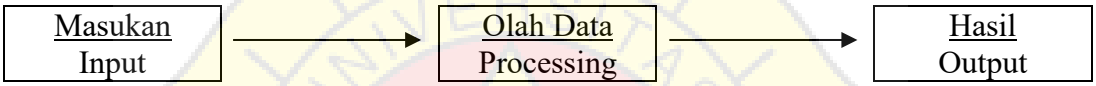
B A B I I

L A N D A S A N T E O R I

2.1.SISTEM INFORMASI

Sistem adalah kumpulan dari elemen-elemen yang berinteraksi untuk mencapai tujuan tertentu (Hartono, 1999). Model umum sebuah sistem terdiri dari masukan, pengolah dan keluaran. Berikut gambaran umum sebuah sistem:



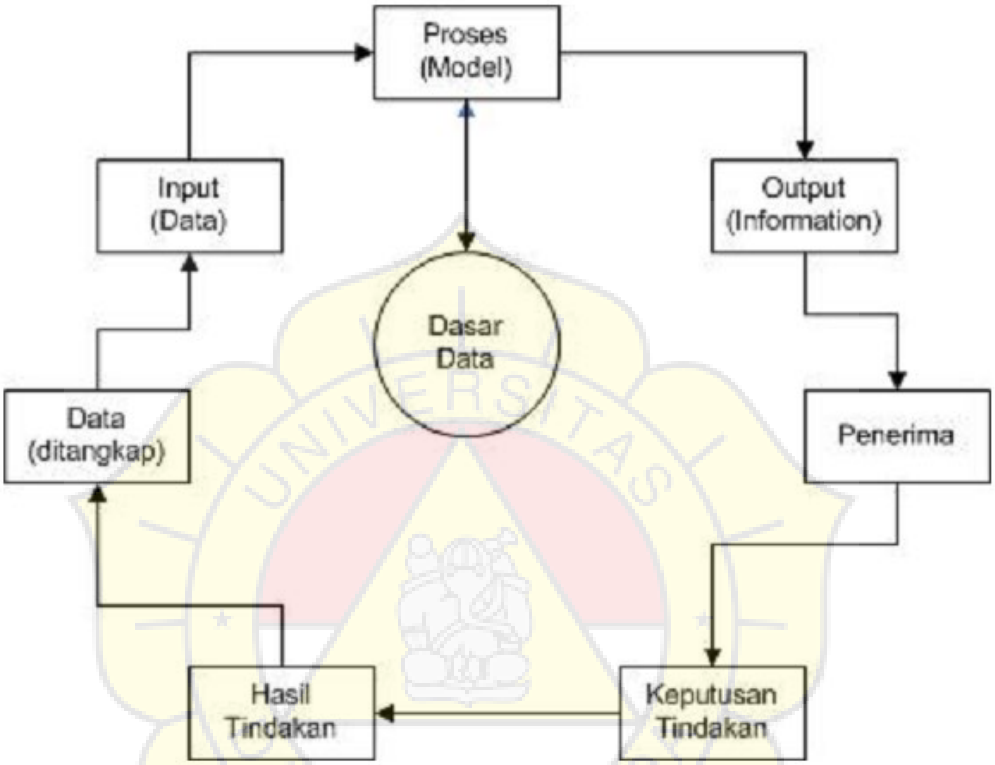
Gambar 2.1 Bentuk Sistem

Sistem bisa terdiri dari beberapa komponen. Komponen-komponen dalam suatu sistem tidak dapat berdiri sendiri. Mereka saling berinteraksi dan bekerja sama membentuk suatu kesatuan untuk mencapai tujuan tertentu.

Informasi adalah data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti bagi yang menerimanya (Hartono, 1999). Sumber informasi adalah data. Data adalah kenyataan yang menggambarkan suatu kejadian-kejadian dan kesatuan nyata (Hartono, 1999).

Data diolah melalui suatu model menjadi informasi, penerima kemudian menerima informasi tersebut, membuat suatu keputusan dan melakukan tindakan, yang berarti menghasilkan suatu tindakan lain yang akan membuat sejumlah data kembali. Data tersebut akan ditangkap sebagai input, diproses kembali lewat suatu model dan seterusnya membentuk suatu siklus informasi (Hartono, 1999).

Siklus ini oleh John Burch disebut sebagai siklus informasi (*information cycle*). Berikut gambar siklus informasi:



Gambar 2.2 Siklus Informasi

Sedangkan sistem informasi sendiri adalah suatu cara yang sudah tertentu untuk menyediakan informasi yang dibutuhkan oleh organisasi untuk beroperasi dengan cara yang sukses dan untuk organisasi bisnis dengan cara yang menguntungkan (Sabarguna, 2005). Peran sistem informasi adalah menghasilkan informasi dari data yang diproses oleh sistem informasi.

Komponen yang terkait dengan sistem informasi adalah (Sabarguna, 2005):

a. Pemakai
b. Tujuan
c. Masukan – proses – keluaran
d. Data
e. Teknologi
f. Model
g. Pengendali

Semua komponen tersebut saling berkait, bila data salah, maka hasilnya akan merupakan informasi yang salah juga.

2.2.REKAM MEDIK

Rekam medik merupakan hasil aktivitas pencatatan pada suatu rumah sakit atau suatu institusi pelayanan kesehatan yang berupa data. Data tersebut meliputi data sosial maupun data medik pasien rawat jalan dan rawat inap dan diproses oleh seorang tenaga rekam medik ataupun paramedik sehingga menjadi informasi yang berguna bagi rumah sakit. Adapun pengertian rekam medik adalah himpunan fakta-fakta yang berhubungan dengan riwayat hidup dan kesehatan tentang seorang pasien tersebut yang ditulis oleh professional di bidang kesehatan (Huffman, 1994).

Menurut petunjuk teknis penyelenggaraan rekam medik di rumah sakit Depkes RI Dirjen Yanmed tahun 1991, tujuan terlaksananya rekam medik adalah untuk menunjang tertib administrasi dalam rangka upaya peningkatan pelayanan kesehatan rumah sakit.

Menurut Huffman (1994) menyatakan bahwa kegunaan rekam medik adalah sebagai berikut:

a. Manajemen pelayanan pasien
b. Tinjauan kualitas (*Quality Review*)
c. Pengurusan klaim asuransi (*Financial reimbursement*)
d. Perkara hukum (*Legal affairs*)
e. Pendidikan (*Education*)
f. Penelitian (*Research*)
g. Kesehatan umum (*Public health*)
h. Perencanaan dan pemasaran (*Planning and marketing*)

Dalam sistem rekam medik ini, terdapat keterkaitan antara komponen-komponen di dalam sistem. Rekam medik dibuat karena adanya kebutuhan pengguna untuk memanfaatkan suatu produk tentang basis data dari suatu kendala yang dapat memberikan kesimpulan tanpa kesulitan mencari-cari data yang dibutuhkan dalam jangka waktu yang relevan. Dengan adanya rekam medik, dapat memberikan solusi mudahnya menata data yang banyak tanpa harus menghabiskan waktu yang banyak dalam pencarian suatu data. Data rekam medik sendiri didapatkan dari riwayat hidup seorang pasien yang menjalani begitu banyak rangkaian pengobatan yang dilakukannya seperti obat-obatan yang dikonsumsi, terapis yang dijalani, dan konsultasi dengan dokter spesialisnya guna mendapatkan informasi dari pasien yang ingin memulihkan kondisinya. Pengguna dapat memanfaatkan informasi yang didapatkan dari basis data rekam

medik ini untuk menyimpulkan apa saja yang dibolehkan dan tidak boleh dijalani oleh pasien yang akan menjalani pemulihan dirinya. Dengan adanya program rekam medik ini diharapkan pengguna dapat dimudahkan dalam mengelola data pasiennya.

2.3.SIKLUS HIDUP PENGEMBANGAN SISTEM

Siklus hidup pengembangan sistem merupakan suatu bentuk yang digunakan untuk menggambarkan tahapan utama dan langkah-langkah didalam tahapan tersebut dalam proses pengembangannya. Proses pengembangan sistem melewati beberapa tahapan dari mulai sistem itu direncanakan sampai dengan sistem tersebut diterapkan, dioperasikan, dan dipelihara (Hartono, 1999). Proses pengembangan sistem yang paling utama adalah analisis sistem, desain sistem, dan implementasi sistem.

2.4.ANALISIS SISTEM

ANALISIS sistem (*system analysis*) dapat didefinisikan sebagai penguraian dari suatu sistem informasi yang utuh kedalam bagian-bagian komponennya dengan maksud untuk mengidentifikasikan dan mengevaluasi permasalahan-permasalahan, kesempatan-kesempatan, hambatan-hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan-perbaikan (Hartono, 1999).

Tujuan utama analisis sistem adalah untuk menentukan hal-hal detail tentang yang akan dikerjakan oleh sistem yang diusulkan. Analisis sistem lebih

menekankan pada isu-isu bisnis (kebutuhan pihak pemakai), bukan masalah teknis atau implementasi. Tahap analisis merupakan tahap yang kritis dan sangat penting, karena kesalahan didalam tahap ini akan menyebabkan juga kesalahan di tahap selanjutnya.

Dalam tahap analisis sistem, terdapat langkah-langkah dasar yang harus dilakukan oleh analis sistem, yaitu (Hartono, 1999):

1. *Identify*, yaitu pengidentifikasian masalah
2. *Understand*, yaitu memahami cara kerja dari sistem yang sudah ada
3. *Analyze*, yaitu menganalisis sistem yang sudah ada
4. *Report*, yaitu membuat laporan dari hasil analisis

2.5.DESAIN SISTEM

Setelah tahap analisis sistem, tahap selanjutnya adalah desain sistem (*systems design*) yang merupakan tahapan untuk memikirkan bagaimana membentuk sebuah sistem. Desain sistem dapat didefinisikan sebagai penggambaran, perencanaan dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah kedalam satu kesatuan yang utuh dan berfungsi (Hartono, 1999).

Desain sistem dapat dibagi dalam dua bagian, yaitu desain sistem secara umum (*general systems design*) dan desain sistem secara terinci (*detailed systems design*).

Desain sistem secara umum disebut juga dengan desain konseptual (*conceptual design*) bertujuan untuk memberikan gambaran secara umum kepada

user tentang sistem yang baru. Desain sistem secara umum merupakan persiapan dari desain terinci. Desain secara umum mengidentifikasikan komponen-komponen sistem informasi yang akan didesain secara rinci (Hartono, 1999).

Desain sistem yang terinci bertujuan untuk memberikan gambaran yang jelas dan rancang bangun yang lengkap kepada pemrogram komputer dan ahli-ahli teknik lainnya (Hartono, 1999).

2.6.IMPLEMENTASI SISTEM

Tahap implementasi sistem (*system implementation*) merupakan tahap meletakkan sistem supaya siap untuk dioperasikan. Tahap ini termasuk juga kegiatan menulis kode program jika tidak digunakan paket perangkat lunak aplikasi (Hartono, 1999).

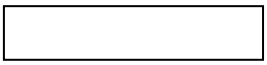
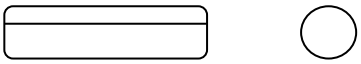
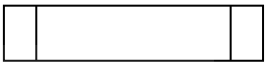
Data implementasi sistem untuk tugas akhir ini sebagian datanya diperoleh dari data klinik xxx, mulai dari proses pencatatan data diri pasien hingga riwayat hidup pasien dalam pengobatannya selama ini dan beberapa data pendukung lainnya.

2.7.DIAGRAM ALIR DATA

Diagram Alir Data (DAD) sering digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika tanpa mempertimbangkan lingkungan fisik dimana data tersebut mengalir atau lingkungan fisik dimana data tersebut akan disimpan (Hartono, 1999).

Daftar notasi simbol DAD ditunjukkan pada tabel 2.1 berikut:

Tabel 2.1. Daftar notasi simbol DAD

Komponen DAD	Symbol
Kesatuan luar (external entity) atau batas system (boundary)	
Arus data atau aliran data (data flow)	
Proses (process)	
Simpanan data (data store)	

2.8.UML (Unified Modelling Language)

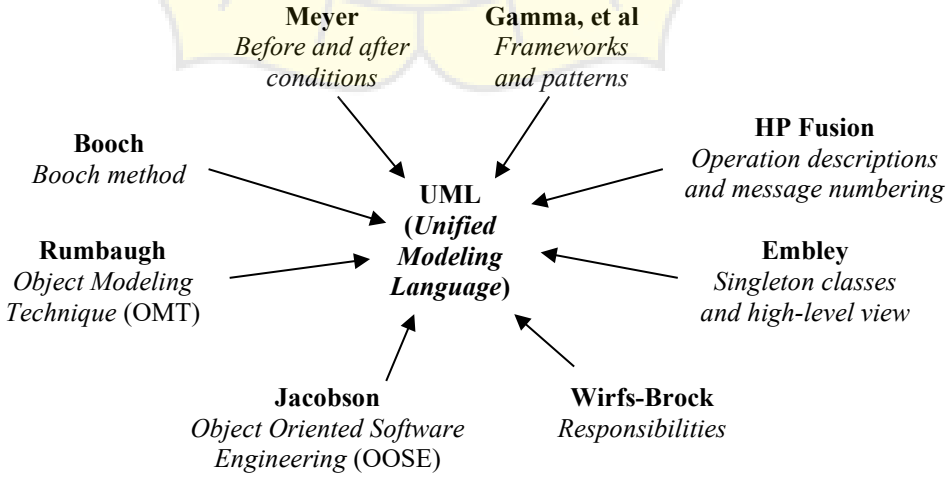
UML adalah salah satu alat bantu yang sangat handal di dunia pengembangan system yang berorientasi obyek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembang system untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka dengan yang lain (Munawar, 2005).

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

Pemodelan OMT yang dikembangkan oleh Rumbaugh didasarkan pada analisis terstruktur dan pemodelan *entity-relationship*. Tahapan utama dalam metodologi ini adalah analisis, desain sistem, desain objek dan implementasi. Keunggulan metode ini adalah dalam penotasian yang mendukung semua konsep *object oriented*.

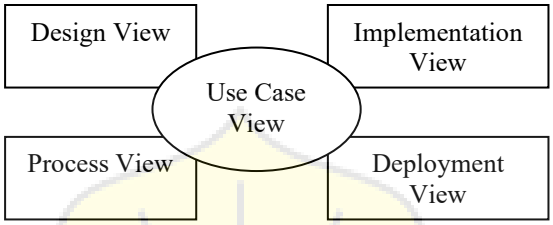
Metode OOSE dari Jacobson lebih memberi penekanan pada *use case*. OOSE memiliki tiga tahapan yaitu membuat model requirement dan analisis, desain dan implementasi, dan model pengujian (*test model*). Keunggulan metode ini adalah mudah dipelajari karena memiliki notasi yang sederhana namun mencakup seluruh tahapan dalam rekayasa perangkat lunak.

Dengan UML, metode Booch, OMT dan OOSE digabungkan dengan membuang elemen-elemen yang tidak praktis ditambah dengan elemen-elemen dari metode lain yang lebih efektif dan elemen-elemen baru yang belum ada pada metode terdahulu sehingga UML lebih ekspresif dan seragam daripada metode lainnya. Berikut unsur-unsur yang membentuk UML.



Gambar 2.3. unsur-unsur pembentuk UML

UML dibangun atas model 4+1 *view*. Model ini didasarkan pada fakta bahwa struktur sebuah sistem dideskripsikan dalam 5 *view* dimana salah satu diantaranya *use case view*. *Use case view* ini memegang peran khusus untuk mengintegrasikan *content* ke *view* yang lain (Munawar, 2005).



Gambar 2.4. model 4+1 *view*

Kelima *view* tersebut tidak berhubungan dengan diagram yang dideskripsikan di UML. Setiap view berhubungan dengan perspektif tertentu dimana sistem akan diuji. *View* yang berbeda akan menekankan pada aspek yang berbeda dari sistem yang mewakili ketertarikan sekelompok *stakeholder* tertentu.

Use case view mendefinisikan perilaku eksternal sistem. Hal ini menjadi daya tarik bagi pengguna, analis dan *tester*. Pandangan ini mendefinisikan kebutuhan system karena mengandung semua view lain yang mendeskripsikan aspek-aspek tertentu dari rancangan sistem. Itulah sebabnya *use case view* menjadi pusat peran dan sering dikatakan yang mengendalikan proses pengembangan perangkat lunak.

Design view mendeskripsikan struktur logika yang mendukung fungsi-fungsi yang dibutuhkan di *use case*. *Design view* ini berisi definisi komponen program, bagian utama bersama dengan spesifikasi data, perilaku dan

interaksinya. Informasi yang terkandung di view ini menjadi perhatian para programmer karena menjelaskan secara detil bagaimana fungsionalitas system akan diimplementasikan.

Implementation view menjelaskan komponen-komponen fisik dari sistem yang akan dibangun. Hal ini berbeda dengan komponen *logic* yang dideskripsikan pada *design view*. Termasuk diantaranya file exe, library dan database. Informasi yang ada di *view* ini relevan dengan aktifitas-aktifitas seperti manajemen konfigurasi dan integrasi sistem.

Process view berhubungan dengan hal-hal yang berkaitan dengan *concurrency* di dalam sistem. Sedangkan *deployment view* menjelaskan bagaimana komponen-komponen fisik didistribusikan ke lingkungan fisik seperti jaringan komputer di mana sistem akan dijalankan. Kedua view ini menunjukkan kebutuhan non fungsional dari system seperti toleransi kesalahan dan hal-hal yang berhubungan dengan kinerja.

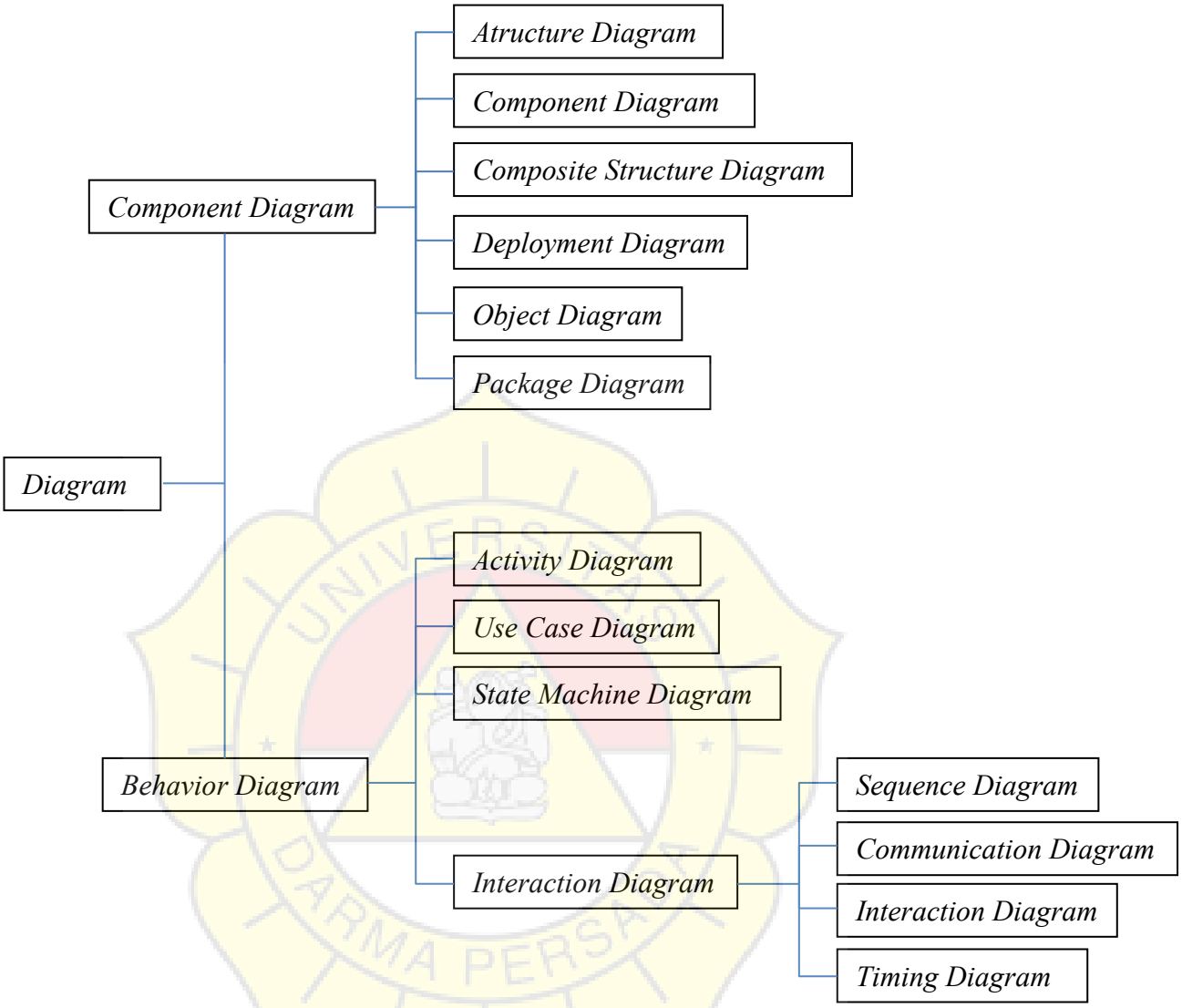
Tabel 2.2. tipe diagram UML

Diagram	Tujuan	Keterangan
Activity	Perilaku proseduran & parallel	Sudah ada di UML 1
Class	Class, fitur dan relasinya	Sudah ada di UML 1
Communication	Interaksi diantara obyek. Lebih menekankan ke link	Di UML 1 disebut collaboration
Component	Struktur dan koneksi dari komponen	Sudah ada di UML 1
Composite structure	Dekomposisi sebuah class saat runtime	Baru untuk UML 2
Deployment	Penyebaran/instalasi ke klien	Sudah ada di UML 1

Diagram	Tujuan	Keterangan
Interaction overview	Gabungan antara activity & sequence diagram	Baru untuk UML 2
Object	Contoh konfigurasi instance	Tidak resmi ada di UML 1
Sequence	Interaksi antar obyek. Lebih menekankan pada urutan	Sudah ada di UML 1
State machine	Bagaimana event mengubah sebuah obyek	Sudah ada di UML 1
Timing	Interaksi antar obyek. Lebih menekankan pada waktu	Baru untuk UML 2
Use case	Bagaimana user berinteraksi dengan sebuah system	Sudah ada di UML 1

Pengembangan sistem adalah aktifitas manusia. Tanpa adanya kemudahan untuk memahami sistem notasi, proses pengembangan kemungkinan besar akan mengalami kesalahan. UML adalah sistem notasi yang sudah dibakukan di dunia pengembangan sistem, hasil kerja bersama dari Grady Booch, James Rumbaugh dan Ivar Jacobson (Munawar, 2005).

UML yang terdiri dari serangkaian diagram memungkinkan bagi sistem analis untuk membuat cetak biru sistem yang komprehensif kepada klien, programmer dan tiap orang yang terlibat dalam proses pengembangan tersebut, karena setiap diagram bisa mewakili *stakeholder* yang berbeda di sistem tersebut. Dengan UML akan bisa menceritakan apa yang seharusnya dilakukan oleh sebuah sistem bukan bagaimana yang seharusnya dilakukan oleh sebuah sistem.



Gambar 2.5 klasifikasi diagram UML versi 2.0

2.8.1. *Use Case*

Use case adalah abstraksi dari interaksi antara sistem dan aktor. Oleh karena itu sangat penting untuk memilih abstraksi yang cocok. Sebagai contoh, saat pasien menelpon klinik untuk menanyakan status dokter, pasien akan berbicara kepada karyawan klinik yang akan mencatat status

tersebut ke sistem. Untuk melakukan hal tersebut, karyawan akan menjalankan peran sebagai resepsionis meskipun pekerjaan tersebut mungkin bukan pekerjaan formalnya. Pada situasi ini, karyawan merupakan *instance* dari aktor resepsionis dan interaksi diantara karyawan dengan sistem adalah *instance* dari *use case*.

Use case adalah konstruksi untuk mendeskripsikan bagaimana sistem akan terlihat di mata pengguna potensial. *Use case* terdiri dari sekumpulan scenario yang dilakukan oleh seorang aktor (orang, perangkat keras, urutan waktu atau sistem yang lain). Sedangkan *use case* diagram memfasilitasi komunikasi di antara analis dan pengguna serta di antara analis dan klien.

2.8.2. *Activity Diagram*

Activity diagram seperti sebuah *flow chart*. *Activity diagram* menunjukkan tahapan, pengambilan keputusan dan percabangan. Diagram ini sangat berguna untuk menunjukkan operation sebuah obyek dan proses bisnis. Kelebihan *activity diagram* dibanding *flow chart* adalah kemampuannya dalam menampilkan aktivitas parallel.

2.8.3. *Package Diagram*

Package adalah pengelompokan konstruksi ke level yang lebih tinggi. Sebuah *package* bisa menjadi anggota *package* yang lain. Bila dibuat hierarki, *package* yang paling tinggi akan mengandung *package-package* yang lain. Demikian seterusnya hingga yang paling bawah adalah *class*.

Manfaat utama penggunaan *package* adalah penerapannya pada sistem skala besar untuk mendapatkan gambaran saling ketergantungan diantara komponen-komponen utama pada sistem.

2.8.4. *Traceability Diagram*

Diagram *traceability* digunakan untuk merunut realisasi dari *use-case*.

2.8.5. *Statechart Diagram*

Jika pada pemodelan interaksi menyiapkan detil spesifikasi dari *use case*, pada *statechart* akan diberikan detil deskripsi dari *class* yaitu perubahan *state* dari *class* menjadi lebih tepat. Perubahan dinamis inilah yang akan menjadi perilaku dari suatu obyek. Biasanya *statechart* ini memodelkan aturan main suatu proses bisnis.

2.8.6. *Collaboration Diagram*

Collaboration diagram adalah bentuk lain *sequence diagram*. Bila *sequence diagram* diorganisir menurut waktu maka *collaboration diagram* diorganisir menurut ruang/space.

Collaboration diagram merupakan asosiasi diantara obyek-obyek. Panah di dekat garis asosiasi menunjukkan *message*, sedangkan *content message* ditunjukkan dengan label. Angka pada *message* menunjukkan urutan *message*.

Dengan *collaboration diagram* memungkinkan untuk memodelkan pengiriman sebuah *message* ke banyak obyek pada class yang sama. Demikian juga halnya untuk menunjukkan adanya obyek aktif yang mengendalikan aliran dari *message*.

2.8.7. Skema Penerjemahan UML ke Pemrograman Gambas

a. Entitas-Entitas Struktur

Entitas-entitas struktur dibagi menjadi dua sesi, yaitu:

1. Sesi Pertama membahas elemen-elemen yang tidak mempunyai pemetaan di bahasa. Entitas-entitas ini disebut entitas-entitas tidak bergantung bahasa.
2. Sesi kedua adalah elemen-elemen yang mempunyai pemetaan sederhana ke bahasa pemrograman, disebut entitas-entitas bergantung bahasa.

b. Entitas-Entitas Independen Gambas

1. *Actor*

Actor merepresentasikan peran dari pemakai. *Actor* merupakan eksternal terhadap sistem yang dikembangkan. *Actor* tidak harus orang. *Actor* dalam sistem ini, terdiri dari: pasien, dokter, staf, perawat, petugas administrasi, dan petugas loket pendaftaran.

2. *Use Case*

   Use case merepresentasikan sekuen dari aksi-aksi yang dijamin sistem perangkat lunak untuk dilakukan ke *Actor*. Satu yang tetap adalah *use case* seharusnya menyediakan hasil dari nilai yang diobservasi ke *Actor*. Diagram *use case* yang digunakan antara lain: mendaftar, mengisi formulir pendaftaran, mengambil kartu rawat, menjalani perawatan, mendapatkan hasil akhir perawatan berupa rekam medik.

3. Kolaborasi

   Kolaborasi sering digunakan untuk memberikan struktur ke model perancangan. Umumnya, kolaborasi akan mempunyai pemetaan satu ke satu terhadap *use case*. Diagram *use case collaboration* yaitu: *login*, melihat daftar formulir, mencatat formulir pendaftaran, merekam data medik pasien, mencatat hasil pemeriksaan dokter.

4. Obyek

   Obyek adalah instant dari satu kelas. Obyek direpresentasikan dengan satu persegi.

c. Entitas-Entitas Bergantung Gambas

   1. Kelas

      Kelas adalah cetak biru untuk obyek

2. Paket

   Paket merupakan mekanisme pengelompokan secara umum. Paket dapat berisi tipe elemen lain. Dalam sistem ini terdapat beberapa paket, yaitu: manajemen klinik, loket pendaftaran, loket administrasi, bagian rekam medik.

3. *Interface*

   Interface adalah kumpulan operasi yang menspesifikasikan layanan dari kelas. *Interface* diterjemahkan secara langsung menjadi *interface* di Gambas.

4. Hubungan

- *Dependency*, terdapat pada diagram *use case* yang menghubungkan sistem pendaftaran, sistem untuk administrasi, sistem untuk petugas rekam medik, dan pengguna *management and system maintenance*.
- *Association*, hubungan asosiasi menghubungkan *business use-case* yang satu dengan yang lain, salah satu contohnya adalah hubungan asosiasi *include* antara *business use-case* mengambil data pasien dengan *business use-case* mendaftar.

2.9.SISTEM OPERASI

Linux adalah nama sebuah sistem operasi (*operating system*) untuk PC yang bekerja secara *multitasking* dan *multiuser*. Linux bekerja secara *multitasking* artinya dapat menjalankan beberapa perangkat lunak secara bersamaan, misalnya

dapat bermain *game* sambil melakukan *download* dari internet. Linux bekerja secara multiuser artinya Linux mendukung penggunaan perangkat lunak atau komputer untuk melayani beberapa user sekaligus, misalnya sebuah program dapat digunakan bersama-sama pada jaringan (*network*) (Nova Novriansyah, 2001).

Linux sebenarnya adalah tiruan (*clone*) dari UNIX yang dirancang untuk dijalankan pada PC. Berbicara tentang UNIX, pada dasarnya sistem operasi ini bersifat portabel (tidak tergantung pada perangkat keras tertentu) sehingga dapat digunakan mulai dari platform notebook hingga super-computer. Demikian pula halnya dengan Linux, sistem operasi ini sudah sangat popular dan banyak diminati para professional (Nova Novriansyah, 2001).

Linux awalnya merupakan proyek hobi yang dibuat oleh Linus Torvald, seorang mahasiswa Universitas Helsinki di Finlandia. Linus Torvald terinspirasi oleh diciptakannya **minix** – sistem tiruan UNIX sederhana yang dibuat oleh Andy Tanenbaum. Oleh karena itu ia pun termotivasi untuk membuat sistem operasi tiruan UNIX yang lebih sempurna, yang kemudian dinamakannya Linux (Nova Novriansyah, 2001).

2.10. GAMBAS

Gambas (*Gambas Almost Means Basic*) merupakan salah satu bahasa pemrograman yang berorientasi pada grafis atau visual, namun begitu dapat juga untuk membuat program *text oriented* (berjalan di konsol). Model bahasa yang dimiliki oleh Gambas mirip dengan bahasa pemrograman Visual Basic karena

pada dasarnya Gambas memang dibuat sebagai interpreter bahasa Basic. Walaupun mirip dengan Visual Basic dan file-file bagian program memiliki kesamaan, Gambas tidak bias membaca program yang dibuat dengan Visual Basic. Menurut pembuatnya, hal tersebut takkan pernah dilakukan (Yunianto, 2005).

Gambas dibuat oleh seorang Perancis bernama Benoit Minisini, yang tinggal didaerah sekitar kota Paris. Pria yang lahir tahun 1973 ini memiliki kesenangan dalam pemrograman sejak berusia 12 tahun dan telah menjalani pekerjaannya sejak tahun 1997. Bahasa pemrograman pertama yang dikuasainya adalah bahasa Basic pada mesin CPC Amstrad 464, yang kemudian beralih pada Atari 520 STE (Yunianto, 2005).

Telah banyak bahasa, compiler, assembler dan interpreter yang ditulis oleh Minisini. Kemudian selama menjalani pendidikan di E.P.I.T.A., Minisini menulis interpreter Lisp untuk platform Windows 3.1. Selama enam bulan bermain dengan Windows, Minisini menemukan begitu banyak bugs pada system operasi tersebut (Yunianto, 2005).

Walaupun seorang programmer, Minisini juga mempunyai hobby yang berkaitan dengan seni. Hal inilah yang membuat pengembangan Gambas begitu lambat, walau begitu Minisini tidak melupakan proyek tersebut yakni membuat Gambas sebagai rasa sayangnya pada bahasa Basic (Yunianto, 2005).

2.11. BASIS DATA

Basis data (atau *database*) adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Database digunakan untuk menyimpan informasi atau data yang terintegrasi dengan baik di dalam komputer (Achmad Solichin, 2010).

Merupakan kumpulan data yang terintegrasi satu sama lain. Setiap pemakai yang diberi wewenang dapat melakukan akses terhadap data tersebut. Database biasanya terorganisasi dalam beberapa komponen yang terdiri dari satu atau lebih tabel. Tabel digunakan untuk menyimpan data dan terdiri dari baris dan kolom. Data dapat ditampilkan, domodifikasi dan dihapus dari tabel oleh pemakai yang diberi wewenang dalam menggunakan database tersebut (Nufransa Wira Sakti, 1999).

Berikut ilustrasi atau gambaran mengenai database yang biasa digunakan pada umumnya;

Tabel 2.3. tabel basis data

Id	Nama	Telepon	
1	Junaidi Gafar	9087678	→ Baris 1
2	Roni Abdullah	2521110	→ Baris 2
3	Baharudin Gani	7667776	→ Baris 3
4	Wawan Kurniawan	4561234	→ Baris 4
↑ Kolom 1	↑ Kolom 2	↑ Kolom 3	

Keterangan gambar:

Kolom: Setiap tabel terdiri dari satu kolom atau lebih, kolom biasa disebut juga dengan *field*

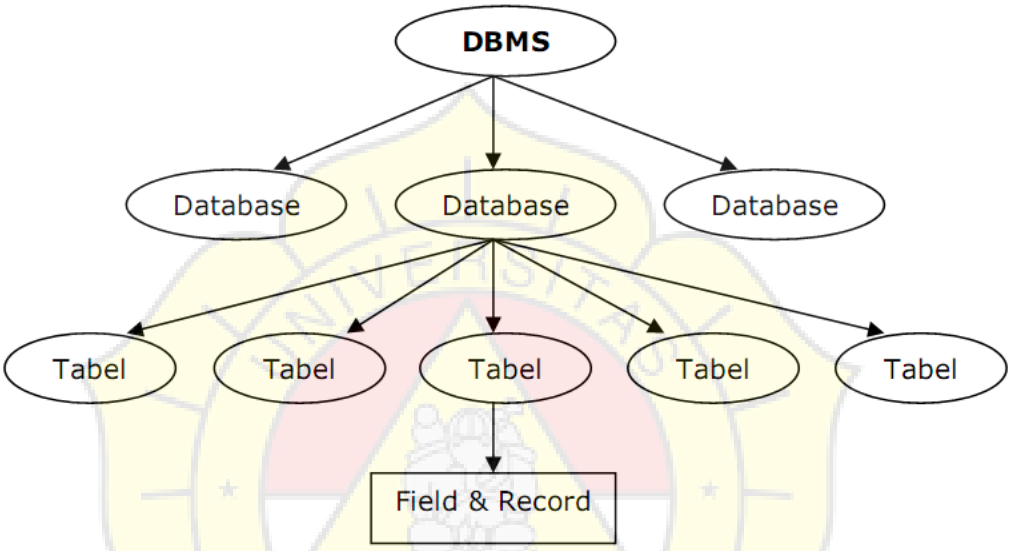
Baris: baris dalam tabel menggambarkan jumlah data yang ada, oleh karena itu satu baris data (satu set data) disebut juga satu *record*. Pada tabel di atas, baris pertama disebut juga record pertama dan data terdiri dari empat buah record.

Database dan tekonologi database mempunyai dampak yang besar dalam perkembangan komputer pada umumnya. Adalah hal yang wajar bila database memainkan peranan penting dalam pemakaian komputer dalam bidang bisnis, teknik, medik, hukum, pendidikan dan lain-lain (Nufransa Wira Sakti, 1999).

Database dapat dijalankan secara manual atau menggunakan mesin, dalam hal ini komputer. Contoh yang manual adalah katalog di perpustakaan. Database yang dikelola oleh komputer dapat dibuat baik oleh program aplikasi yang khusus dibuat maupun dengan *database management system* (Nufransa Wira Sakti, 1999).

Database Management System (DBMS) adalah kumpulan program perangkat lunak (*software*) yang memperbolehkan user untuk membuat dan memelihara database. DBMS berupa sebuat software yang dapat menyediakan fasilitas untuk melakukan proses *defining*, *constructing* dan *manipulating*. Proses *defining* adalah merincikan model, struktur dan pembatas data untuk disimpan dalam database. Proses *constructing* adalah proses menyimpan data ke dalam

berbagai macam media penyimpanan yang pengendaliannya diatur oleh DBMS. Sedangkan *manipulating* adalah fungsi untuk menampilkan data tertentu (*retrieve*), mengubah data dan membuat laporan (*report*) dari data yang ada (Nufransa Wira Sakti, 1999).



Gambar 2.6. struktur basis data

Relational Database Management System (RDBMS) adalah sebuah sistem yang secara otomatis menyatukan semua DBMS yang saling berhubungan. RDBMS biasanya menggunakan 4th *Generation Languange* (4GL) dan sangat fleksibel sehingga data dapat dimodifikasi dengan mudah, demikian pula dengan struktur databasenya. Berdasarkan hal tersebut, SQL-lah yang memungkinkan untuk membuat suatu RDBMS (Nufransa Wira Sakti, 1999).

2.12. RDBMS dan Perkembangannya

Beberapa konsep dari RDBMS yang pertama kali diperkenalkan oleh Dr. E.F. Codd pada tahun 1970 antara lain:

1. Sebuah RDBMS harus mampu untuk mengelola semua database secara keseluruhan melalui kemampuan relasionalnya.
2. Semua informasi dalam RDBMS diwakili oleh nilai dalam tabel secara eksplisit.
3. Setiap nilai dalam RDBMS dijamin dapat diakses dengan menggunakan kombinasi nama tabel, nilai kunci primer dan nama kolom.
4. *Physical Data Independence*: program aplikasi secara logika tidak terpengaruh ketika struktur penyimpanan dan metode pengaksesan ditambah.
5. *Logical Data Independence*: program aplikasi secara logika tidak terpengaruh ketika diadakan perubahan pada struktur tabel.
6. *Distribution Independence*: program aplikasi secara logika tidak terpengaruh ketika data untuk pertama kali didistribusikan atau ketika didistribusikan ulang.

Ide RDBMS ini menggunakan konsep matematika yaitu aljabar relasional untuk membagi data dalam beberapa himpunan (*set*) dan saling berhubungan dalam subset. Dalam model relasional, data dipisahkan dalam beberapa set yang parallel dengan struktur tabel. Struktur tabel ini mengandung elemen data individual yang disebut kolom atau field. Satu set kumpulan kolom disebut *record/row* (Nufransa Wira Sakti, 1999).

Hal yang paling penting bagi *database designer* setelah memilih *platform hardware* adalah membuat struktur tabel. Pembuatan struktur ini, dapat mempengaruhi performa dan pemrograman selama proses pengembangan system. Proses memisahkan data dalam bentuk yang unik disebut *normalisasi* (Nufransa Wira Sakti, 1999).

Teknologi komputer telah mengubah cara kerja melakukan bisnis di seluruh dunia. Informasi yang dulu disimpan dalam sebuah gudang yang penuh dengan *filling cabinet* sekarang dapat diakses secara langsung dengan mudah hanya dengan mengklik *mouse*. Meskipun 20 tahun yang lalu database yang sangat besar telah dimasukkan ke dalam komputer mainframe, namun harganya sangat mahal. Apabila pemakai diminta untuk menampilkan data, maka seseorang akan memberitahu bagian data kemudian data yang diminta akan diantarkan secepat mungkin (Nufransa Wira Sakti, 1999).

2.13. SQL (*Structured Query Languange*)

SQL pertama kali dikembangkan pada akhir tahun 1970-an di laboratorium IBM San Jose, California. SQL yang biasanya dibaca dengan "sequel", pada mulanya dikembangkan untuk produk DB2 (*Database*) yang dimiliki oleh IBM. SQL adalah bahasa non procedural yang sangat kontras dengan bahasa generasi ketiga (*Third_Generation Languages*/3GL). Yang dimaksud dengan bahasa non procedural adalah SQL menjelaskan bagaimana data ditampilkan, dihapus atau dimasukkan dan bukan menjalankan prosedur pemrograman untuk menampilkan data. Dengan bahasa yang lebih mudah, SQL

memerintahkan database untuk melakukan apa yang harus dilakukan, bukan bagaimana melakukannya (Nufransa Wira Sakti, 1999).

Karakteristik yang membedakan RDBMS dan DBMS adalah RDBMS menyediakan bahasa database yang *set-oriented* (bahasa yang memproses beberapa set data dalam group). Kebanyakan RDBMS menggunakan bahasa SQL. Disini terlihat betapa pentingnya SQL untuk membangun sebuah RDBMS.

Dua organisasi standar internasional, American National Standars Instate (ANSI) dan International Standards Organization (ISO), mempromosikan SQL sebagai standar industri pada Oktober 1986.

SQL adalah bahasa standar yang meliputi perintah-perintah untuk menyimpan, menerima, memelihara dan mengatur akses-akses ke basis data serta digunakan untuk memanipulasi dan menampilkan data dari RDBMS. SQL membuat programmer atau database administrator dapat melakukan hal-hal berikut ini:

- Memodifikasi struktur database
- Mengganti *setting system security*
- Menambah wewenang pemakai pada database atau tabel
- Menampilkan informasi dari database
- Mengubah isi dari database
- Membuat keamanan data
- Menangani proses transaksi diantara aplikasi
- Mentransfer data antara database yang berbeda

2.14. MySQL

MySQL merupakan *software database open source*. MySQL pertama kali dibuat dan dikembangkan di Swedia, yaitu oleh David Axmark, Allan Larsson dan Michael "Monty" Widenius. Mereka mengembangkan MySQL sejak tahun 1980-an. Saat ini versi MySQL yang sudah stabil mencapai versi 5x, dan sedang dikembangkan versi 6x. Untuk lebih lengkapnya dapat dilihat di situs resmi MySQL (Achmad Solichin, 2010).

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (bahasa Inggris: *database management system*) atau DBMS yang *multithread*, *multi-user*. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis di bawah lisensi GNU *General Public License* (GPL), tetapi mereka juga menjual di bawah lisensi komersial untuk kasus-kasus di mana penggunaannya tidak cocok dengan penggunaan GPL. Tidak seperti PHP atau Apache yang merupakan software yang dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh penulisnya masing-masing, MySQL dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia yaitu MySQL AB. MySQL AB memegang penuh 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.

Fitur-fitur MySQL antara lain:

- *Relational Database System*. Seperti halnya software database lain yang ada di pasaran, MySQL termasuk RDBMS.

- Arsitektur *Client-Server*. MySQL memiliki arsitektur client-server dimana server database MySQL terinstal di server. Client MySQL dapat berada di komputer yang sama dengan server, dan dapat juga di komputer lain yang berkomunikasi dengan server melalui jaringan bahkan internet.
- Mengenal perintah SQL standar. SQL (*Structured Query Language*) merupakan suatu bahasa standar yang berlaku di hampir semua software database. MySQL mendukung SQL versi SQL:2003.
- Mendukung *Sub Select*. Mulai versi 4.1 MySQL telah mendukung *select* dalam *select* (*sub select*).
- Mendukung *Views*. MySQL mendukung *views* sejak versi 5.0
- Mendukung *Procedure Stored* (SP). MySQL mendukung SP sejak versi 5.0
- Mendukung *Triggers*. MySQL mendukung trigger pada versi 5.0 namun masih terbatas. Pengembang MySQL berjanji akan meningkatkan kemampuan *trigger* pada versi 5.1.
- Mendukung *replication*.
- Mendukung transaksi.
- Mendukung *foreign key*.
- Tersedia fungsi GIS.
- *Free* (bebas didownload)
- Stabil dan tangguh
- Fleksibel dengan berbagai pemrograman
- *Security* yang baik

- Dukungan dari banyak komunitas
- Perkembangan software yang cukup cepat.