiap sistem tersebut. Bagian kontrol ini bertugas menjaga agar setiap proses/tugas dalam sistem dapat berjalan secara normal.

j. Mempunyai Umpan Balik (Feed Back)

Umpan balik diperlukan oleh bagian kendali (control) sistem untuk mengecek terjadinya penyimpangan yang terjadi dalam sistem dan mengembalikannya ke dalam kondisi normal.

Dari beberapa pengertian diatas dapat diambil kesimpulan bahwa sistem merupakan suatu kumpulan beberapa komponen yang saling berhubungan untuk melakukan suatu kegiatan atau menyelesaikan suatu sasaran tertentu.

2.2. Data

Menurut Laudon (2005), data adalah baris fakta yang mewakili peristiwa yang terjadi pada organisasi atau pada lingkungan fisik sebelum diolah ke dalam format yang bisa dimengerti dan digunakan orang.

Menurut Fatansyah (2001), data adalah representasi fakta dunia nyata yang mewakili suatu obyek, seperti manusia, barang, hewan, peristiwa, konsep, keadaan

dan sebagainya, yang direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi, atau kombinasinya.

Dari beberapa pengertian diatas dapat diambil kesimpulan bahwa data merupakan sekumpulan fakta-fakta yang menggambarkan suatu peristiwa-peristiwa nyata pada organisasi atau lingkungan nyata dan diolah menjadi format yang dapat di pahami orang lain.

2.3. Informasi

Menurut Murdick (1998), mendefinisikan bahwa informasi adalah data yang telah disusun sedemikian rupa sehingga bermakna dan bermanfaat karena dapat di komunikasikan kepada seseorang yang akan menggunakannya untuk membuat suatu keputusan.

Menurut Paker (1989), suatu informasi yang baik dan lengkap jika informasi tersebut mempunyai syarat-syarat sebagai berikut:

a. Ketersediaan

   Ketersediaan informasi merupakan syarat yang mendasar, maka informasi harus dapat diperoleh orang yang akan memanfaatkannya.

b. Mudah dipahami

   Informasi harus mudah dipahami oleh pembuat keputusan, informasi yang rumit akan membuat kurang efektifnya keputusan manajemen.

c. Relevan

   Dalam konteks organisasi, informasi yang diperlukan adalah yang benar-benar relevan dengan permasalahannya, misi dan tujuan organisasi.

d. Bermanfaat

Sebagai konsekuensi dari syarat relevansi, informasi juga harus bermanfaat bagi organisasi. Karena itu informasi juga harus tersaji dalam bentuk-bentuk yang memungkinkan pemakaian oleh organisasi yang bersangkutan.

e. Tepat waktu

Informasi harus tersedia tepat pada waktunya, karena informasi merupakan landasan dalam proses pengambilan keputusan.

f. Keandalan

Informasi harus diperoleh dari sumber-sumber yang dapat diandalkan kebenarannya. Pengolah data atau pemberi informasi harus dapat menjamin tingkat kepercayaan yang tinggi atas informasi yang disajikan.

g. Akurat

Syarat ini mengharuskan bahwa informasi bersih dari kekeliruan dan kesalahan. Informasi yang akurat mencerminkan makna yang terkandung dari data pendukungnya.

h. Konsisten

Informasi tidak boleh mengandung kontradiksi di dalam penyajiannya, karena konsistensi merupakan syarat penting bagi dasar pengambilan keputusan.

Dari beberapa pengertian diatas dapat diambil kesimpulan bahwa informasi adalah kumpulan dari data-data yang menjadi satu, dan saling berhubungan membentuk suatu kesatuan sehingga lebih bermanfaat serta memperlancar komunikasi guna menentukan suatu keputusan.

2.4. Sistem Informasi

Menurut Jogiyanto (2001), sistem informasi adalah suatu sistem yang terdapat di dalam organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategis dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan.

Menurut Laudon (2005), suatu sistem informasi dapat didefinisikan secara teknis sebagai satuan komponen yang saling berhubungan yang mengumpulkan (atau mendapatkan kembali), memproses, menyimpan dan mendistribusikan informasi untuk mendukung pengambilan keputusan dan kendali dalam suatu organisasi.

Dari beberapa pengertian diatas dapat diambil kesimpulan bahwa sistem informasi merupakan kumpulan dari data-data yang menjadi satu dan saling berhubungan yang membentuk suatu kesatuan sehingga lebih bermanfaat serta memperlancar informasi untuk mendukung pengambilan keputusan dalam suatu organisasi.

2.5. Manajemen

Menurut Sutanta (2003), manajemen dapat didefinisikan sebagai proses memanfaatkan berbagai sumber daya yang tersedia untuk mencapai suatu tujuan. Manajemen juga dapat dimaksudkan sebagai suatu sistem kekuasaan dalam suatu organisasi agar orang-orang menjalankan pekerjaan. Umumnya sumber daya tersedia dalam manajemen meliputi manusia, material, dan modal.

Dari pengertian diatas dapat diambil kesimpulan bahwa manajemen adalah suatu sistem yang digunakan suatu organisasi untuk memanfaatkan sumber daya yang tersedia agar tercapainya suatu tujuan.

2.6. Sistem Informasi Manajemen

Menurut Sutanta (2003), sistem informasi manajemen dapat didefinisikan sebagai sekumpulan subsistem yang saling berhubungan, berkumpul, bersama-sama dan membentuk suatu kesatuan, saling berinterakasi dan bekerja sama antara bagian satu dengan yang lainnya. Cara-cara melakukan fungsi pengolahan data, menerima masukan *(input)* berupa data-data, kemudian mengolahnya *(proccesing)*, dan menghasilkan keluaran *(output)* berupa informasi sebagai dasar pengambilan keputusan yang berguna dan mempunyai nilai nyata serta dapat di rasakan, akibatnya baik pada saat itu juga maupun dimasa mendatang. Informasi juga dapat mendukung kegiatan operasional, manajerial, dan strategi organisasi, dengan memanfaatkan berbagai sumber daya yang ada dan tersedia bagi fungsi tersebut guna mencapai tujuan.

Menurut Jogiyanto (2001), sistem informasi manajemen merupakan penerapan sistem informasi sebagai pendukung informasi yang dibutuhkan oleh semua tingkatan manajemen.

Dari beberapa pengertian yang telah diuraikan diatas dapat diambil kesimpulan bahwa sistem informasi manajemen merupakan sistem informasi yang diciptakan dan diterapkan untuk mendukung informasi-informasi yang dibutuhkan

oleh semua tingkatan manajemen pada suatu organisasi dalam pembuatan keputusan.

2.7. Basis Data

Menurut Fathansyah (1999), basis data terdiri dari 2 kata yaitu Basis dan Data. Basis kurang lebih dapat diartikan sebagai markas atau gudang, tempat bersarang atau berkumpul. Sedangkan Data adalah representasi fakta dunia nyata yang mewakili suatu objek seperti manusia (pegawai, siswa, pembeli, pelanggan), barang, hewan, peristiwa, konsep, keadaan, dan sebagainya, yang direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi, atau kombinasinya.

Basis Data sendiri dapat didefinisikan dalam sejumlah sudut pandang, seperti :

1. Himpunan kelompok data (arsip) yang saling berhubungan diorganisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redundansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan.
3. Kumpulan file/tabel/arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronis.

Menurut Jogiyanto (2001), basis data adalah kumpulan data satu dengan yang lainnya yang tersimpan dalam satu tempat penyimpanan luar dan

membutuhkan suatu perangkat lunak untuk menjalankannya. Suatu sistem manajemen basis data dibentuk oleh elemen-elemen basis data sebagai berikut:

a) Entitas

   Entitas merupakan orang, tempat, kejadian atau konsep yang informasinya direkam.

b) Attribute

   Suatu objek memiliki attribute yang menyatakan ciri dari objek tersebut.

c) Data value

   Data adalah data aktual atau informasi yang tersimpan pada tiap elemen atau attribute.

d) Record

   Merupakan elemen-elemen yang saling berkaitan menginformasikan suatu entitas, suatu record mewakili suatu data.

e) File

   Merupakan kumpulan dari record sejenis yang mempunyai panjang elemen dan attribute yang sama namun berbeda data valuenya adalah suatu file data.

f) Database

   Merupakan suatu koleksi data tertentu dari kumpulan file yang mempunyai kaitan antara satu file dengan file yang lain, sehingga membentuk suatu bangunan data.

Menurut Kristanto (2004), basis data adalah kumpulan file-file yang mempunyai kaitan antara satu file dengan file yang lain sehingga membentuk satu

bangunan data untuk menginformasikan satu perusahaan, instansi dalam batasan tertentu.

Dari beberapa pengertian diatas dapat diambil kesimpulan bahwa basis data merupakan kumpulan data yang disimpan pada media tertentu sehingga dapat diakses dengan mudah jika sewaktu-waktu data dibutuhkan.

2.8. Sistem Pengelola Basis Data (Database Management System/DBMS)

Menurut Fathansyah (1999), pengelolaan basis data secara fisik tidak dilakukan oleh pemakai secara langsung, tetapi ditangani oleh sebuah perangkat lunak (sistem) yang khusus/spesifik. Perangkat lunak inilah (disebut DBMS) yang akan menentukan bagaimana data diorganisasi, disimpan, diubah dan diambil kembali.

Perangkat lunak yang termasuk DBMS seperti dBase III+, dBase IV, FokBase, Rbase, MS-Access dan Borland-Paradox, MuSQL (untuk kelas sederhana) atau Borland Interbase, MS-SQLServer, CA-Open Ingres, Oracle, Informix dan Sybase (untuk kelas kompeks/berat).

2.9. MySQL

MySQL adalah sebuah implementasi dari sistem manajemen basis data relasional (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (General Public License). Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan

salah satu konsep utama dalam basis data yang telah ada sebelumnya; SQL (Structured Query Language). SQL adalah sebuah konsep pengoperasian basis data, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Dari awal pembuatannya, para pengembang *MySQL* menitik beratkan pengembangan *MySQL* pada sisi performa dan skalabilitasnya. Hasilnya adalah sebuah perangkat lunak yang sangat teroptimasi, walaupun dari sisi fitur memiliki kekurangan dibandingkan solusi basis data kelas enterprise lain. Akan tetapi *MySQL* menarik minat banyak pengguna. Saat ini, tercatat lebih dari lima juta basis data *MySQL* yang terpasang dan aktif di seluruh dunia.

MySQL memiliki beberapa keistimewaan diantara Relational Database Management System (DBMS)yang lain, diantaranya:

a. Portabilitas. MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, Mac Os X Server, Solaris, Amiga, dan masih banyak lagi.
b. Perangkat lunak sumber terbuka. MySQL didistribusikan sebagai perangkat lunak sumber terbuka, dibawah lisensi GPL sehingga dapat digunakan secara gratis.
c. Multi-user. MySQL dapat digunakan oleh beberapa pengguna dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.
d. *Performance tuning*, MySQL memiliki kecepatan yang menakjubkan dalam menangani query sederhana, dengan kata lain dapat memproses lebih banyak SQL per satuan waktu.

e. Ragam tipe data. MySQL memiliki ragam tipe data yang sangat kaya, seperti signed / unsigned integer, float, double, char, text, date, timestamp, dan lain-lain.

f. Perintah dan Fungsi. MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah Select dan Where dalam perintah (*query*).

g. Keamanan. MySQL memiliki beberapa lapisan keamanan seperti level subnetmask, nama host, dan izin akses *user* dengan sistem perizinan yang mendetail serta sandi terenkripsi.

h. Skalabilitas dan Pembatasan. MySQL mampu menangani basis data dalam skala besar, dengan jumlah rekaman (records) lebih dari 50 juta dan 60 ribu tabel serta 5 milyar baris. Selain itu batas indeks yang dapat ditampung mencapai 32 indeks pada tiap tabelnya.

i. Konektivitas. MySQL dapat melakukan koneksi dengan klien menggunakan protokol TCP/IP, Unix soket (UNIX), atau Named Pipes (NT).

j. Lokalisasi. MySQL dapat mendeteksi pesan kesalahan pada klien dengan menggunakan lebih dari dua puluh bahasa. Meski pun demikian, bahasa Indonesia belum termasuk di dalamnya.

k. Antar Muka. MySQL memiliki antar muka (interface) terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi API (Application Programming Interface).

l. Klien dan Peralatan. MySQL dilengkapi dengan berbagai peralatan (tool) yang dapat digunakan untuk administrasi basis data, dan pada setiap peralatan yang ada disertakan petunjuk online.

m. Struktur tabel. MySQL memiliki struktur tabel yang lebih fleksibel dalam menangani ALTER TABLE, dibandingkan basis data lainnya semacam PostgreSQL ataupun Oracle.

2.10. JAVA

1. Sejarah Singkat Java

Pada 1991, sekelompok insinyur Sun dipimpin oleh Patrick Naughton dan James Gosling ingin merancang bahasa komputer untuk perangkat konsumer seperti cable TV Box. Karena perangkat tersebut tidak memiliki banyak memori, bahasa harus berukuran kecil dan mengandung kode yang liat. Juga karena manufaktur – manufaktur berbeda memilih processor yang berbeda pula, maka bahasa harus bebas dari manufaktur manapun. Proyek diberi nama kode ”Green”.

Kebutuhan untuk fleksibilitas, kecil, liat dan kode yang netral terhadap platform mengantar tim mempelajari implementasi Pascal yang pernah dicoba. Niklaus Wirth, pencipta bahasa Pascal telah merancang bahasa portabel yang menghasilkan intermediate code untuk mesin hipotesis. Mesin ini sering disebut dengan mesin maya (virtual machine). Kode ini kemudian dapat digunakan di sembarang mesin yang memiliki interpreter. Proyek Green menggunakan mesin maya untuk mengatasi isu utama tentang netral terhadap arsitektur mesin.

Karena orang – orang di proyek Green berbasis C++ dan bukan Pascal maka kebanyakan sintaks diambil dari C++, serta mengadopsi orientasi objek dan bukan prosedural. Mulanya bahasa yang diciptakan diberi nama ”Oak” oleh James Gosling yang mendapat inspirasi dari sebuah pohon yang berada pada seberang

kantornya, namun dikarenakan nama Oak sendiri merupakan nama bahasa pemrograman yang telah ada sebelumnya, kemudian SUN menggantinya dengan JAVA. Nama JAVA sendiri terinspirasi pada saat mereka sedang menikmati secangkir kopi di sebuah kedai kopi yang kemudian dengan tidak sengaja salah satu dari mereka menyebutkan kata JAVA yang mengandung arti asal bijih kopi. Akhirnya mereka sepakat untuk memberikan nama bahasa pemrograman tersebut dengan nama Java.

Produk pertama proyek Green adalah Star 7 (*7), sebuah kendali jarak jauh yang sangat cerdas. Dikarenakan pasar masih belum tertarik dengan produk konsumer cerdas maka proyek Green harus menemukan pasar lain dari teknologi yang diciptakan. Pada saat yang sama, implementasi WWW dan Internet sedang mengalami perkembangan pesat. Di lain pihak, anggota dari proyek Green juga menyadari bahwa Java dapat digunakan pada pemrograman internet, sehingga penerapan selanjutnya mengarah menjadi teknologi yang berperan di web.

Java telah mengakomodasi hampir seluruh fitur penting bahasa – bahasa pemrograman yang ada semenjak perkembangan komputasi modern manusia :

1. Dari SIMULA, bahasa pada tahun 65-an, bahasa yang paling mempengaruhi Java sekaligus C++. Dari bahasa ini diadopsi bentukan – bentukan dasar dari pemrograman berorientasi objek.
2. Dari LISP – bahasa tahun 55-an. Diadopsi fasilitas garbage collection, serta kemampuan untuk meniru generic list processing, meski fasilitas ini jarang yang memanfaatkannya.

3. Dari Algol – bahasa pada tahun 60-an, diambil struktur kendali yang dimilikinya.
4. Dari C++, diadopsi sintaks, sebagian semantiks dan exception handling
5. Dari bahasa Ada, diambil strongly type, dan exception handling.
6. Dari Objective C, diambil fasilitas interface.
7. Dari bahasa SmallTalk, diambil pendekatan single-root class hiérarchie, dimana objek adalah satu kesatuan hirarki pewarisan
8. Dari bahasa Eiffel, fasilitas assertion yang mulai diterapkan di sebagian JDK 1.4

Sebagai salah satu bahasa pemograman, teknologi java dapat di uraikan sebagai berikut:

- **Sebuah Bahasa Pemrograman**

  Sebagai sebuah bahasa pemrograman, Java dapat membuat seluruh bentuk aplikasi, desktop, web dan lainnya, sebagaimana dibuat dengan menggunakan bahasa pemrograman konvensional yang lain.

  Java adalah bahasa pemrograman yang berorientasi objek (OOP) dan dapat dijalankan pada berbagai platform sistem operasi. Perkembangan Java tidak hanya terfokus oada satu sistem operasi, tetapi dikembangkan untuk berbagai sistem operasi dan bersifat open source.

- **Sebuah Development Environment**

  Sebagai sebuah peralatan pembangun, teknologi Java menyediakan banyak tools : compiler, interpreter, penyusun dokumentasi, paket kelas dan sebagainya.

- **Sebuah Aplikasi**

  Aplikasi dengan teknologi Java secara umum adalah aplikasi serbt a guna yang dapat dijalankan pada seluruh mesin yang memiliki Java Runtime Environment (JRE).

- **Sebuah Deployment Environment**

  Terdapat dua komponen utama dari Deployment Environment. Yang pertama adalah JRE, yang terdapat pada paket J2SDK, mengandung kelas – kelas untuk semua paket teknologi Java yang meliputi kelas dasar dari Java, komponen GUI dan sebagainya. Komponen yang lain terdapat pada Web Browser. Hampir seluruh Web Browser komersial menyediakan interpreter dan runtime environment dari teknologi Java.

2. Kelebihan Bahasa Java

Berdasarkan white paper resmi dari SUN, Java memiliki karakteristik berikut :

1. Sederhana (Simple) Bahasa pemrograman Java menggunakan Sintaks mirip dengan C++ namun sintaks pada Java telah banyak diperbaiki terutama menghilangkan penggunaan pointer yang rumit dan multiple inheritance. Java juga menggunakan automatic memory allocation dan memory garbage collection.
2. Berorientasi objek (Object Oriented) Java mengunakan pemrograman berorientasi objek yang membuat program dapat dibuat secara modular dan dapat dipergunakan kembali. Pemrograman berorientasi objek memodelkan dunia nyata kedalam objek dan melakukan interaksi antar objek-objek tersebut.

3. Terdistribusi (Distributed) Java dibuat untuk membuat aplikasi terdistribusi secara mudah dengan adanya libraries networking yang terintegrasi pada Java.
4. Interpreted Program Java dijalankan menggunakan interpreter yaitu Java Virtual Machine (JVM). Hal ini menyebabkan source code Java yang telah dikompilasi menjadi Java bytecodes dapat dijalankan pada platform yang berbeda-beda.
5. Robust Java mempuyai reliabilitas yang tinggi. Compiler pada Java mempunyai kemampuan mendeteksi error secara lebih teliti dibandingkan bahasa pemrograman lain. Java mempunyai runtime-Exception handling untuk membantu mengatasi error pada pemrograman.
6. Secure Sebagai bahasa pemrograman untuk aplikasi internet dan terdistribusi, Java memiliki beberapa mekanisme keamanan untuk menjaga aplikasi tidak digunakan untuk merusak sistem komputer yang menjalankan aplikasi tersebut.
7. Architecture Neutral Program Java merupakan platform independent. Program cukup mempunyai satu buah versi yang dapat dijalankan pada platform berbeda dengan Java Virtual Machine.
8. Portable Source code maupun program Java dapat dengan mudah dibawa ke platform yang berbeda-beda tanpa harus dikompilasi ulang.
9. Performance Performance pada Java sering dikatakan kurang tinggi. Namun performance Java dapat ditingkatkan menggunakan kompilasi Java lain seperti buatan Inprise, Microsoft ataupun Symantec yang menggunakan Just In Time Compilers (JIT).

10. Multithreaded Java mempunyai kemampuan untuk membuat suatu program yang dapat melakukan beberapa pekerjaan secara sekaligus dan simultan.
11. Dynamic Java didesain untuk dapat dijalankan pada lingkungan yang dinamis. Perubahan pada suatu class dengan menambahkan properties ataupun method dapat dilakukan tanpa menggangu program yang menggunakan class tersebut.

Pada tahun 2009 Oracle telah berhasil mengakuisisi Sun dengan nilai total US$ 7,4 miliar. Pekan lalu penawaran US$ 9,50 per saham oleh Oracle sudah disetujui oleh dewan direksi Sun. "Akuisisi Sun mengubah industri TI, memadukan software enterprise terbaik di kelasnya dengan sistem komputasi mission-critical," kata Larry Ellison (CEO, Oracle) dalam rilis persnya.

2.11. NETBEANS

NetBeans mengacu pada dua hal, yakni platform untuk pengembangan aplikasi desktop java, dan sebuah *Integrated Development Environment* (IDE) yang dibangun menggunakan platform NetBeans.

Platform NetBeans memungkinkan aplikasi dibangun dari sekumpulan komponen perangkat lunak moduler yang disebut 'modul'. Sebuah modul adalah suatu arsip Java (Java *archive*) yang memuat kelas-kelas Java untuk berinetraksi dengan*NetBeans Open API* dan file manifestasi yang mengidentifikasinya sebagai modul. Aplikasi yang dibangun dengan modul-modul dapat dikembangkan dengan menambahkan modul-modul baru. Karena modul dapat dikembangkan secara

independen, aplikasi berbasis platform NetBeans dapat dengan mudah dikembangkan oleh pihak ketiga secara mudah dan *powerful*.

1. Sejarah

Pengembangan NetBeans diawali dari Xelfi, sebuah proyek mahasiswa tahun 1997 di bawah bimbingan Fakultas Matematika dan Fisika Universitas Charles, Praha. Sebuah perusahaan kemudian dibentuk untuk proyek tersebut dan menghasilkan versi komersial NetBeans IDE hingga kemudian dibeli oleh Sun Microsystem pada tahun 1999. Sun kemudian menjadikan NetBeans open source pada bulan Juni tahun 2000. Sejak itu komunitas NetBeans terus berkembang.

2. Platform NetBeans

Platform NetBeans adalah framework yang dapat digunakan kembali (*reusable*) untuk menyederhanakan pengembangan aplikasi desktop. Ketika aplikasi berbasis platform NetBeans dijalankan, kelas Main dari platform dieksekusi. Modul-modul yang tersedia ditempatkan di sebuah *registry* di dalam memori, dan tugas *startup* modul dijalankan. Secara umum, kode modul dimuatkan ke dalam memori hanya ketika ia diperlukan.

Aplikasi dapat menginstal modul secara dinamis. Aplikasi dapat memasukkan modul *Update Center* untuk mengijinkan pengguna aplikasi men-*download digitally-signed upgrade* dan fitur-fitur baru secara langsung ke dalam aplikasi yang berjalan. Penginstalan kembali sebuah upgrade atau rilis baru tidak memaksa pengguna untuk men-*download* keseluruhan aplikasi lagi.

Platform NetBeans menawarkan layanan-layanan yang umum bagi aplikasi desktop, mengijinkan pengembang untuk fokus ke logika yang spesifik terhadap aplikasi. Fitur-fitur yang disediakan oleh platform NetBeans:

- Manajemen antarmuka (misal: menu & toolbar)
- Manajemen pengaturan pengguna
- Manajemen penyimpanan (menyimpan dan membuka berbagai macam data)
- Manajemen jendela
- *Wizard framework* (mendukung dialog langkah demi langkah)

3. NetBeans IDE

NetBeans **IDE** adalah IDE open source yang ditulis sepenuhnya dengan bahasa pemrograman Java menggunakan platform NetBeans. NetBeans IDE mendukung pengembangan semua tipe aplikasi Java (J2SE, web, EJB, dan aplikasi *mobile*). Fitur lainnya adalah sistem proyek berbasis Ant, kontrol versi, dan *refactoring*.

Versi terbaru saat ini adalah NetBeans IDE 5.5.1 yang dirilis Mei 2007 mengembangkan fitur-fitur Java EE yang sudah ada (termasuk *Java Persistence support*, EJB-3 dan JAX-WS). Sementara paket tambahannya, NetBeans Enterprise Pack mendukung pengembangan aplikasi perusahaan Java EE 5, meliputi alat desain visual SOA, skema XML, *web service* dan pemodelan UML. NetBeans C/C++ Pack mendukung proyek C/C++.

Modularitas: Semua fungsi IDE disediakan oleh modul-modul. Tiap modul menyediakan fungsi yang didefinisikan dengan baik, seperti dukungan untuk

bahasa pemrograman Java, editing, atau dukungan bagi CVS. NetBeans memuat semua modul yang diperlukan dalam pengembangan Java dalam sekali *download*, memungkinkan pengguna untuk mulai bekerja sesegera mungkin. Modul-modul juga mengijinkan NetBeans untuk bisa dikembangkan. Fitur-fitur baru, seperti dukungan untuk bahasa pemrograman lain, dapat ditambahkan dengan menginstal modul tambahan. Sebagai contoh, Sun Studio, Sun Java Studio Enterprise, dan Sun Java Studio Creator dari Sun Microsystem semuanya berbasis NetBeans IDE.

4. Lisensi

Sejak Juli 2006, NetBeans IDE dilisensikan di bawah *Common Development and Distribution License* (CDDL), yaitu lisensi yang berbasis *Mozilla Public License* (MPL).

5. Paket-Paket Tambahan NetBeans IDE

1. NetBeans Mobility Pack

   NetBeans Mobility Pack adalah alat untuk mengembangkan aplikasi yang berjalan pada perangkat bergerak (*mobile*), umumnya telepon seluler, tetapi juga mencakup PDA, dan lain-lain.

   NetBeans Mobility Pack dapat digunakan untuk menulis, menguji, dan *debugging* aplikasi untuk perangkat bergerak yang menggunakan teknologi berplatform Java Micro Edition (platform Java ME). Paket ini mengintegrasikan dukungan terhadap*Mobile Information Device Profile* (MIDP) 2.0, *Connected Limited Device Configuration* (CLDC)

1.1, dan *Connected Device Configuration (CDC). Emulator dari pihak ketiga dapat diintegrasikan dengan mudah untuk lingkungan pengujian yang lebih kokoh.*NetBeans Mobility Pack *saat ini tersedia dalam dua klaster yang berbeda, yang satu memuat CDC dan yang lainnya CLDC.*

2. *NetBeans Profiler*

   NetBeans ***Profiler*** adalah alat untuk mengoptimalkan aplikasi Java, membantu menemukan kebocoran memori dan mengoptimalkan kecepatan.

   Profiler ini berdasarkan sebuah proyek riset *Sun Laboratories* yang dahulu bernama Jfluid. Riset tersebut mengungkap teknik tertentu yang dapat digunakan untuk menurunkan *overhead* proses *profiling* aplikasi Java. Salah satu dari teknik tersebut adalah instrumentas i kode byte dinamis, yang berguna untuk *profiling* aplikasi Java yang besar. Dengan menggunakan instrumentasi kode byte dinamis dan algoritma-algoritma tambahan, *Netbeans Profiler* mampu mendapatkan informasi *runtime*aplikasi yang terlalu besar atau kompleks bagi *profiler* lain. NetBeans IDE 6.0 akan mendukung *Profiling Point* yang memungkinkan kita memprofilkan titik yang tepat dari eksekusi dan mengukur waktu eksekusi.

3. *NetBeans C/C++ Pack*

   NetBeans ***C/C++ Pack*** menambahkan dukungan terhadap pengembang C/C++ ke NetBeans IDE 5.5. Paket ini memperbolehkan pengembang menggunakan sekumpulan kompiler dan alat sendiri bersama dengan

NetBeans IDE untuk membangun aplikasi *native* untuk MS Windows, Linux, dan Solaris. Paket ini membuat editor mengenali bahasa C/C++ dan menyediakan *project template*, browser kelas yang dinamis, dukungan pembuatan file dan fungsionalitas *debugger*. Para pengembang juga dapat mengembangkan paket tersebut dengan fungsionalitas tambahan mereka sendiri.

4. *NetBeans Enterprise Pack*

   NetBeans Enterprise Pack memperluas dukungan terhadap pengembangan aplikasi perusahaan dan *web service* di NetBeans IDE 5.5.

   Enterprise Pack ini mengembangkan kemampuan untuk menulis, menguji, dan *debug* aplikasi dengan arsitektur berorientasi layanan (*Service-Oriented Architecture*) menggunakan XML, BPEL, dan Java *web service*. Paket ini menambahkan alat desain visual untuk pemodelan UML, skema XML, dan *web service orchestration*, juga dukungan untuk *web service* dengan menggunakan identitas yang aman. Paket ini juga menginstal dan mengkonfigurasi *runtime* yang diperlukan, termasuk mesin BPEL dan server manajemen identitas yang terintegrasi dengan *Sun Java System Application Server*.

5. *NetBeans Ruby Pack*

   Versi NetBeans 6.0 mendatang akan mengijinkan pengembangan IDE menggunakan Ruby dan Jruby, sebagaimana Rails untuk dua implementasi Ruby yang lain. *Preview NetBeans Ruby Pack* tersedia sejak rilis *Milestone 7* NetBeans 6.

Ruby Pack memasukkan fungsionalitas editor seperti:

- Pengeditan dasar
- Pewarnaan sintaks untuk ruby
- Pelengkapan kode
- *Occurence* highlighting
- Pop-up dokumentasi yang terintegrasi untuk pemanggilan ruby api
- Analisis semantik dengan *highlighting* parameter dan variabel lokal yang tidak terpakai

6. *NetBeans JavaScript Editor*

NetBeans JavaScript ***Editor*** menyediakan perluasan dukungan terhadap JavaScript dan CSS. Fitur-fiturnya antara lain:

1. Editor JavaScript
    1. *Syntax highlighting*
    2. Pelengkapan kode untuk objek dan fungsi *native*
    3. Semua fitur dalam editor netbeans
    4. Pembuatan kerangka kelas javascript secara otomatis
    5. Pembuatan pemanggilan ajax dari *template*
2. Ekstensi editor CSS
    6. Pelengkapan kode untuk nama-nama *style*
    7. Navigasi cepat melalui panel navigator
    8. Penampilan deklarasi aturan css di *list view*
    9. Penampilan struktur file di *tree view*

10. Mengurutkan *outline view* berdasarkan nama, tipe, atau urutan deklarasi (*list & tree*)
11. Pembuatan deklarasi aturan (hanya *tree*)
12. Pemfaktoran kembali sebagian nama rule (hanya *tree*)

2.12. BIOMETRIK

Biometrik (berasal dari bahasa Yunani bios yang artinya hidup dan metron yang artinya mengukur) secara umum adalah studi tentang karakteristik biologi yang terukur. Dalam dunia teknologi informasi, biometrik relevan dengan teknologi yang digunakan untuk menganalisa fisik dan kelakuan manusia dalam autentifikasi.

Pengidentifikasi biometrik sangat khas, karakteristik yang terukur digunakan untuk mengidentifikasi individu. Dua kategori pengidentifikasi biometrik meliputi karakteristik fisiologis dan perilaku. Karakteristik fisiologis berhubungan dengan bentuk tubuh, dan termasuk tetapi tidak terbatas pada: sidik jari, pengenalan wajah, DNA, telapak tangan, geometri tangan, pengenalan iris (yang sebagian besar telah diganti retina), dan bau/aroma. Karakteristik perilaku terkait dengan perilaku seseorang, termasuk namun tidak terbatas pada:Ritme mengetik, kiprah, dan suara.

1. Pemindai biometrik

   Pemindai biometrik adalah alat yang menggunakan data biometrik untuk mengidentifikasi individu berdasarkan pengukuran karakteristik fisiologisnya. Karakteristik fisiologis ini memberikan kemampuan untuk mengontrol dan melindungi integritas data sensitif yang tersimpan dalam sistem informasi.

Biometrik merupakan suatu metode komputerisasi yang menggunakan aspek-aspek biologi terutama karakteristik unik yang dimiliki oleh manusia.

Karakterisitik fisiologi unik yang dapat digunakan adalah sidik jari dan retina mata. Kedua hal ini terdapat pada tubuh manusia namun selalu berbeda setiap orangnya, sehingga dapat dijadikan sandi untuk pengindentifikasian.

2.13. PAYROLL

1. Pengertian Payroll atau Gaji

Penggajian (payroll) biasa diartikan sebagai jumlah total yang dibayarkan kepada karyawan atas jasa-jasa yang mereka berikan selama periode tertentu. Penggajian adalah sesuatu yang penting, karyawan biasanya sangat sensitif terhadap kesalahan-kesalahan atau hal-hal yang tidak wajar dalam pelaksanaan penggajian. Untuk tetap membuat kinerja dan moral karyawan tetap tinggi, perusahaan harus membayar gaji secara akurat dan tepat waktu. Suatu perusahaan yang sudah maju maupun yang baru berkembang, ingin sekali memperoleh keuntungan secara maksimal dengan rencana jangka pendek, jangka menengah, maupun jangka panjang. Untuk memperoleh keuntungan yang diharapkan, perusahaan sangat tergantung pada kinerja para karyawan. Masalah yang penting diantara sekian banyak masalah yang berhubungan dengan para pegawai adalah masalah gaji. Dengan pemberian balas jasa yang setimpal, maka akan mendorong para karyawan untuk lebih giat bekerja dalam mengembangkan kegiatan usaha perusahaan.

Menurut Mulyadi (2008:373), menjelaskan bahwa :

"Gaji adalah pembayaran atas penyerahan jasa yang dilakukan oleh karyawan yang mempunyai jenjang jabatan manager, pegawai administrasi, supervisor dan lain-lain, dan pada umunya gaji dibayarkan secara tetap tiap perbulan".

Menurut Soemarso S.R (2005 : 307), menjelaskan bahwa :

"Gaji adalah imbalan kepada pegawai yang diberi tugas-tugas administratif dan pimpinan yang jumlahnya, biasanya tetap secara bulanan atau tahunan".

Menurut Amir Abadi Yusuf (2000 : 285), menjelaskan bahwa :

"Penggajian adalah sistem yang menyajikan cara-cara penggajian pegawai secara memadai dan akurat, menghasilkan laporan-laporan penggajian yang diperlukan, dan menyajikan informasi kebutuhan pegawai kepada manajemen".

Menurut Niswonger, Fess dan Warren (2005 : 314), menjelaskan bahwa :

"Penggajian (payroll) adalah jumlah total yang dibayarkan kepada karyawan atas jasa-jasa yang mereka berikan selama suatu periode tertentu ".

Dari definisi diatas, penulis dapat menarik kesimpulan gaji adalah pembayaran balas jasa atau penghargaan seseorang atas hasil kerja seseorang, jasa manajemen, administratif, dan jasa-jasa lainnya yang telah dilakukan oleh karyawan dalam satu periode tertentu dan dibayarkan secara konstan dalam periode yang telah ditentukan.

Berdasarkan pengertian gaji diatas, bisa dilihat bahwa gaji merupakan hal yang penting karena alasan tertentu. Para karyawan sangat sensitive terhadap kesalahan atau ketidakwajaran dalam gaji.

2.14. UML

UML singkatan dari *Unified Modeling Language* yang berarti bahasa pemodelan standar. (chonoles, 2003:bab1) mengatakan sebagai bahasa, berarti UML memiliki sintaks dan sematik. Ketika kita membuat model menggunakan konsep UML ada aturan-aturan yang harus diikuti. Bagaimana elemen pada model-model yang kita buat berhubungan satu dengan lainnya harus mengikuti standar yang ada.[11]

UML diaplikasikan untuk maksud tertentu, antara lain untuk :

1. Merancang perangkat lunak
2. Sarana komunikasi antara perangkat lunak dengan proses bisnis
3. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasi sistem yang ada, proses-proses dan organisasinya.

Pada UML 2.3 ada 9 macam diagram yang dikelompokkan menjadi tiga kategori, yaitu :

1. Structure diagram, yaitu kumpulan diagram yang digunakan untuk menggammbarkan suatu struktur statis dalam sistem yang dimodelkan.
2. Behavior diagram, yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
3. Interaction diagram, yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antarsubsistem pada suatu sistem.

1) Class Diagram

Class Diagram adalah sebuah spesifikasi yang jika diinstansikan akan menghasilkan sebuah obyek dan merupakan inti dari pengembangan dan desain berorientasi objek.[11]

Class menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metode/fungsi). Class diagram menggambarkan struktur dan deskripsi *class*, *package* dan *object* beserta hubungan satu sama lain seperti *containment*, *pewarisan*, *asosiasi*, dan lain-lain. Sebuah class memiliki tiga area pokok:

1. Nama , merupakan nama dari sebuah kelas.
2. Atribut, merupakan properti dari sebuah kelas. Atribut melambangkan batas nilai yang mungkin ada pada obyek dari class.
3. Operasi, adalah sesuatu yang bisa dilakukan oleh sebuah atau yang dapat dilakukan oleh class lain terhadap sebuah class. Atribut dan metoda dapat memiliki salah sifat berikut:
    a. *Private*, tidak dapat dipanggil dari luar class yang bersangkutan
    b. *Protected*, hanya dapat dipanggil oleh class yang bersangkutan dan anak-anak yang mewarisinya.
    c. *Public*, dapat dipanggil oleh siapa saja.
    d. *Package*, hanya dapat dipanggil oleh instance sebuah class pada paket yang sama.

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

tetapi harus diimplementasikan dahulu menjadi sebuah class. Dengan demikian *interface* mendukung resolusi metoda pada saat *runtime*.

Relasi atau *relationship* merupakan keterhubungan antar kelas yang muncul pada saat sebuah kelas berinteraksi dengan kelas-kelas lainnya. Didalam *class diagram,* setiap kelas pasti akan berinteraksi dengan baik satu maupun lebih kelas. Relasi yang muncul pada setiap keterhubungan antar kelas juga memiliki atribut-atribut yang akan lebih menjelaskan defenisi dari sebuah relasi yang terjadi.

1. Asosiasi (*Association*)

   Para pemodel menggunakan pemahaman asosiasi adalah pada saat beberapa kelas saling terhubung satu sama lain secara konseptual. Asosiasi juga dapat menjadi lebih kompleks pada saat beberapa kelas terhubung ke satu kelas

2. Constraint pada Asosiasi

   Kadangkala sebuah asosiasi diantara dua *class* harus mengikuti sebuah aturan dan aturan ini bias diletakkan dalam sebuah constraint pada garis asosiasi dan diletakkan dalam kurung kurawal. Bentuk lain dari tipe constraint adalah relasi OR yang ditulis dengan {*or*} dalam garis putus-putus yang menghubungkan dua garis asosiasi. Kondisi OR ini menghadapkan kepada keadaan bahwa sebuah kelas terhubung dengan dua kelas tetapi pada saat mendefenisikan relasinya kelas harus memilih salah satu dari kedua kelas tersebut, sedangkan kondisi untuk memilih keduanya adalah invalid.

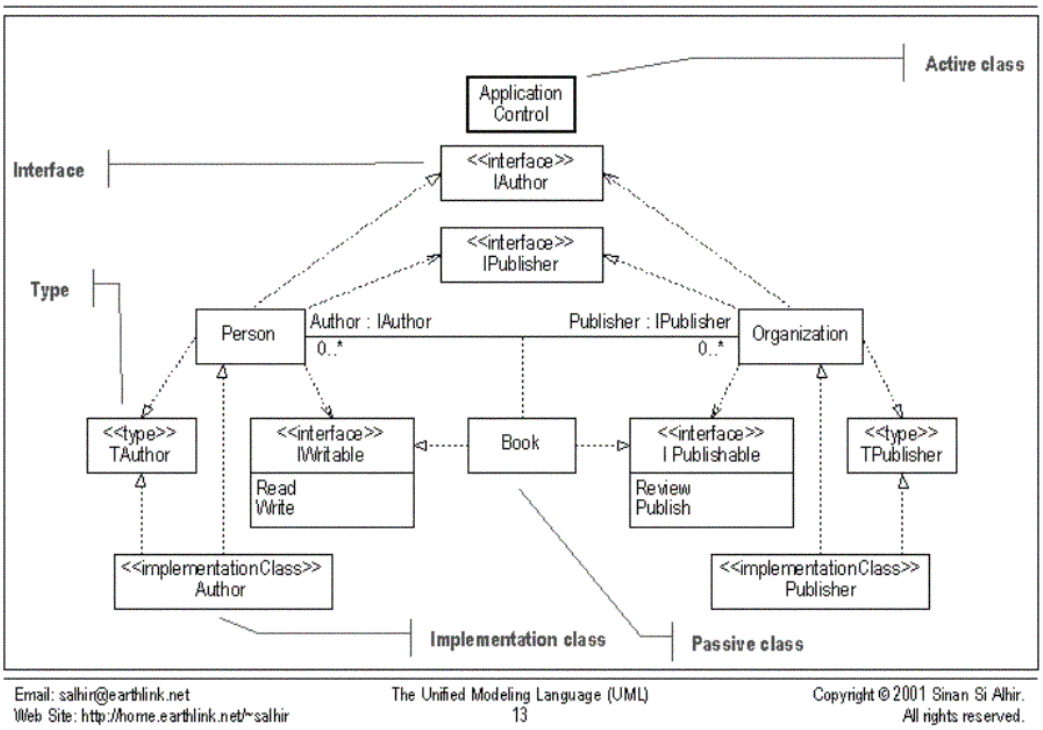
3. Kelas Asosiasi (*Association Class*)

   Sebuah asosiasi dapat memiliki atribut dan operasi sepertinya halnya sebuah *class*. Sebuah *association class* sebenarnya diperlukan apabila salah satu dari

kelas yang terhubung mempunya sebuah atau beberapa atribut yang tidak layak dimiliki kelas tersebut, karena secara logis atribut tersebut lebih layak dimiliki oleh asosiasi yang menghubungkan kedua kelas tersebut.

4. Multiplisitas (*Multiplicity*)

   Multiplisitas atau kardinalitas adalah jumlah banyaknya objek sebuah class yang berelasi dengan sebuah objek lain pada class lain yang berasosiasi pada class tersebut. Untuk menyatakan multiplisitas anda dapat meletakkannya diatas garis asosiasi berdekatan dengan class yang sesuai. Ada banyak multiplisitas yang mingkin untuk dipakai.



Gambar 2. 1. Class Diagram

2) Use Case Diagram

Use case adalah deskripsi fungsi dari sebuah sistem dari perspektif / sudut pandang para pengguna sistem. *Use case* mendefenisikan “apa” yang dilakukan

oleh sistem dan elemen-elemennya saling berinteraksi [11]. *Use case* bekerja dengan menggunakan "*scenario*", yaitu deskripsi urutan-urutan langkah yang menerangkan apa yang dilakukan pengguna terhadap sistem maupun sebaliknya. *Use case* diagram mengindentifikasin fungsionalitas yang dipunyai oleh sistem (*use case*), user yang berinteraksi dengan sistem dan asosiasi/ keterhubungan antara user dengan fungsionalitas sistem.

Fungsi dari diagram use *case* adalah menggambarkan fungsionalitas yang diharapkan dari sebuah sistem, dengan menekankan aspek-aspek apa yang dilakukan sistem bukan bagaimana sistem melakukannya. Sebuah *use case* merepresentasikan sebuah interaksi antara actor dengan sistem.

Berbicara mengenai *use case* diagram tidak akan terlepas dengan hal yang disebut *stereotype*. *Stereotype* adalah sebuah model khusus yang terbatas untuk kondisi tertentu. Untuk menunjukkan *stereotype* digunakan symbol "<<" diawalnya dan ">>" diakhirnya. Terdapat 2 *stereotype* paling sering digunakan dalam *use case* yaitu << *include* >> dan << *extend* >>.

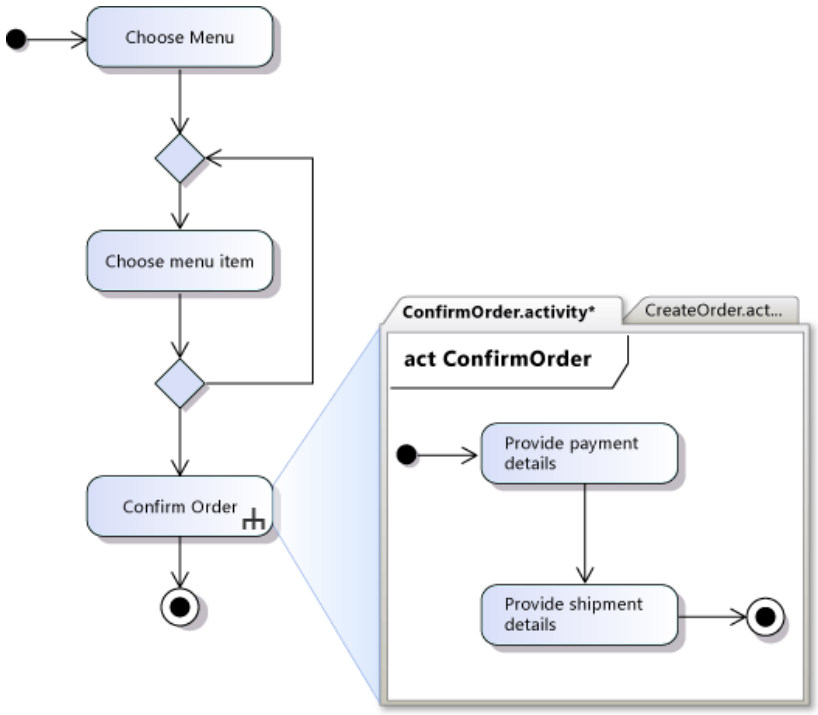
<< *extend* >> digunakan untuk menunjukkan bahwa satu *use case* merupakan tambahan fungsional dari *use case* yang lain jika kondisi atau syarat tertentu dipenuhi. Sedangkan << *include* >> digunakan untuk menggambarkan bahwa suatu *use case* seluruhnya merupakan fungsionalitas dari *use case* lainnya.



Gambar 2. 2. Use Diagram Case

3) Activity Diagram

Activity diagram digunakan untuk mendokumentasikan alur kerja pada sebuah sistem, yang dimulai dari pandangan *business* level hingga ke operational *level* [11]. Pada dasarnya, *activity diagram* merupakan variasi dari *statechart diagram*. *Activity diagram* mempunyai peran seperti halnya *flowchart*, akan tetapi perbedaannya dengan *flowchart* adalah *activity diagram* bisa mendukung perilaku parallel sedangkan flowchart tidak bisa.



Gambar 2. 3. Activity Diagram

4) Sequence Diagram

Sequence diagram menurut Munawar adalah grafik dua dimensi dimana objek ditunjukkan dalam dimensi horizontal, sedangkan *lifeline* ditunjukkan dalam dimensi vertikal[11].

5) Component Diagram

Komponen perangkat lunak adalah bagian fisik dari sebuah sistem yang menetap di komputer. komponen merupakan implementasi *software* dari sebuah *class*. Komponen bisa berupa tabel, file data, file exe, file DLL, dokumen dan lain-lain.

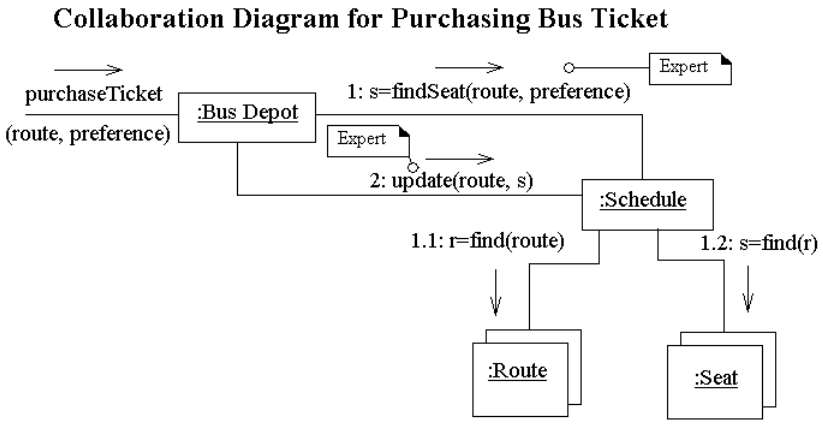
Component diagram mengandung komponen, *interface* dan *relationship*. Komponen diagram ini digunakan pada saat anda ingin memecah sistem menjadi komponen-komponen dan ingin menampilkan hubungan-hubungan mereka dengan antarmuka atau pemecahan komponen menjadi struktur yang lebih rendah. Secara umum dapat kita katakan bahwa *component diagram* kita gunakan untuk menjelaskan kebergantungan antar beragam komponen-komponen *software* seperti misalnya kebergantungan antara file-file executable dengan file-file sumbernya (source file) dll. Berikut adalah notasi dari component diagram :



Gambar 2. 4. Contoh Notasi Component

6) Collaboration Diagram

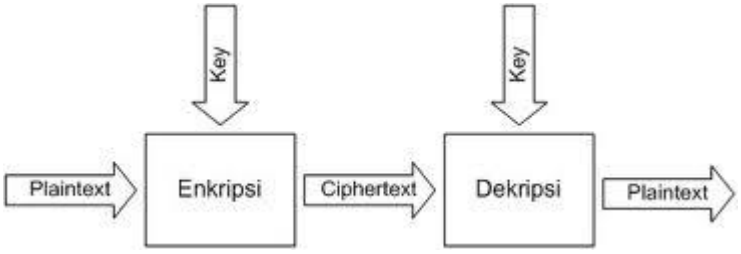
Collaboration diagram menurut Munawar adalah perluasan dari objek diagram. Objek diagram menunjukkan objek-objek yang hubungannya satu dengan yang lain. *Collaboration diagram* menunnjukkan *message-message* objek yang dikirim satu sama lain [11].



Gambar 2. 5. Collaboration Diagram

7) Enkripsi

Enkripsi digunakan untuk menyandikan data-data atau informasi sehingga tidak dapat dibaca oleh orang yang tidak berhak. Dengan *enkripsi* data anda disandikan (*encrypted*) dengan menggunakan sebuah kunci (*key*). Untuk membuka (*decrypt*) data tersebut digunakan juga sebuah kunci yang dapat sama dengan kunci untuk mengenkripsi (untuk kasus *private key cryptography*) atau dengan kunci yang berbeda (untuk kasus *public key cryptography*). [3]



Gambar 2. 6. Diagram Proses Enkripsi dan Dekripsi

Secara matematis, proses atau fungsi enkripsi (E) dapat dituliskan sebagai:

E(M) = C

dimana: M adalah *plaintext* (*message*) dan C adalah *ciphertext*.

Proses atau fungsi *dekripsi* (D) dapat dituliskan sebagai:

D(C) = M

Algoritma dari *Enkripsi* dan *Dekripsi*. Algoritma dari *enkripsi* adalah fungsi-fungsi yang digunakan untuk melakukan fungsi *enkripsi* dan *dekripsi*. Algoritma yang digunakan menentukan kekuatan dari enkripsi, dan ini biasanya dibuktikan dengan basis matematika.

Kunci yang digunakan dan panjangnya kunci. Kekuatan dari penyandian bergantung kepada kunci yang digunakan. Beberapa algoritma *enkripsi* memiliki kelemahan pada kunci yang digunakan. Untuk itu, kunci yang lemah tersebut tidak boleh digunakan. Selain itu, panjangnya kunci, yang biasanya dalam ukuran bit, juga menentukan kekuatan dari *enkripsi*. Kunci yang lebih panjang biasanya lebih aman dari kunci yang pendek. Jadi enkripsi dengan menggunakan kunci 128-bit lebih sukar dipecahkan dengan algoritma enkripsi yang sama tetapi dengan kunci 56-bit. Semakin panjang sebuah kunci, semakin besar *keyspace* yang harus dijalani untuk mencari kunci dengan cara *brute force attack* atau coba-coba karena *keyspace* yang harus dilihat merupakan pangkat dari bilangan 2. Jadi kunci 128-bit memiliki *keyspace* 2128, sedangkan kunci 56-bit memiliki *keyspace* 256. Artinya semakin lama kunci baru bisa diketahui. *Plaintext* adalah pesan atau informasi yang dikirimkan sedangkan *Ciphertext* adalah informasi yang sudah dienkripsi.

2.15. Pengujian Alpa

Pengujian alpha bertujuan untuk identifikasi dan menghilangkan sebanyak mungkin masalah sebelum akhirnya sampai ke user, dilakukan oleh orang-orang yang tidak terlibat dalam pengembangan dan memang ahli dibidangnya. Terdapat formulir resmi evaluasi.

1. Pengertian Pengujian

Pengujian adalah satu set aktifitas yang direncanakan dan sistematis untuk menguji atau mengevaluasi kebenaran yang diinginkan. Aktifitas pengujian terdiri dari satu set atau sekumpulan langkah dimana dapat menempatkan design kasus uji yang spesifik dan metode pengujian. Secara umum pola pengujian pada perangkat lunak adalah sebagai berikut[1] :

Pengujian dimulai dari level komponen hingga integrasi antar komponen menjadi sebuah sistem.

Teknik pengujian berbeda-beda sesuai dengan berbagai sisi atau unit uji dalam waktu yang berbeda-beda pula bergantung pada pengujian pada bagian mana yang dibutuhkan.

Pengujian dilakukan oleh pengembang perangkat lunak, dan jika untuk proyek besar, pengujian bisa dilakukan oleh tim uji yang tidak terkait dengan tim pengembang perangkat lunak (*independent Test Group* (ITG)).

Pengujian dan penirkutuan (*debugging*) merupakan aktifitas yang berbeda, tapi penirkutuan (*debugging*) harus diakomodasi pada berbagai strategi pengujian. Pengujian lebih fokus untuk mencari adanya kesalahan (*error*) baik dari sudut pandang orang secara umum atau dari sudut pandang pengembang tanpa harus

menemukan lokasi kesalahan pada kode program. Penirkutuan (*debugging*) adalah proses mencari lokasi kesalahan (*error*) pada kode program sehingga dapat segera diperbaiki oleh pembuat program (*programmer*).

Pengujian untuk validasi memiliki pendekatan sebagai berikut[1] :

1. *Black-Box Testing* (Pengujian kotak hitam)

Black-Box Testing yaitu menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dank ode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan.

Pengujian kotak hitam dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan. Kasus uji yang dibuat untuk melakukan pengujian kotak hitam harus dibuat dengan kasus benar dan kasus salah, misalkan untuk kasus proses login maka kasus uji yang dibuat adalah :

a. Jika *user* memasukan nama pemakai (*username*) dan kata sandi (*password*) yang benar.

b. Jika *user* memasukan nama pemakai (*username*) dan kata sandi (*password*) yang salah, misalnya nama pemakai benar tapi kata sandi salah, atau sebaliknya, atau keduanya salah.

2. *White-Box Testing (Pengujian kotak putih)*

White-Box Testing yaitu menguji perangkat lunak dari segi desain dan kode program apakah mampu menghasilkan fungsi-fungsi, masukan, dan keluaran yang sesuai dengan spesifikasi kebutuhan. Pengujian kotak putih dilakukan dengan

memerikas lojik dari kode program. Pembuatan kasus uji bisa mengikuti standar pengujian dari standar pemrograman yang seharusnya.

2.16. Pengujian Beta

Pengujian beta dilakukan pada satu atau lebih pengguna akhir dalam lingkungan yg sebenarnya, pengembang biasanya tidak ada pada pengujian ini. pengguna merekan semua masalah yg ditemui selama pengujian dan melaporkan pada pengembang pada interval waktu tertentu.

2.17. Referensi Jurnal

1) ***Champtek, : BR300***, (Online). (http://www.champtek.com/productdetail. aspx?cid= C00000090&pid=P00000080, di akses pada Oktober 20, 2012)

2) ***Eko Riswanto, 2007***, *Analisa dan Perancangan Sistem.* STMIK El Rahma, Yogyakarta. (online) (http://hendri83.wordpress.com/category/ analisa-dan-perancangan-sistem/. Di akses pada 15 Oktober 2012)

3) ***Herufal. 2010***. *Pengertian Teknologi.* (online). (http://blog.trisakti.ac.id/ herufal/2010/11/04/pengertian-teknologi/. Di akses pada 14 Oktober 2012)