

BAB II LANDASAN TEORI

2.1 Konsep Dasar Sistem

2.1.1 Pengertian Sistem

Sistem berasal dari bahasa Latin (*systēma*) dan bahasa Yunani (*sustēma*) adalah suatu kesatuan yang terdiri komponen atau elemen yang dihubungkan bersama untuk memudahkan aliran informasi, materi atau energi untuk mencapai suatu tujuan. Istilah ini sering dipergunakan untuk menggambarkan suatu set entitas yang berinteraksi, di mana suatu model matematika seringkali bisa dibuat. Sistem juga merupakan kesatuan bagian-bagian yang saling berhubungan yang berada dalam suatu wilayah serta memiliki item-item penggerak, contoh umum misalnya seperti negara. Negara merupakan suatu kumpulan dari beberapa elemen kesatuan lain seperti provinsi yang saling berhubungan sehingga membentuk suatu negara dimana yang berperan sebagai penggerakya yaitu rakyat yang berada dinegara tersebut.

Kata "sistem" banyak sekali digunakan dalam percakapan sehari-hari, dalam forum diskusi maupun dokumen ilmiah. Kata ini digunakan untuk banyak hal, dan pada banyak bidang pula, sehingga maknanya menjadi beragam. Dalam pengertian yang paling umum, sebuah sistem adalah sekumpulan benda yang memiliki hubungan di antara mereka.

(<http://id.wikipedia.org/wiki/Sistem>) Diakses Tanggal 17 April 2013

2.2 Konsep Dasar Informasi

2.2.1 Pengertian Informasi

Informasi adalah pesan (ucapan atau ekspresi) atau kumpulan pesan yang terdiri dari order sekuens dari simbol, atau makna yang dapat ditafsirkan dari pesan atau kumpulan pesan. Informasi dapat direkam atau ditransmisikan. Hal ini dapat dicatat sebagai tanda-tanda, atau sebagai sinyal berdasarkan gelombang. Informasi adalah jenis acara yang mempengaruhi suatu negara dari sistem dinamis. Para konsep memiliki banyak arti lain dalam konteks yang berbeda.

Informasi bisa di katakan sebagai pengetahuan yang didapatkan dari pembelajaran, pengalaman, atau instruksi. Namun demikian, istilah ini memiliki banyak arti bergantung pada konteksnya, dan secara umum berhubungan erat dengan konsep seperti arti, pengetahuan, negentropy, Persepsi, Stimulus, komunikasi, kebenaran, representasi, dan rangsangan mental.

Informasi adalah data yang telah diberi makna melalui konteks. Sebagai contoh, dokumen berbentuk spreadsheet (semisal dari Microsoft Excel) seringkali digunakan untuk membuat informasi dari data yang ada di dalamnya. Laporan laba rugi dan neraca merupakan bentuk informasi, sementara angka-angka di dalamnya merupakan data yang telah diberi konteks sehingga menjadi punya makna dan manfaat.

(<http://id.wikipedia.org/wiki/Informasi>) Diakses Tanggal 17 April 2013

2.3 Konsep Dasar Sistem Informasi

2.3.1 Pengertian Sistem Informasi

Sistem Informasi (SI) adalah kombinasi dari teknologi informasi dan aktivitas orang yang menggunakan teknologi itu untuk mendukung operasi dan manajemen. Dalam arti yang sangat luas, istilah sistem informasi yang sering digunakan merujuk kepada interaksi antara orang, proses algoritmik, data, dan teknologi. Dalam pengertian ini, istilah ini digunakan untuk merujuk tidak hanya pada penggunaan organisasi teknologi informasi dan komunikasi (TIK), tetapi juga untuk cara di mana orang berinteraksi dengan teknologi ini dalam mendukung proses bisnis.

Ada yang membuat perbedaan yang jelas antara sistem informasi, dan komputer sistem TIK, dan proses bisnis. Sistem informasi yang berbeda dari teknologi informasi dalam sistem informasi biasanya terlihat seperti memiliki komponen TIK. Hal ini terutama berkaitan dengan tujuan pemanfaatan teknologi informasi. Sistem informasi juga berbeda dari proses bisnis. Sistem informasi membantu untuk mengontrol kinerja proses bisnis.

Alter berpendapat untuk sistem informasi sebagai tipe khusus dari sistem kerja. Sistem kerja adalah suatu sistem di mana manusia dan/atau mesin melakukan pekerjaan dengan menggunakan sumber daya untuk memproduksi

produk tertentu dan/atau jasa bagi pelanggan. Sistem informasi adalah suatu sistem kerja yang kegiatannya ditujukan untuk pengolahan (menangkap, transmisi, menyimpan, mengambil, memanipulasi dan menampilkan) informasi.

Dengan demikian, sistem informasi antar-berhubungan dengan sistem data di satu sisi dan sistem aktivitas di sisi lain. Sistem informasi adalah suatu bentuk komunikasi sistem di mana data yang mewakili dan diproses sebagai bentuk dari memori sosial. Sistem informasi juga dapat dianggap sebagai bahasa semi formal yang mendukung manusia dalam pengambilan keputusan dan tindakan.

Sistem informasi merupakan fokus utama dari studi untuk disiplin sistem informasi dan organisasi informatika.

Sistem informasi adalah gabungan yang terorganisasi dari manusia, perangkat lunak, perangkat keras, jaringan komunikasi dan sumber data dalam mengumpulkan, mengubah, dan menyebarkan informasi dalam organisasi.

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.

(http://id.wikipedia.org/wiki/Sistem_informasi) Diakses Tanggal 17 April 2013

2.4 UML (*Unified Model Language*)

Unified Modelling Language (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa-bahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Seperti dikutip dari (Dennis et al 2009:29) defnisi dari UML adalah seperti berikut. "*The Objective of UML was to provide a common vocabulary of object-oriented term and diagramming techniques rich enough to model any systems development project from analysis through implementation.*" Maka dari

itu dalam mendesain system diperlukan komponen untuk menyusun seluruh bagian yang mendukungnya termasuk pengguna dan sub system.



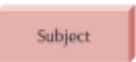

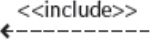
2.4.1 Use Case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya.

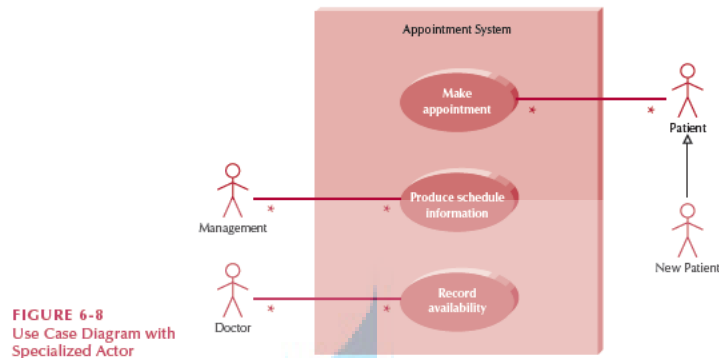
Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. *Use case diagram* dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem.

Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah *use case* juga dapat meng-*extend use case* lain. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain.

Tabel 2.1 Daftar Simbol Use Case Diagram

Nama	Simbol	Keterangan
Actor		Actor adalah orang atau system yang diturunkan dari subjek proses bisnis yang berjalan
Use Case		Use case merepresentasikan bagian fungsionalitas dari sistem
Subject Boundary		Merupakan Lingkup keseluruhan dari sistem proses yang berjalan
Association Relationship		Merupakan Penghubung antara actor dengan use case
Include Relationship		Merepresentasikan bahwa suatu use case merupakan bagian dari use case yang lain

Extend Relationship	\dashrightarrow \leftarrow	Merepresentasikan pendeskripsian dari use case umum ke use case yang lebih detail
Generalization Relationship	\leftarrow	Merepresentasikan beberapa use case menjadi use case tunggal







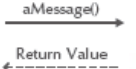
the Make Appointment use case as many times as they wish and that it is possible for the Make Appointment use case appointment to be executed by many different patients. In most cases, this type of many-to-many relationship is appropriate. However, it is possible to restrict the number of patients that can be associated with the Make Appointment use case. We will discuss the multiplicity issue in detail in the next chapter in regards to class diagrams.

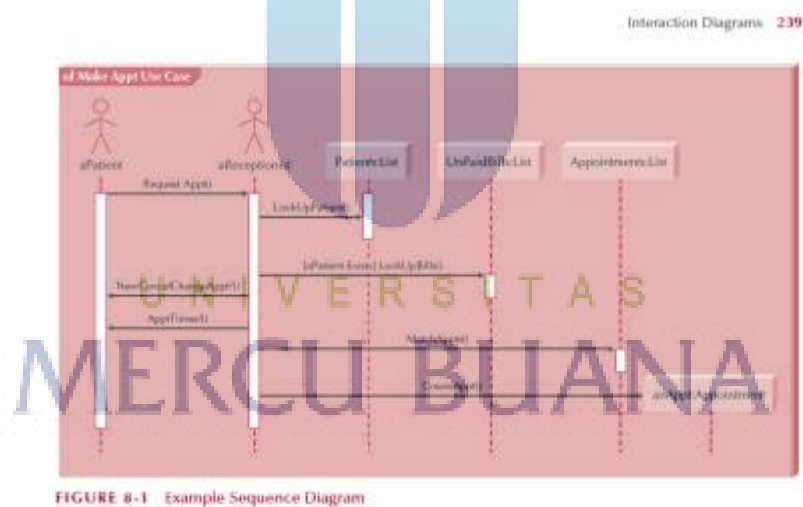
Gambar 2.1 Contoh Use Case

2.4.2 Sequence Diagram

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan. Masing-masing objek, termasuk aktor, memiliki *lifeline* vertikal.

Tabel 2.2 Daftar Simbol Sequence Diagram

Nama	Simbol	Keterangan
Actor		Actor adalah seorang atau objek sistem yang didefinisikan di dalam sistem
Object		Suatu objek melakukan pengiriman dan atau menerima message dari objek lain
Lifetime		Menyatakan bahwa suatu objek beraktivitas
Execution Occurrence		Menyatakan bahwa suatu objek beraktivitas selama proses sequence berlangsung
Message		Mengirimkan informasi dari satu objek ke objek lainnya



Gambar 2.5 Contoh Sequence Diagram

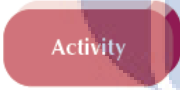







2.4.3 Activity Diagram


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

Activity diagram merupakan *state diagram* khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

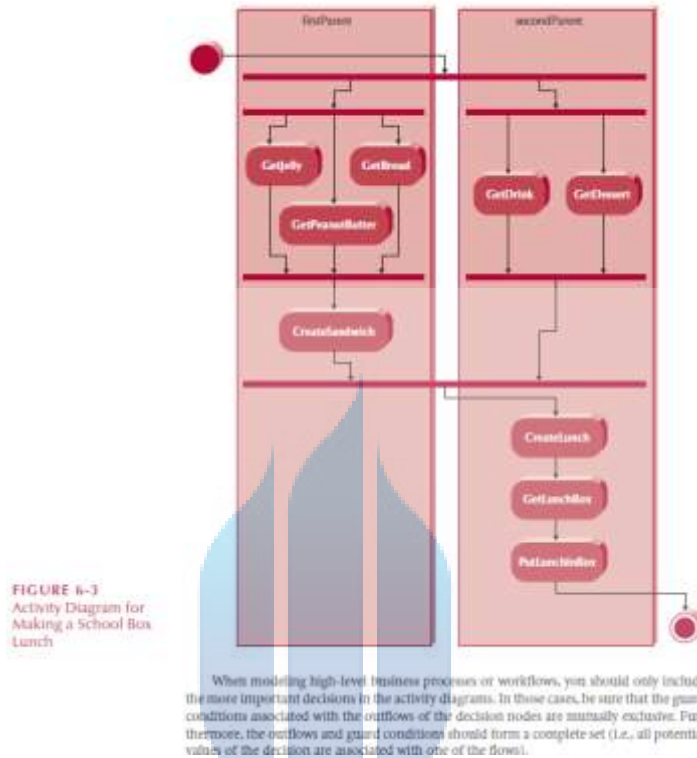
Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas. *Decision* digunakan untuk menggambarkan behaviour pada kondisi tertentu. Untuk mengilustrasikan proses-proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal. *Activity diagram* dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.

Tabel 2.3 Daftar Simbol Activity Diagram

Nama	Simbol	Keterangan
Activity		Merepresentasikan aktifitas bisnis dengan diwakili symbol activity berikut
Object Node		Merepresentasikan objek yang terhubung dengan konektor
Control Flow		Menunjukkan aliran proses bisnis yang sedang berjalan
Object Flow		Menunjukkan aliran objek dari satu aktifitas ke aktifitas yang lain dari objek class
Initial Node		Menunjukkan permulaan dari awal proses activity
Final – Activity Node		Menunjukkan akhir dari proses activity
Fork Node		Digunakan untuk membagi dari aktivitas umum menjadi aktivitas yang detail
Join Node		Digunakan untuk menyimpulkan dari beberapa aktivitas menjadi satu aktivitas

Swimlane		Digunakan untuk mengelompokkan aktivitas menjadi kolom dari setiap objek
----------	---	--

170 Chapter 6 Functional Modeling



Gambar 2.3 Contoh Activity Diagram

2.4.4 Class Diagram

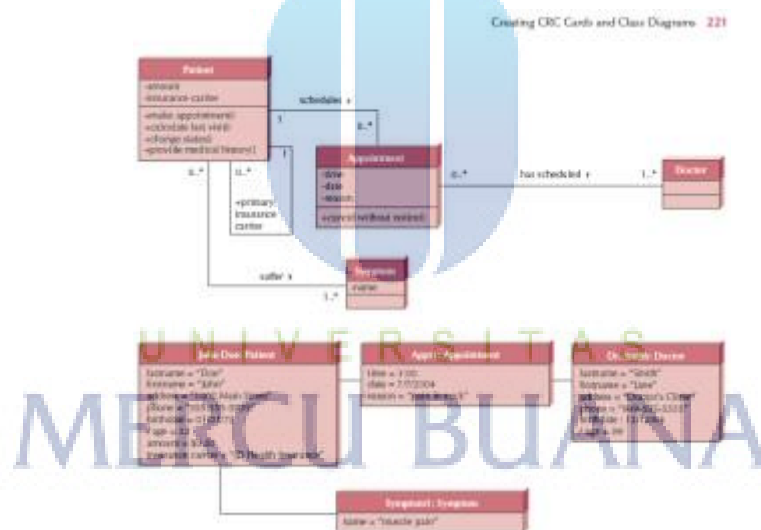
Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class diagram* menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

Class memiliki tiga area pokok : nama (stereotype), atribut, dan metoda. Atribut dan metoda dapat memiliki salah satu sifat berikut :

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

Tabel 2.4 Daftar Simbol Class Diagram

Nama	Simbol	Keterangan
Class	<pre> classDiagram class Class1 { -attribute1 +operation1() } </pre>	Merepresentasikan semua orang , tempat, atau sesuatu yang berkaitan dengan yang terjadi di dalam sistem dan menyimpan informasi berupa atribut dan operasi
Attribute	attribute name /derived attribute name	Merepresentasikan properti yang mendeskripsikan bagian dari objek class
Operation	operation name ()	Merepresentasikan aktivitas atau fungsi yang dilakukan oleh objek class
Association	1..* verb phrase 0..1	Merepresentasikan hubungan antara banyak class atau class dengan class tersebut



Gambar 2.4 Contoh Class Diagram

2.5 JAVA

Java adalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di Sun Microsystems saat ini merupakan bagian dari Oracle dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada bahasa pemrograman C dan bahasa pemrograman C++, namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras

bawah yang minimal. Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam (*bytecode*) dan dapat dijalankan pada berbagai Mesin Virtual Java (JVM). Java merupakan bahasa pemrograman yang bersifat umum/non-spesifik (*general purpose*), dan secara khusus didisain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa platform sistem operasi yang berbeda, java dikenal pula dengan slogannya, "*Tulis sekali, jalankan di mana pun*". Saat ini java merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi berbasis web.

(<http://id.wikipedia.org/wiki/Java>) Diakses Tanggal 17 April 2013

2.5.1 Versi Awal

Versi awal Java ditahun 1996 sudah merupakan versi release sehingga dinamakan Java Versi 1.0. Java versi ini menyertakan banyak paket standar awal yang terus dikembangkan pada versi selanjutnya:

- java.lang: Peruntukan kelas elemen-elemen dasar.
- java.io: Peruntukan kelas *input* dan *output*, termasuk penggunaan berkas.
- java.util: Peruntukan kelas pelengkap seperti kelas struktur data dan kelas kelas penanggalan.
- java.net: Peruntukan kelas TCP/IP, yang memungkinkan berkomunikasi dengan komputer lain menggunakan jaringan TCP/IP.
- java.awt: Kelas dasar untuk aplikasi antarmuka dengan pengguna (GUI)
- java.applet: Kelas dasar aplikasi antar muka untuk diterapkan pada penjelajah web.

2.5.2 Kelebihan

- **Multiplatform.** Kelebihan utama dari Java ialah dapat dijalankan di beberapa *platform* / sistem operasi komputer, sesuai dengan prinsip *tulis sekali, jalankan di mana saja*. Dengan kelebihan ini pemrogram cukup menulis sebuah program Java dan dikompilasi (diubah, dari bahasa yang dimengerti manusia menjadi bahasa mesin / *bytecode*) sekali lalu hasilnya

dapat dijalankan di atas beberapa platform tanpa perubahan. Kelebihan ini memungkinkan sebuah program berbasis java dikerjakan diatas operating system Linux tetapi dijalankan dengan baik di atas Microsoft Windows. Platform yang didukung sampai saat ini adalah Microsoft Windows, Linux, Mac OS dan Sun Solaris. Penyebabnya adalah setiap sistem operasi menggunakan programnya sendiri-sendiri (yang dapat diunduh dari situs Java) untuk meninterpretasikan *bytecode* tersebut.

- **OOP** (*Object Oriented Programming* - Pemrogram Berorientasi Objek)
- **Perpustakaan Kelas Yang Lengkap**, Java terkenal dengan kelengkapan *library*/perpustakaan (kumpulan program program yang disertakan dalam pemrograman java) yang sangat memudahkan dalam penggunaan oleh para pemrogram untuk membangun aplikasinya. Kelengkapan perpustakaan ini ditambah dengan keberadaan komunitas Java yang besar yang terus menerus membuat perpustakaan-perpustakaan baru untuk melingkupi seluruh kebutuhan pembangunan aplikasi.
- **Bergaya C++**, memiliki sintaks seperti bahasa pemrograman C++ sehingga menarik banyak pemrogram C++ untuk pindah ke Java. Saat ini pengguna Java sangat banyak, sebagian besar adalah pemrogram C++ yang pindah ke Java. Universitas-universitas di Amerika Serikat juga mulai berpindah dengan mengajarkan Java kepada murid-murid yang baru karena lebih mudah dipahami oleh murid dan dapat berguna juga bagi mereka yang bukan mengambil jurusan komputer.
- **Pengumpulan sampah** otomatis, memiliki fasilitas pengaturan penggunaan memori sehingga para pemrogram tidak perlu melakukan pengaturan memori secara langsung (seperti halnya dalam bahasa C++ yang dipakai secara luas).

2.5.3 Kekurangan

- **Tulis sekali, jalankan di mana saja** - Masih ada beberapa hal yang tidak kompatibel antara *platform* satu dengan *platform* lain. Untuk J2SE, misalnya *SWT-AWT bridge* yang sampai sekarang tidak berfungsi pada Mac OS X.

- **Mudah didekompilasi.** Dekompilasi adalah proses membalikkan dari kode jadi menjadi kode sumber. Ini dimungkinkan karena kode jadi Java merupakan *bytecode* yang menyimpan banyak atribut bahasa tingkat tinggi, seperti nama-nama kelas, metode, dan tipe data. Hal yang sama juga terjadi pada Microsoft .NET Platform. Dengan demikian, algoritma yang digunakan program akan lebih sulit disembunyikan dan mudah dibajak/direverse-engineer.
- **Penggunaan memori yang banyak.** Penggunaan memori untuk program berbasis Java jauh lebih besar daripada bahasa tingkat tinggi generasi sebelumnya seperti C/C++ dan Pascal (lebih spesifik lagi, Delphi dan Object Pascal). Biasanya ini bukan merupakan masalah bagi pihak yang menggunakan teknologi terbaru (karena trend memori terpasang makin murah), tetapi menjadi masalah bagi mereka yang masih harus berkatut dengan mesin komputer berumur lebih dari 4 tahun.

Contoh program Halo dunia yang ditulis menggunakan bahasa pemrograman Java adalah sebagai berikut:

```
// Outputs "Hello, world!" and then exits
public class HelloWorld {
    public static void main(String args[] ) {
        System.out.println("Hello, world!");
    }
}
```

Gambar 2.5 Contoh Kode Program Sederhana Dalam Java

2.6 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (bahasa Inggris: *database management system*) atau DBMS yang *multithread*, *multi-user*, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi GNU General Public License (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL. Tidak sama dengan proyek-proyek seperti Apache, dimana perangkat lunak dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh penulisnya masing-masing, MySQL dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia MySQL AB, dimana memegang hak cipta hampir atas semua kode sumbernya. Kedua orang Swedia dan satu orang Finlandia yang

mendirikan MySQL AB adalah: David Axmark, Allan Larsson, dan Michael "Monty" Widenius.

(<http://id.wikipedia.org/wiki/MySQL>) Diakses Tanggal 17 April 2013

2.6.1 Sistem manajemen basis data relasional

MySQL adalah sebuah implementasi dari sistem manajemen basisdata 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 basisdata yang telah ada sebelumnya; SQL (Structured Query Language). SQL adalah sebuah konsep pengoperasian basisdata, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Kehandalan suatu sistem basisdata (DBMS) dapat diketahui dari cara kerja pengoptimasi-nya dalam melakukan proses perintah-perintah SQL yang dibuat oleh pengguna maupun program-program aplikasi yang memanfaatkannya. Sebagai peladen basis data, MySQL mendukung operasi basisdata transaksional maupun operasi basisdata non-transaksional. Pada modus operasi non-transaksional, MySQL dapat dikatakan unggul dalam hal unjuk kerja dibandingkan perangkat lunak peladen basisdata kompetitor lainnya. Namun demikian pada modus non-transaksional tidak ada jaminan atas reliabilitas terhadap data yang tersimpan, karenanya modus non-transaksional hanya cocok untuk jenis aplikasi yang tidak membutuhkan reliabilitas data seperti aplikasi blogging berbasis web (wordpress), CMS, dan sejenisnya. Untuk kebutuhan sistem yang ditujukan untuk bisnis sangat disarankan untuk menggunakan modus basisdata transaksional, hanya saja sebagai konsekuensinya unjuk kerja MySQL pada modus transaksional tidak secepat unjuk kerja pada modus non-transaksional.



Gambar 2.6 Simbol MySQL

MySQL memiliki beberapa keistimewaan, antara lain :

1. **Portabilitas.** MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, Mac Os X Server, Solaris, Amiga, dan masih banyak lagi.
2. **Perangkat lunak sumber terbuka.** MySQL didistribusikan sebagai perangkat lunak sumber terbuka, dibawah lisensi GPL sehingga dapat digunakan secara gratis.
3. **Multi-user.** MySQL dapat digunakan oleh beberapa pengguna dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.
4. **'Performance tuning'**, MySQL memiliki kecepatan yang menakjubkan dalam menangani query sederhana, dengan kata lain dapat memproses lebih banyak SQL per satuan waktu.
5. **Ragam tipe data.** MySQL memiliki ragam tipe data yang sangat kaya, seperti signed / unsigned integer, float, double, char, text, date, timestamp, dan lain-lain.
6. **Perintah dan Fungsi.** MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah Select dan Where dalam perintah (*query*).
7. **Keamanan.** MySQL memiliki beberapa lapisan keamanan seperti level subnetmask, nama host, dan izin akses *user* dengan sistem perizinan yang mendetail serta sandi terenkripsi.

8. **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.
9. **Konektivitas.** MySQL dapat melakukan koneksi dengan klien menggunakan protokol TCP/IP, Unix socket (UNIX), atau Named Pipes (NT).
10. **Lokalisasi.** MySQL dapat mendeteksi pesan kesalahan pada klien dengan menggunakan lebih dari dua puluh bahasa. Meski pun demikian, bahasa Indonesia belum termasuk di dalamnya.
11. **Antar Muka.** MySQL memiliki antar muka (interface) terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi API (Application Programming Interface).
12. **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.
13. **Struktur tabel.** MySQL memiliki struktur tabel yang lebih fleksibel dalam menangani ALTER TABLE, dibandingkan basis data lainnya semacam PostgreSQL ataupun Oracle.

2.6.2 Administrasi MySQL

Untuk melakukan administrasi dalam basis data MySQL, dapat menggunakan modul yang sudah termasuk yaitu *command-line* (perintah: `mysql` dan `mysqladmin`). Juga dapat diunduh dari situs MySQL yaitu sebuah modul berbasis grafik (*GUI*): *MySQL Administrator* dan *MySQL Query Browser*. Selain itu terdapat juga sebuah perangkat lunak gratis untuk administrasi basis data MySQL berbasis web yang sangat populer yaitu `phpMyAdmin`. Untuk perangkat lunak untuk administrasi basis data MySQL yang dijual secara komersial antara lain: `MySQL front`, `Navicat` dan `EMS SQL Manager for MySQL`.

2.7 Aplikasi

Perangkat lunak aplikasi adalah suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna. Biasanya dibandingkan dengan perangkat lunak sistem yang mengintegrasikan berbagai kemampuan komputer, tapi tidak secara langsung menerapkan kemampuan tersebut untuk mengerjakan suatu tugas yang menguntungkan pengguna. Contoh utama perangkat lunak aplikasi adalah pengolah kata, lembar kerja, dan pemutar media.

Untuk membangun perangkat lunak aplikasi ini diperlukan perencanaan dengan melakukan perancangan berupa Rekayasa Perangkat Lunak, sehingga suatu aplikasi bisa memenuhi kebutuhan pengguna sistem yang dibangun tersebut. Selain itu aplikasi modern saat ini telah menggunakan konsep teknologi object oriented seperti dikutip dari (Miftakhul 1:2010) bahwa Teknologi Object Oriented memandang software sebagai sebuah interaksi antarbagian dalam sebuah sistem, dan menggambarkan satu bagian tersebut dalam satu objek (Visual Modelling Menggunakan UML dan Rational Rose, A Shendar & Hariman Gunadi) yang memiliki sifat/properti/data dan kemampuan untuk melakukan suatu tugas tertentu.

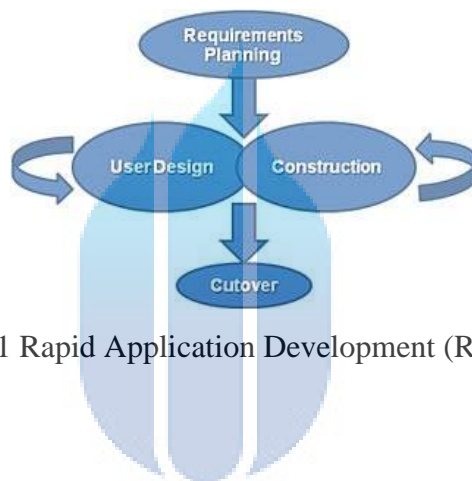
Beberapa aplikasi yang digabung bersama menjadi suatu paket kadang disebut sebagai suatu **paket** atau **suite aplikasi** (*application suite*). Contohnya adalah Microsoft Office dan OpenOffice.org, yang menggabungkan suatu aplikasi pengolah kata, lembar kerja, serta beberapa aplikasi lainnya. Aplikasi-aplikasi dalam suatu paket biasanya memiliki antarmuka pengguna yang memiliki kesamaan sehingga memudahkan pengguna untuk mempelajari dan menggunakan tiap aplikasi. Sering kali, mereka memiliki kemampuan untuk saling berinteraksi satu sama lain sehingga menguntungkan pengguna. Contohnya, suatu lembar kerja dapat dibenamkan dalam suatu dokumen pengolah kata walaupun dibuat pada aplikasi lembar kerja yang terpisah.

(<http://id.wikipedia.org/wiki/Aplikasi>) diakses 18 April 2013

2.8 Pengembangan Software dengan Metodologi RAD

Rapid application development (RAD) adalah metodologi pengembangan perangkat lunak yang menggunakan perencanaan minimal dalam mendukung prototyping cepat. Perencanaan dari software yang dikembangkan menggunakan RAD disisipkan dengan menulis perangkat lunak itu sendiri. Kurangnya ekstensif pra-perencanaan umumnya memungkinkan perangkat lunak yang akan ditulis jauh lebih cepat, dan membuat lebih mudah untuk mengubah requirements. RAD tidak sesuai jika risiko teknis yang tinggi.

(http://en.wikipedia.org/wiki/Rapid_application_development) diakses tanggal 7 juni 2013.



Gambar 2.1 Rapid Application Development (RAD) Model

Empat fase RAD

1. **Requirements Planning phase** : memadukan unsur-unsur sistem perencanaan dan sistem fase analisis Development Life Cycle Sistem (SDLC). Pengguna, manajer, dan anggota staf TI membahas dan menyepakati kebutuhan bisnis, ruang lingkup proyek, kendala, dan persyaratan sistem. Itu berakhir ketika tim setuju pada isu-isu kunci dan memperoleh otorisasi manajemen untuk melanjutkan.
2. **User design phase** : selama fase ini, pengguna berinteraksi dengan sistem analis dan mengembangkan model dan prototipe yang mewakili semua proses sistem, input, dan output. Kelompok RAD atau subkelompok biasanya menggunakan kombinasi Joint Application Development (JAD) teknik dan alat CASE untuk menerjemahkan kebutuhan pengguna ke dalam model kerja. Desain Pengguna adalah proses interaktif yang berkesinambungan yang memungkinkan pengguna untuk memahami,

memodifikasi, dan akhirnya menyetujui model kerja dari sistem yang memenuhi kebutuhan mereka.

3. **Construction phase** : berfokus pada program dan pengembangan aplikasi tugas yang sama dengan SDLC. Dalam RAD, bagaimanapun, pengguna terus berpartisipasi dan masih dapat menyarankan perubahan atau perbaikan sebagai layar yang sebenarnya atau laporan dikembangkan. Tugasnya adalah pemrograman dan pengembangan aplikasi, coding, unit integrasi dan pengujian sistem.
4. **Cutover phase** : menyerupai tugas akhir dalam tahap implementasi SDLC, termasuk konversi data, pengujian, pergantian ke sistem baru, dan pelatihan pengguna. Dibandingkan dengan metode tradisional, seluruh proses dikompresi. Akibatnya, sistem baru dibangun, disampaikan, dan ditempatkan dalam operasi lebih cepat.

2.9 Pengertian Data

Data adalah deskripsi dari sesuatu dan kejadian yang kita hadapi(*the descriptions of things and events of that we face*). Sementara data bisnis(business data) didefinisikan sebagai deskripsi organisasi tentang ssuatu (Iresources) dan kejadian (transactions) yang terjadi (business data is an organization's description of things (resources) and events (transactions)that it face).

(Al-Bahra 8:2013)

2.10 Rekayasa Perangkat Lunak

Pemodelan dalam suatu rekayasa perangkat lunak merupakan suatu hal yang dilakukan di tahapan awal. Di dalam suatu rekayasa perangkat lunak sebenarnya masih memungkinkan tanpa melakukan suatu pemodelan . hal itu tidak lagi dapat dilakukan dalam suatu industry perangkat lunak . pemodelan dalam perangkat lunak merupakan suatu yang harus dikerjakan di bagian awal dari rekayasa, dan pemodelan ini akan mepengaruhi pekerjaan-pekerjaan dalam rekayasa perangkat lunak tersebut.

“rekayasa perangkat lunak adalah sebuah disiplin yang mengintegrasikan proses, metode, dan alat-alat bantu bagi perkembangan proses perangkat lunak komputer.” (Yasin 15:2012)

2.11 Pengertian Database

Dengan Mengutip dari (Indrajani 3:2011) yang menyebutkan beberapa pengertian tentang basis data, yaitu:

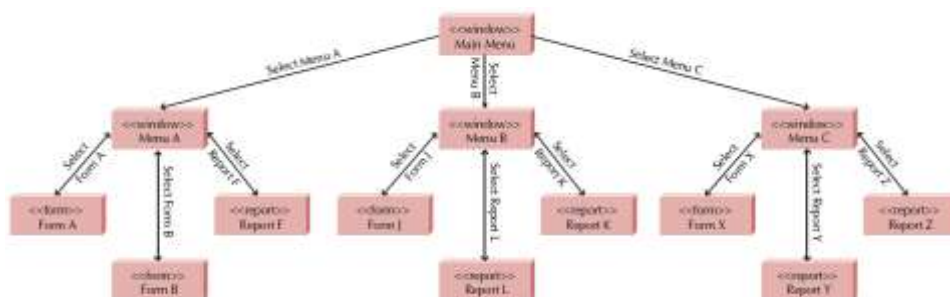
- Kumpulan terpadu dari elemen data logis yang saling berhubungan. Basis data mengonsolidasi banyak catatan yang sebelumnya dalam file terpisah.
- Merupakan suatu kumpulan data yang berhubungan secara logis dan deskripsi data tersebut, yang dirancang untuk memenuhi informasi yang dibutuhkan oleh suatu organisasi.

2.12 Navigation Diagram

Navigation Diagram merupakan pemodelan konsep untuk menggambarkan seluruh fungsional dari aplikasi sehingga pengguna sistem mampu memahami terlebih dahulu tanpa harus menggunakan aplikasi sebenarnya terlebih dahulu.

Seperti dikutip dari (Dennis et al 2009:423) seperti berikut:

“The interface structure defines the basic components of the interface and how they work together to provide functionality to users. A *window navigation diagram (WND)* is used to show how all the screens, forms, and reports used by the system are related and how the user moves from one to another.”



Dennis, Alan et al. System Analysis and Design with UML Version 2.0 An Object-Oriented Approach 3rd ed: John Wiley & Sons, Inc., 2005.

Gambar 2.8 Windows Navigation Diagram