BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Definisi sistem berkembang sesuai dengan konteks dimana pengertian sistem itu digunakan. Berikut akan diberikan beberapa defenisi sistem secara umum :

1. Kumpulan dari bagian-bagian yang bekerja sama untuk mencapai tujuan yang sama.
2. Sekumpulan objek-objek yang saling berelasi dan berinteraksi serta hubungan antar objek bisa dilihat sebagai satu kesatuan yang dirancang untuk mencapai satu tujuan.

Dengan demikian, secara sederhana sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur atau variabel-variabel yang saling terorganisasi, saling berinteraksi, dan saling bergantung satu sama lain (Hanif Al Fatta ; 2007: 3)

II.2. Informasi

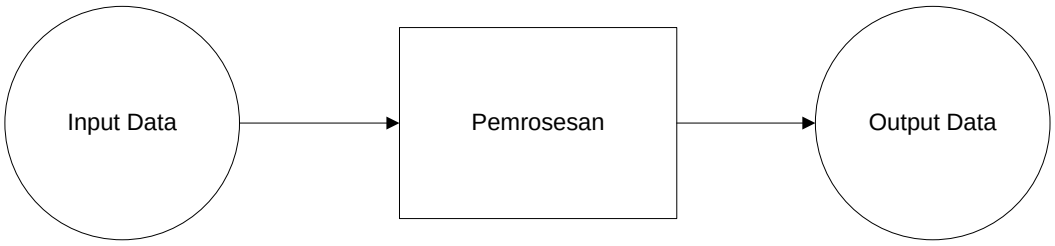
Untuk memahami pengertian sistem informasi, harus dilihat keterkaitan antara data dan informasi sebagai entitas penting pembentuk sistem informasi. Data merupakan nilai, keadaan, atau sifat yang berdiri sendiri lepas dari konteks apapun. Sementara informasi adalah data yang telah diolah menjadi sebuah bentuk

yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan saat ini atau mendatang. (Hanif Al Fatta ; 2007: 9)

II.3. Sistem Informasi

Sistem Informasi didefinisikan sebagai suatu alat untuk menyajikan informasi dengan cara sedemikian rupa sehingga bermanfaat bagi penerimanya.Tujuannya adalah untuk menyajikan informasi guna pengambilan keputusan pada perencanaan, pemrakarsaan, pegorganisasian, pengendalian kegiatan operasi subsistem suatu perusahaan, dan menyajikan sinergi organisasi pada proses. (Hanif Al Fatta ; 2007: 9)

Dengan demikian, sistem informasi berdasarkan konsep (*input, processing, output - IPO*) dapat dilihat pada gambar berikut :



Gambar II.1. Konsep Input-Proses-Output
Sumber : (Hanif Al Fatta ; 2007: 9)

II.4. Sekilas Perkembangan Akuntansi

Pada dasarnya akuntansi sudah ada sejak abad pertama. Pada saat itu, sering dilakukan oleh para pemegang catatan persediaan, namun belum ada yang membukukannya secara sistematis sebagai ilmu pengetahuan. Perkembangan

akuntansi ada tiga fase, yaitu akuntansi sebelum Pacioli, akuntansi masa Pacioli, dan akuntansi pasca Pacioli.

A. Akuntansi Sebelum Pacioli

Pacioli tidak mengklaim dirinya sebagai penemu akuntansi *double entry*. Hal ini karena Pacioli bekerja untuk pertama kalinya sebagai pemegang buku dibawah aturan akuntansi Venetian, yang dikembangkan akibat adanya perdagangan internasional. Selama abad ke 11 dan 12, kota-kota di Italia, seperti Geneva, Florence, dan Venice merupakan kota pusat perdagangan dan rute sirkulasi produk dari berbagai kota atau negara lainnya, untuk itu besar kemungkinan para pedagang /ahli ari bangsa Italia mengadopsi pengetahuan *double entry bookkeeping* dari Alexandria, Contantinopel, atau kota lainnya dari Timur Tengah. Misalnya pada periode Mamluk (1250-1517), dimana pada periode ini tumbuh kerajaan-kerajaan kecil yang merupakan pecahan dari kekuasaan Abbasiyah, namun perekonomian tumbuh pesat. Oleh karena itu, sangat rasional jika sistem *double entry bookkeeping* sudah digunakan di Egyp dan Syria. Koleksi Mesir kuno menunjukkan adanya bentuk jurnal dan akun empat kolom, dengan mencantmkan istilah debit dan kredit.

B. Akuntansi Masa Pacioli

Luca Pacioli (1445-1517) dilahirkan pada tahun 1445 di Borgo San Sepulcro, kota kecil di Tuscan, Italia. Ia ahli matematika yang berkonsentrasi dalam aritmatika, aljabar an geometri. Secara tampak, Pacioli mempresentasikan aritmatika dan aljabar yang dikembangkan di sekolah-sekolah Moorish, Spayol, selama dominasi Arab dan ketika perekonomian Eropa berhubungan langsung

dengan dunia Arab. Pacioli menjelaskan prosedur akuntansi yang diadopsi di dunia bisnis dengan memasukkan mekanisme *double entry* dalam akuntansinya. Setiap jurnal dan *posting* ke buku besar melalui dua sisi, satu sisi debit dan sisi lainnya kredit. Pacioli mencatat transaksinya ke dalam tiga tahap , yaitu (1) di*memorandum*, sebagai buku harian dan merupakan catatan kronologis dan rinci. Dari *memorandum* pemegang buku besar mencatat transaksi tersebut ke dalam buku (2) *journal* yaitu mencerminkan jumlah uang yang sama dalam debit dan kredit. Pada akhir periode ringkasan di-*posting* ke buku besar (*ledger*), sebagai representasi dari akun-akun.

C. Akuntansi Pasca Pacioli

Pada era setelah Pacioli sampai sebelum abad ke-19, sistem *double entry accounting* tidak banyak mengalami perkembangan dan perubahan. Baru pada era revolusi industri (abad 19), sebagai dampak dari kapitalisme, sistem akuntansi dan teori akuntansi mengalami perkembangan, pada abad 20 mengalami perkembangan yang cukup signifikan, dengan isu sentral pada *cost accounting*. Walaupun akuntansi berkembang pesat sampai dengan sekarang, namun Pacioli dan karyanya tidak pernah dilupakan oleh para periset dan para ilmuan akuntansi. Mereka tetap menghargai dan mneyatakan bahwa karya pertama yang terbesar dalam perkembangan sistem akuntansi ditulis oleh Luca Pacioli. Ini ditandai sebagai karya ”*Summa*”. (Mursyidi ; 2010 :11-13)

II.4.1. Defenisi Akuntansi

Defenisi akuntansi ditinjau dari dua sudut pandang, yaitu dari sudut pandang pengguna akuntansi dan proses kegiatan akuntansi. Defenisi akuntansi dari sudut pandang pengguna akuntansi, yaitu suatu disiplin ilmu dan atau aktivitas jasa yang menyediakan infomasi yang diperlukan untuk melaksanakan kegiatan secara efisien dan mengevaluasi kegiatan suatu entitas atau transaksi yang bersifat keuangan (*financial*). Adapun defenisi akuntansi dari sudut pandang proses kegiatannya, akuntansi adalah proses pencatatan, penggolongan, peringkasan, pelaporan dan penganalisaan data keuangan suatu entitas.

Dari dua defenisi tersebut dapat disimpulakan bahwa secara umum akuntansi adalah suatu sistem informasi keuangan yang manghasilkan laporan kepada pihak-pihak yang berkepentingan mengenai aktivitas ekonomi dan kondisi perusahaan. (Epi Indirani ; 2011 : 12)

Kegiatan selama periode akuntansi adalah kegiatan mencatat transaksi-transaksi hingga kegiatan menutup buku, yang dapat dirinci sebagai berikut :

1. Jurnal yaitu : kegiatan mencatat transaksi-transaksi keuangan yang terjadi pada perusahaan.
2. Posting yaitu : kegiatan pembukuan catatan dari jurnal ke dalam rekening buku besar yang bersangkutan.
3. Neraca saldo (*trial balance*) yaitu : kegiatan menguji kebenaran saldo-saldo debit dan kredit rekening buku besar dengan cara menyusun saldo-saldo rekening buku besar ke dalam suatu daftar yang disebut neraca saldo.

4. Ayat penyesuaian (*adjusting entries*) yaitu : kegiatan menyesuaian jumlah-jumlah yang ada pada necara saldo, yang belum sesuai, sehingga jumlah-jumlah tersebut sesuai dengan keadaan yang sebenarnya pada akhir periode.
5. Laporan keuangan (*financial statement*) yaitu : kegiatan menyusun neraca (*balance sheet*), laporan laba-rugi (*income statement*), dan laporan sisa laba berdasarkan data-data neraca saldo yang telah disesuaikan.
6. Ayat penutup (*clossing entries*) yaitu : kegiatan menyusun pos-pos penutup. (F. Winarni ; 2011 : 27-28)

Akuntansi adalah sebuah sistem informasi yang menghasilkan informasi keuangan kepada pihak-pihak yang berkepentingan mengenai aktivitas ekonomi dan kondisi suatu perusahan. (Rudianto ; 2009 : 4)

Akuntansi adalah aktivitas mengumpulkan, menganalisis, menyajikan dalam bentuk angka, mengklasifikasikan, mencatat, meringkas dan melaporkan aktivitas/transaksi perusahaan dalam bentuk informasi keuangan. (Rudianto ; 2009 : 14)

Aktiva (Harta) = Kewajiban (Hutang) + Modal
Aktiva (Harta) – Kewajiban (Hutang) = Modal

Gambar II.2. Persamaan Akuntansi
Sumber : (Rudianto ; 2009 : 24)

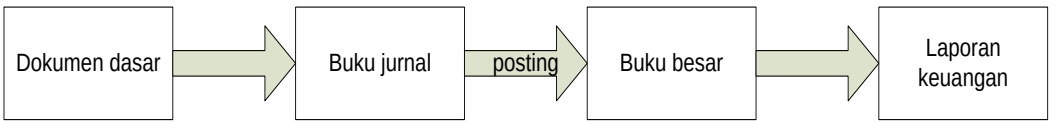
II.5. Sistem Akuntansi

Sistem akuntansi dapat dilakukan secara manual atau menggunakan perangkat lunak komputer (komputerisasi). Memahami sistem akuntansi manual dapat membantu dalam mengenali hubungan antara data akuntansi dengan laporan akuntansi. Selain itu, kebayakan sistem komputerisasian tetap menggunakan prinsip-prinsip yang digunakan dalam sistem manual. Oleh karena itu, kita akan memberi ilustrasi mengenai sistem manual. (James M.Reeve, dkk ; 2008 : 224)

Sistem komputerisasi akuntansi merupakan aplikasi dari accounting system yang berbasis sistem database dengan menggunakan teknologi komputer. Dalam sistem komputerisasi, sejumlah kegiatan akuntansi tidak dilakukan lagi sebab sitem komputerisasi akuntansi dapat memproses transaksi dan menyususn laporan keuangan dengan sedikit sekali ikut campur tangan manusia. Komputerisasi akuntansi di desain untuk mempermudah pekerjaan akuntans, namun keberhasilan komputer akuntansi membutuhkan penguasaan teknologi informasi dan komputer. (Mardi ; 2011 : 29)

II.5.1. Siklus Akuntansi

Siklus Akuntansi adalah urutan kerja yang harus dibuat oleh akuntan, sejak awal hingga menghasilkan laporan keuangan suatu perusahaan.



Gambar II.3. Siklus Akuntansi

Sumber : (Rudianto ; 2009 : 14)

Keterangan gambar :

a. Dokumen dasar

   Adalah bukti transaksi yang dijadikan dasar oleh akuntan untuk mencatat, seperti : faktur, kuitansi, nota penjualan, invoice, dll.

b. Jurnal (*Journal*)

   Adalah aktivitas meringkas dan mencatat transaksi perusahaan berdasarkan dokumen dasar. Tempat untuk mencatat dan meringkas transaksi tersebut disebut Buku jurnal.

c. *Posting*

   Adalah aktivitas memindahkan catatan di buku jurnal ke dalam buku besar sesuai jenis transaksi dan nama mperkiraan masing-masing.

d. Buku besar (*General Ledger*)

   Adalah kumpulan dari semua akun/ perkiraan yang dimiliki oleh perusahaan yang saling berhibungan satu sama lainnya dan merupakan suatu kesatuan.

e. Akun/ Perkiraan (*Account*)

   Adalah suatu kelas informasi di dalam suatu sistem akuntansi. Atau suatu media yang digunakan untuk mencatat informasi sumber daya perusahaan dan informasi lainnya berdasarkan jenisnya. Misalnya perkiraan kas, perkiraan piutang, akun modal, dsb. (Rudianto ; 2009 : 14)

II.6. Sistem Informasi Akuntansi

Sistem informasi akuntansi adalah suatu komponen organisasi yang mengumpulkan, mengklasifikasi, mengolah, menganalisis, dan

mengkomunikasikan informasi finansial dan pengambilan keputusanyang relevan kepada pihak di luar dan di dalam perusahaan. (Teguh Wahyono ; 2004 : 13)

II.7. Jurnal Khusus Penjualan

Jurnal khusus penjualan adalah buku harian yang hanya digunakan untuk mencatat transaksi penjualan produk perusahaan secara secara kredit. Transaksi penjualan secara kredit jika dicatat dengan jurnal umum dengan mendebet akun piutang dan mengkredit akun penjualan sebesar nilai transaksinya. (Rudianto ; 2009 : 136-139)

II.8. Pengertian Piutang

Piutang adalah Nilai sisa utang yang timbul karena perusahaan menjual barangnya atau memberikan jasanya kepada para pelanggan dan menerima janji bahwa pelanggan akan memberikan sejumlah uang kepada perusahaan pada suatu waktu dimasa yang akan datang.(Rudianto ; 2009 : 107)

II.9. Pemodelan UML

Pemodelan adalah gambaran dari realita yang simpel dan dituangkan dalam bentuk pemetaan dengan aturan tertentu. (Rosa A.S, M. Shalahuddin; 2011: 116)

Pada perkembangan teknik pemrograman beroientasi objek, muncullah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modeling Languange* (UML). *UML* hanya bergungsi untuk melakukan pemodelan. Jadi penggunaan *UML* tidak terbatas pada metodologi tertentu,

meskipun pada kenyataannya *UML* paling banyak digunakan pada metodologi berorientasi objek. (Rosa A.S, M. Shalahuddin; 2011: 118)

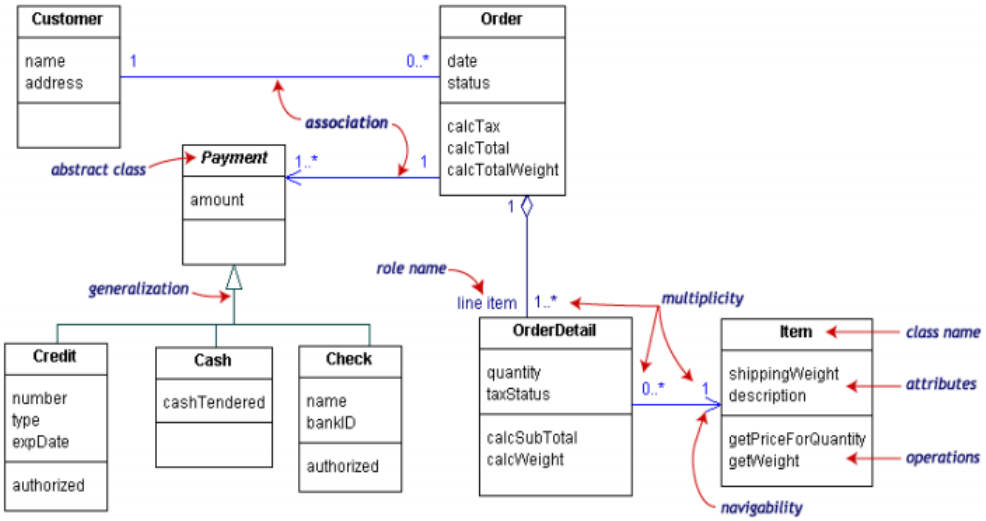
Diagram *UML* terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori yaitu :

1. *Structure Diagram*

Yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan. Yang termasuk dalam structure diagram adalah sebagai berikut:

a. *Class diagram*

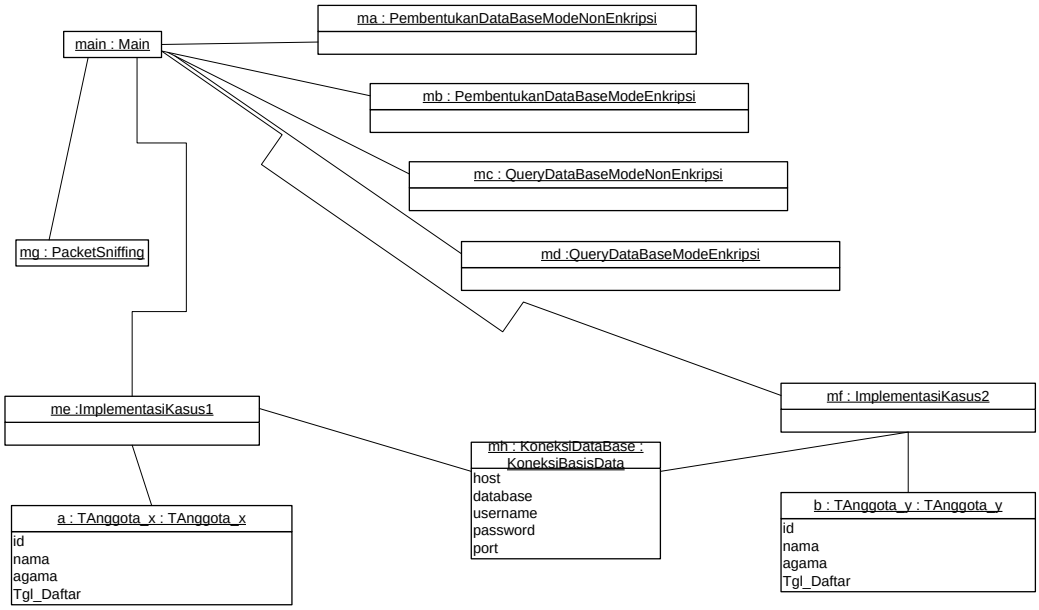
Digaram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefenisian kelas-kelas yang akan dibuat untuk membangun sistem. (Rosa A.S, M. Shalahuddin; 2011: 122). Contoh *class diagram* dapat dilihat pada gambar II.4.



Gambar II.4. Contoh Class Diagram
Sumber : (Romi Satria Wahono dan Sri Dharwiyanti ; 2003 : 6)

b. Object diagram

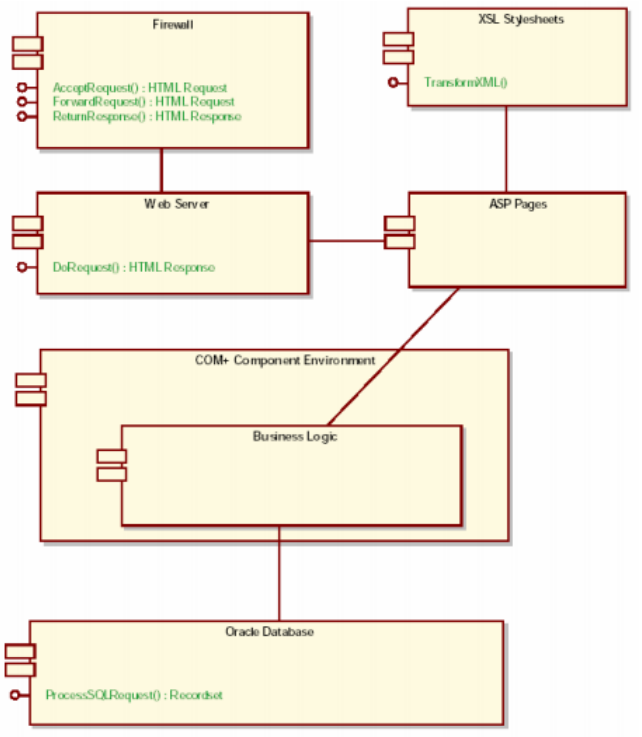
Diagram objek menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem. (Rosa A.S, M. Shalahuddin; 2011: 124)



Gambar II.5. Contoh Object Diagram
Sumber : (Rosa A.S. dan M. Shalahuddin ; 2011 : 163)

c. Component diagram

Diagram komponen atau *component diagram* dibuat untuk menunjukkan organisasi dan ketergantungan di antara kumpulan komponen di dalam sebuah sistem. (Rosa A.S, M. Shalahuddin; 2011: 125)



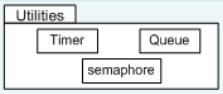
Gambar II.6. Contoh Component Diagram
Sumber : (Romi Satria Wahono dan Sri Dharwiyanti ; 2003 : 10)

d. Composite diagram

Composite strukture diagram baru mulai ada pada UML versi 2.0, pada versi 1.x diagram ini belum muncul. Digaram ini dapat digunakanuntuk menggambarkan struktur dari bagian-bagian yang saling terhubung maupun mendeskripsikan struktur pada saat berjalan (*runtime*) dari *instance* yang saling terhubung. (Rosa A.S, M. Shalahuddin; 2011: 127)

e. Package diagram

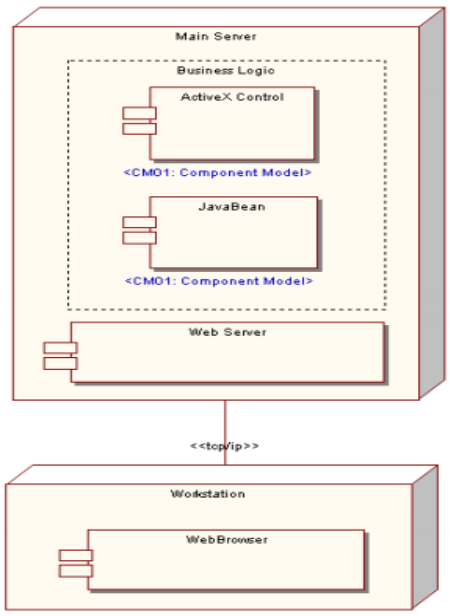
Diagram ini meyediakan cara mengumpulkan elemen-elemen yang saling terkait dalam diagram UML. (Rosa A.S, M. Shalahuddin; 2011: 128)

Simbol	Arti	Deskripsi
	Pakage	Merupakan sebuah bungkusan dari satu atau lebih kelas atau elemen
	Elemen dalam package	Elemen dalam package digambarkan dalam package

Gambar II.7. Contoh Package Diagram
Sumber : (Rosa A.S. dan M. Shalahuddin ; 2011 : 128)

f. Deployment diagram

Diagram deployment atau *deployment diagram* menunjukkan konfigurasi komponen dalam proses eksekusi aplikasi. (Rosa A.S, M. Shalahuddin; 2011: 129)



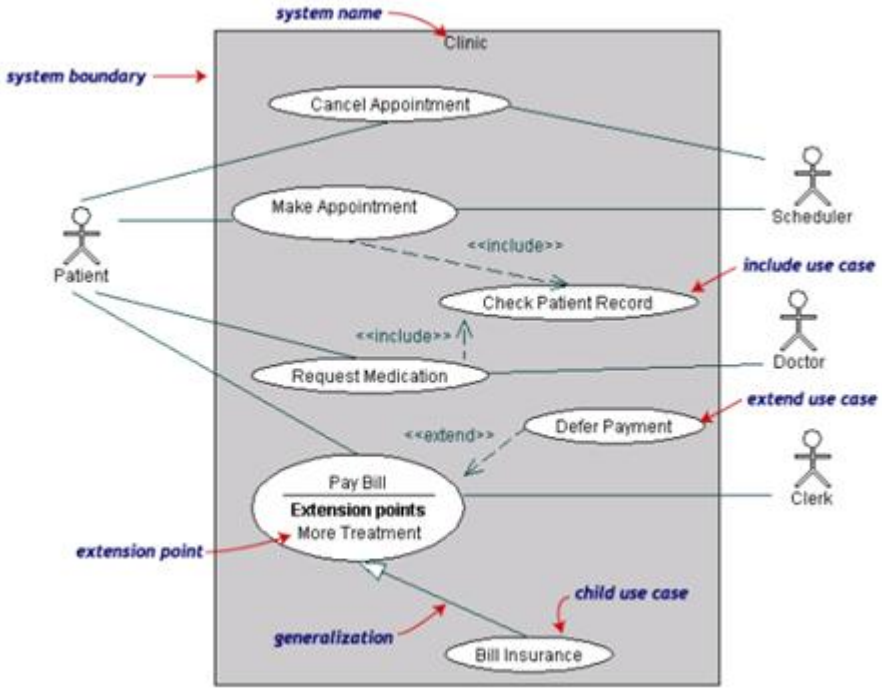
Gambar II.8. Contoh Deployment Diagram
Sumber : (Romi Satria Wahono dan Sri Dharwiyanti ; 2003 : 11)

2. Behavior Diagram

Yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem. Yang termasuk dalam *bahvior* diagram adalah sebagai berikut :

a. Use case

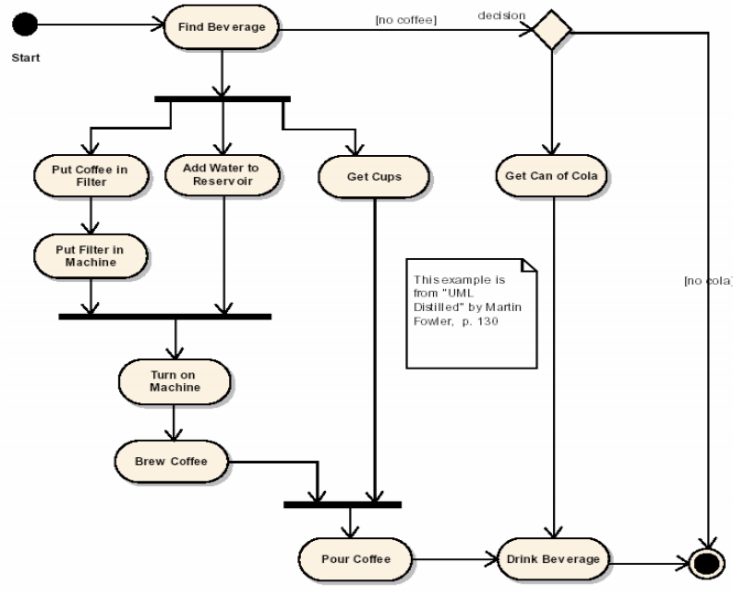
Use case atau diagram use case merupakan pemodelan untuk kelakuan (*bahavior*) sistem informasi yang akan dibuat. (Rosa A.S, M. Shalahuddin; 2011: 130)



Gambar II.9. Contoh Use Case Diagram
Sumber : (Romi Satria Wahono dan Sri Dharwiyanti ; 2003 : 5)

b. Activity diagram

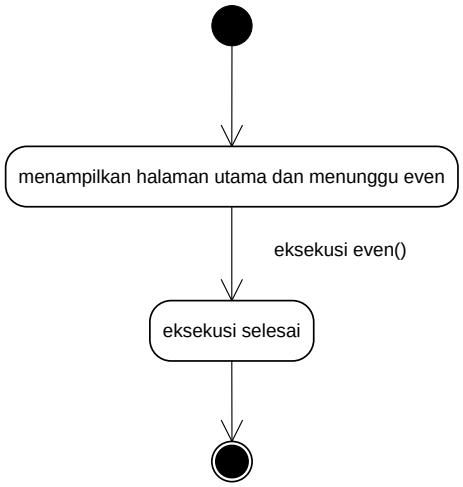
Digaram aktivitas atau activity diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. (Rosa A.S, M. Shalahuddin; 2011: 134)



Gambar II.10. Contoh Activity Diagram
Sumber : (Romi Satria Wahono dan Sri Dharwiyanti ; 2003 : 8)

c. State machine diagram

Diagram mesin status digunakan untuk menggambarkan perubahan status atau transisi status dari sebuah mesin atau sistem. (Rosa A.S, M. Shalahuddin; 2011: 136). Contoh *state machine diagram* dapat dilihat pada gambar II.11.



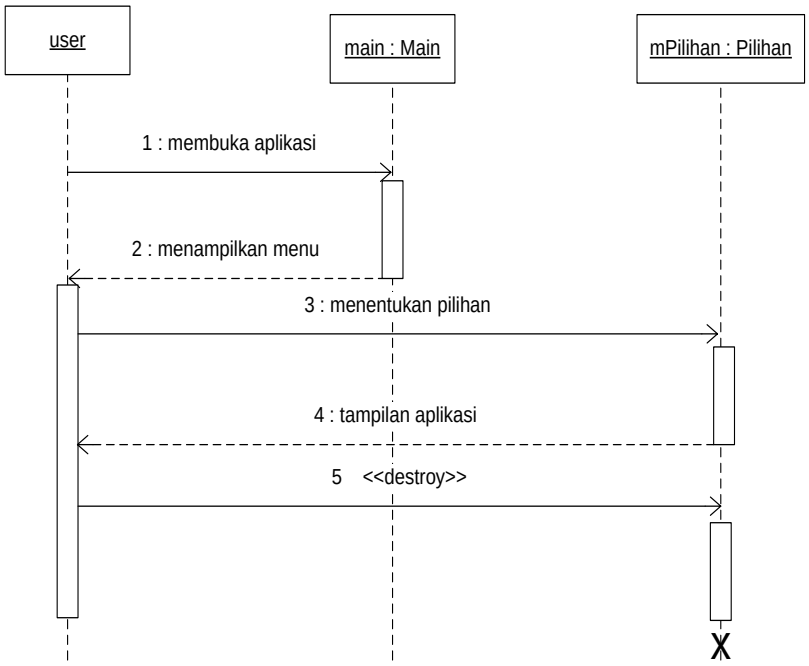
Gambar II.11. Contoh Class Diagram
Sumber : (Rosa A.S, M. Shalahuddin; 2011: 174)

3. Interaction Diagram

Yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi antar sub sistem pada suatu sistem. Yang termasuk dalam *interaction* diagram adalah sebagai berikut :

a. Sequence diagram

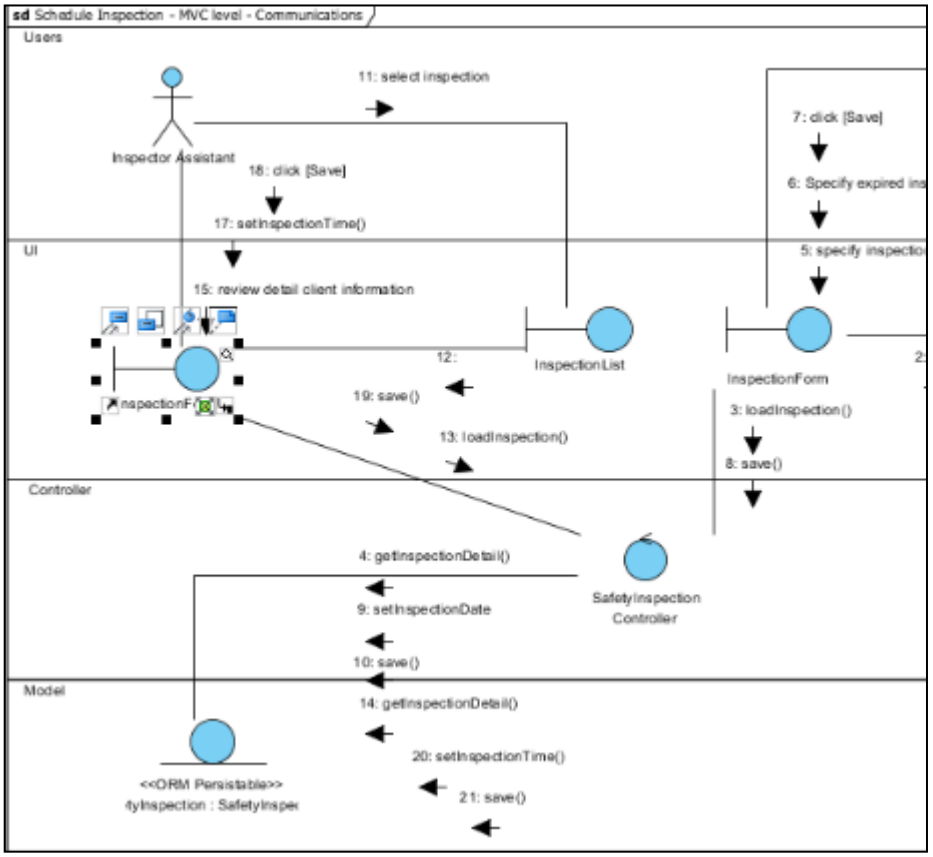
Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. (Rosa A.S, M. Shalahuddin; 2011: 137). Contoh s*equence diagram* dapat dilihat pada gambar II.12.



Gambar II.12. Contoh Class Diagram
Sumber : (Rosa A.S, M. Shalahuddin; 2011: 164)

b. Communication diagram

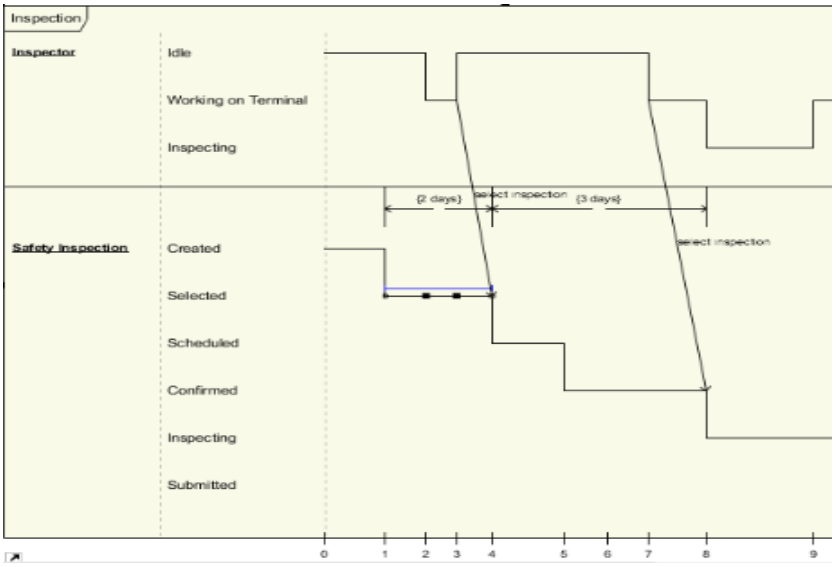
Communication diagram atau diagram komunikasi menggambarkan interaksi antar objek /bagian dalam bentuk urutan pengiriman pesan. (Rosa A.S, M. Shalahuddin; 2011: 140). Contoh *communication diagram* dapat dilihat pada gambar II.13.



Gambar II.13. Contoh Communication Diagram
Sumber : (Tgl : 7-8-2012 ; Jam : 10.52 ; http://www.visual-paradigm.com/product/vpuml/provides/behavioralmodeling.jsp)

c. *Timing diagram*

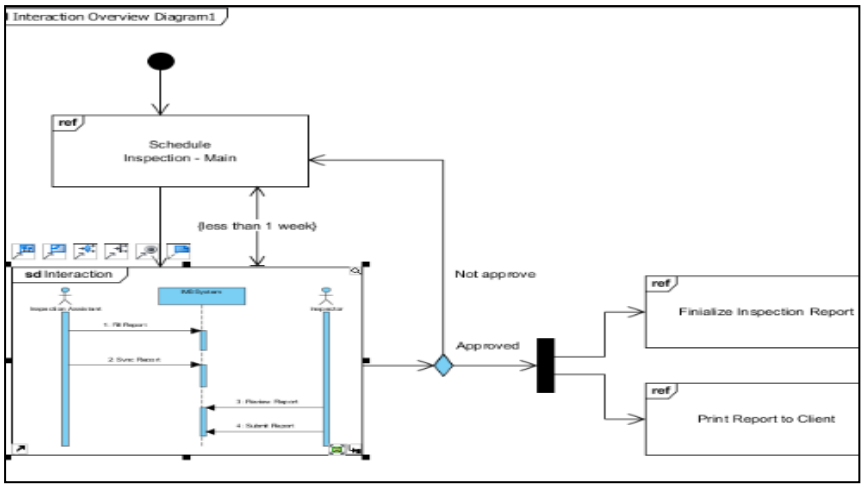
Timing diagram merupakan diagram yang fokus pada penggambaran terkait batasan waktu. (Rosa A.S, M. Shalahuddin; 2011: 141). Contoh *timing diagram* dapat dilihat pada gambar II.14.



Gambar II.14. Contoh Timing Diagram
Sumber : (Tgl : 7-8-2012 ; Jam : 10.52 ; http://www.visual paradigm.com/product/vpuml/provides/behavioralmodeling.jsp)

d. *Interaction overview diagram*

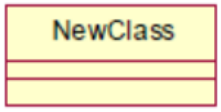
Diagram ini mirip dengan diagram aktivitas yang berfungsi untuk menggambarkan sekumpulan urutan aktivitas. (Rosa A.S, M. Shalahuddin; 2011: 143)

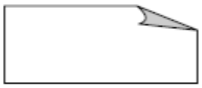


Gambar II.15. Contoh Interaction Overview Diagram
Sumber : (Tgl : 7-8-2012 ; Jam : 10.52 ; http://www.visual-paradigm.com/product/vpuml/provides/behavioralmodeling.jsp)

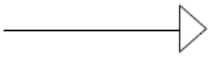
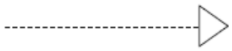
Tabel II.2. Notasi-notasi dalm UML

Notasi	Keterangan
Actor 	*Actor* menggambarkan segala pengguna *software* aplikasi (*user*). *Actor* memberikan suatu gambaran jelas tentang apa yang harus dikerjakan *software* aplikasi. Sebagai contoh sebuah actor dapat memberikan input kedalam danmenerima informasi dari *software* aplikasi, perlu dicatat bahwa sebuah *actor* berinteraksi dengan *use case*, tetapi tidak memiliki kontrol atas *use case*. Sebuah *actor* mungkin seorang manusia, satu *device*, *hardware* atau sistem informasi lainnya.
Use Case 	*Use case* menjelaskan urutan kegiatan yang dilakukan *actor* dan sistem untuk mencapai suatu tujuan tertentu. Walaupun menjelaskan kegiatan, namun *use case* hanya menjelaskan apa yang dilakukan oleh *actor* dan sistem bukan bagaimana *actor* dan sistem melakukan kegiatan tersebut. 1. *Use-case* Konkret adalah *use case* yang dibuat langsung karena keperluan *actor*. *Actor* dapat melihat dan berinisiatif terhadapnya . 2. *Use-case* Abstrak adalah *use case* yang tidak pernah berdiri sendiri. *Use case* abstrak senantiasa termasuk didalam (*include*), diperluas dari (*extend*) atau

	memperumum (*generalize*) *use case* lainnya. Untuk menggambarkannya dalam *use case* model biasanya digunakan *association relationship* yang memiliki *stereotype include*, *extend* atau *generalization relationship*. Hubungan *include* menggambarkan bahwa suatu *use case* seluruhnya meliputi fungsionalitas dari *use case* lainnya. Hubungan *extend* antar *use case* berarti bahwa satu *use case* merupakan tambahan fungsionalitas dari *use case* yang lain jika kondisi atau syarat tertentu terpenuhi.
Class 	*Class* merupakan pembentuk utama dari sistem berorientasi obyek, karena *class* menunjukkan kumpulan obyek yang memiliki atribut dan operasi yang sama. *Class* digunakan untuk mengimplementasikan *interface*. *Class* digunakan untuk mengabstraksikan elemen-elemen dari sistem yang sedang dibangun. *Class* bisa merepresentasikan baik perangkat lunak maupun perangkat keras, baik konsep maupun benda nyata. Notasi *class* berbentuk persegi panjang berisi 3 bagian: persegi panjang paling atas untuk nama *class*, persegi panjang paling bawah untuk operasi, dan persegi panjang ditengah untuk atribut. Atribut digunakan untuk menyimpan informasi. Nama atribut menggunakan kata benda yang bisa dengan jelas

	merepresentasikan informasi yang tersimpan didalamnya. Operasi menunjukkan sesuatu yang bisa dilakukan oleh obyek dan menggunakan kata kerja.
Interface 	*Interface* merupakan kumpulan operasi tanpa implementasi dari suatu *class*. Implementasi operasi dalam *interface* dijabarkan oleh operasi didalam *class*. Oleh karena itu keberadaan *interface* selalu disertai oleh *class* yang mengimplementasikan operasinya. *Interface* ini merupakan salah satu cara mewujudkan prinsip enkapsulasi dalam obyek.
Interaction 	*Interaction* digunakan untuk menunjukkan baik aliran pesan atau informasi antar obyek maupun hubungan antar obyek. Biasanya *interaction* ini dilengkapi juga dengan teks bernama operation *signature* yang tersusun dari nama operasi, parameter yang dikirim dan tipe parameter yang dikembalikan.
Note 	*Note* digunakan untuk memberikan keterangan atau komentar tambahan dari suatu elemen sehingga bisa langsung terlampir dalam model. *Note* ini bisa disertakan ke semua elemen notasi yang lain.
Dependency	*Dependency* merupakan relasi yang menunjukan bahwa perubahan pada salah satu elemen memberi pengaruh pada elemen lain. Elemen yang ada di bagian tanda panah

	adalah elemen yang tergantung pada elemen yang ada dibagian tanpa tanda panah. Terdapat 2 *stereotype* dari *dependency,* yaitu *include* dan *extend*. *Include* menunjukkan bahwa suatu bagian dari elemen (yang ada digaris tanpa panah) memicu eksekusi bagian dari elemen lain (yang ada di garis dengan panah). *Extend* menunjukkan bahwa suatu bagian dari elemen di garis tanpa panah bisa disisipkan kedalam elemen yang ada di garis dengan panah.
Association	*Association* menggambarkan navigasi antar *class* (*navigation*), berapa banyak obyek lain yang bisa berhubungan dengan satu obyek (*multiplicity* antar *class*) dan apakah suatu *class* menjadi bagian dari *class* lainnya (*aggregation*). *Navigation* dilambangkan dengan penambahan tanda panah di akhir garis. *Bidirectional navigation* menunjukkan bahwa dengan mengetahui salah satu *class* bisa didapatkan informasi dari *class* lainnya. Sementara *UniDirectional navigation* hanya dengan mengetahui *class* diujung garis *association* tanpa panah kita bisa mendapatkan informasi dari *class* di ujung dengan panah, tetapi tidak sebaliknya. *Aggregation* mengacu pada hubungan has-a , yaitu bahwa suatu *class* memiliki *class* lain, misalnya Rumah memiliki *class*

	Kamar.
Generalization 	*Generalization* menunjukkan hubungan antara elemen yang lebih umum ke elemen yang lebih spesifik. Dengan *generalization*, *class* yang lebih spesifik (*subclass*) akan menurunkan atribut dan operasi dari *class* yang lebih umum (*superclass*) atau *subclass* is *superclass* . Dengan menggunakan notasi *generalization* ini, konsep *inheritance* dari prinsip hirarki dapat dimodelkan.
Realization 	*Realization* menunjukkan hubungan bahwa elemen yang ada di bagian tanpa panah akan merealisasikan apa yang dinyatakan oleh elemen yang ada di bagian dengan panah. Misalnya *class* merealisasikan *package*, *component* merealisasikan *class* atau *interface*.

Sumber : (Rosa A.S, M. Shalahuddin; 2011: 123-139)

II.10. Visual Basic.Net

Semenjak *Visual Studio.Net*, *Microsoft* telah banyak melakukan pengembangan dan perubahan pada tampilan *software* ini. Jadi apabila anda sudah terbiasa menggunakan rilis *Visual Basic* sebelumnya, anda harus mulai beradaptasi dengan tampilan baru *Visual Basic*. Pada dasarnya tampilan baru ini memudahkan kita dalam menggunakan *software Visual Basic* (disingkat VB).

Sebelum kita membuat aplikasi baru, ada dua istikah yang perlu kita ketahui dalam *Visual Basic Studio 2008* yaitu :

1. *Project*, merupakan sebutan bagi sebuah *software* yang sedang melalui tahap pembuatan menggunakan *Visual Studio*.
2. *Solution*, adalah kumpulan beberapa buah *project*. (Rahmat Priyanto ; 2009 ; 1-3)

Microsoft Visual Basic Studio 2008 merupakan kelanjutan dari *Micrososft Visual Studio* sebelumnya, yaitu *Visual Studio.Net 2003* yang diproduksi oleh *Micrososft*. Pada bulan Februari tahun 2002 *Microsoft* memproduksi teknologi *.Net Framwork versi 1.0*, Teknologi *.Net* ini didasarkan atas susunan berupa *.Net Framwork*, sehingga setiap produk baru yang terkait dengan teknologi *.Net* akan selalu berkembang mengikuti perkembangan *.Net Framwork*-nya. Pada perkembangan nantinya, mungkin untuk membuat program dengan teknologi *.Net*, dan memungkinkan para pengembang perangkat lunak akan dapat menggunakan lintas sistem operasi, yaitu dapat dikembangkan di sistem operasi *Windows* juga dapat dijalankan pada sistem operasi *Linux*, seperti yang telah dilakukan pada pemrograman *Java* oleh *Sun Microsystem*. Pada saat ini perusahaan-perusahaan sudah banyak meng-*update* aplikasi yang lama yang dibuat dengan *Microsoft Visual Basic 6.0* ke teknologi *.Net* karena kelebihan-kelebihan yang ditawarkan, terutama memungkinkan pengembang perangkat lunak secara cepat mampu membuat program yang *robust*, serta berbasiskan intergrasi ke *internet* yang dikenal dengan *XML Web Service*. (Ketut Darmayuda ; 2009 :1-2)

II.11. Database (Basis Data)

Sistem basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. Pada intinya basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat. (Rosa A.S, M. Shalahuddin; 2011: 44)

DBMS (*Database Management System*) atau dalam bahasa indonesia sering disebut Sistem Manajemen Basis Data adalah suatu sistem aplikasi yang digunakan untuk menyimpan, mengelola, dan menampilkan data. Suatu sistem aplikasi disebut DBMS jika memenuhi persyaratan minimal sebagai berikut ;

a. Menyediakan fasilitas untuk mengelola akses data
b. Mampu menangani integritas data
c. Mampu menangani akses data yang dilakukan secara bersamaan
d. Mampu menangani back-up data (Rosa A.S, M. Shalahuddin; 2011: 45)

Berikut ini adalah 4 macam DBMS versi komersial yang paling banyak digunakan d dunia saat ini, yaitu :

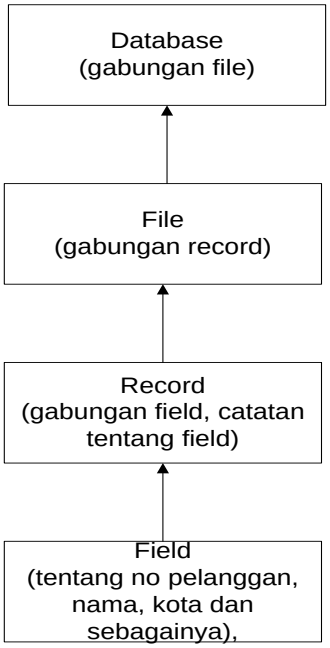
1. *Oracle*
2. *Microsoft SQL Server*
3. *IBM DB2*
4. *Microsoft Access*

Sedangkan DBMS versiopen source yang cukup berkembang dan paling banyak digunakan saat ini adalah sebagai berikut :

1. *MySQL*
2. *PostgreSQL*
3. *Firebird*
4. *Sqlite*

Hampir semua DBMS mengadopsi SQL sebagai bahasa untuk mengelola data pada DBMS. (Rosa A.S, M. Shalahuddin; 2011: 46)

Sistem *database* merupakan gabungan *file* yang membentuk *file* utama (*master file*) yang saling terkait, dalam *database file* merupakan gabungan *record* dan *record* merupakan gabungan dari *field* atau *atribut* dari entitas.



Gambar II.16. Skema Hirarki Data dalam Database
Sumber : (Mardi ; 2011 : 34)

II.11.1. ERD (*Entity Relationship Diagram*)

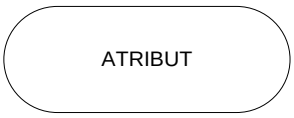
ERD (*Entity Relationship Diagram*) adalah gambar atau diagram yang menunjukkan infomasi dibuat, disimpan, dan digunakan dalam sebuah sistem bisnis. Entitas biasanya menggambarkan jenis informasi yang sama. Dalam entitas digunakan untuk menghubungkan antar entitas yang sekaligus menunjukkan hubungan antar data. Pada akhirnya ERD juga bisa digunakan untuk menunjukkan aturan0aturan bisnis yang ada pada sistem informasi yang akan dibangun.

Elemen-elemen dari ERD adalah sebagai berikut :

1. *Entitas*, biasanya berupa orang, kejadian, atau benda dimana data akan dikumpulkan. Untuk menjadi entitas suatu objek harus menampilkan beberapa kali *event*.

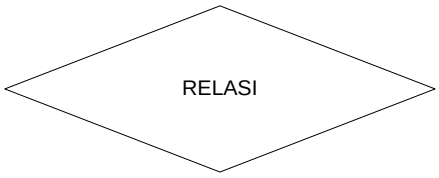
Gambar II.17. Lambang Entity
Sumber : (Kusrini ; 2007 : 21)

2. *Attribut*, informasi yang diambil tentang sebuah *entitas*, hanya digunakan oleh organisasi yang dimasukkan dalam model, nama atribut harus merupakan kata benda, kadang nama *entitas* diletakkan di depan nama *atribut* untuk ketelitian.



Gambar II.18. Lambang Atribut
Sumber : (Kusrini ; 2007 : 22)

3. *Identifier*, satu atau lebih atribut dapat menjadi identifier entitas, yang secara unik mengidentifikasi setiap anggota dari entitas, *identifier* gabungan terdiri dari beberapa atribut, bisa saja artifisial, seperti membuat nomor ID, dan tidak akan dikembangkan sampai fase desain.
4. *Relationship,* hubungan antar entitas, entitas pertama dalam *relationship* disebut entitas induk, entitas kedua disebut entitas anak. *Relationship* harus memiliki nama berupa kata kerja, *relationship* berjalan dua arah.



Gambar II.19. Lambang Relasi
Sumber : (Kusrini ; 2007 : 21)

5. *Kardinalitas,* mengacu pada berapa kali *instance* dari suatu entitas.
6. *Modalitas,* mengacu apakah suatu *instance* dari entitas anak dapat ada tanpa suatu telasi dengan *instance* induk atau tidak. (Hanif Al Fatta ; 2007: 121-127)

II.11.2. Normalisasi

Normalisasi merupakan cara pendekatan dalam membangun desain logika basis data relasional yang tidak secara langsung berkaitan dengan model data, tetapi dengan menerapkan sejumlah aturan dan kriteria standar untuk menghasilkan struktur tabel yang normal. Pada dasarnya desain logika basis data transformasi dari model E-R ke bentuk fisik.

Dalam presfektif normalisasi sebuah *database* dikatakan baik jika setiap tabel yang membentuk basis data sudah berada dalam keadaan normal. Sebuah tabel dikatakan normal jika :

1. Jika ada dekomposisi/penguraian tabel, maka dekomposisinya dijamin aman (*lossless-join decomposition*)
2. Terpeliharanya ketergantungan functional pada saat perubahan data (*depedency preservation*)
3. Tidak melanggar *Boyce Code Normal Form* (BCNF), jika tidak bisa minimal tidak melanggar bentuk normalisasi ketiga. (Kusrini ; 2007 : 39-40)

II.11.3. Bentuk-Bentuk Normalisasi

Adapun bentuk-bentuk dalam normalisasi adalah sebagai berikut :

a. Bentuk tidak normal
b. Bentuk normal tahap pertama (*1st Normal Form*)
c. Bentuk normal tahap kedua (*2nd Normal Form*)
d. Bentuk normal tahap ketiga (*3rd Normal Form*)
e. Bentuk normal tahap keempat dan kelima
f. *Boyce Code Normal Formal* (BCNF) (Kusrini ; 2007 : 41-43)

II.12. SQL Server 2005

SQL Server 2005 adalah sebuah RDBMS (*Relational Database Management System*) yang di-*develop* oleh *Microsoft*, yang digunakan untuk menyimpan dan mengolah data. Pada *SQL Server 2005*, kita bisa melakukan

pengambilan dan modifikasi data yang ada dengan cepat dan efisien. Pada *SQL Server 2005*, kita bisa membuat *object-object* yang sering digunkan pada aplikasi bisnis, seperti mebuat *database*, *table*, *function*, *stored procedure*, *trigger*, dan *view*. Selain *object*, kita juga menjalankan peritah SQL (*Structure Query Languange*) untuk mengambil data. (Cybertron Solution SmitDev Community ; 2010 : 101)