BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Sistem merupakan sekumpulan elemen-elemen yang saling terintegrasi serta melaksanakan fungsinya masing-masing untuk mencapai tujuan yang telah ditetapkan. Karakteristik sistem terdiri dari :

1. Komponen Sistem

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk suatu kesatuan. Komponen-komponen sistem atau elemen-elemen sistem dapat berupa suatu subsistem atau bagian-bagian dari sistem.

2. Batasan Sistem

   Batasan merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batasan sistem ini memungkinkan suatu sistem dipandang suatu kesatuan. Batasan suatu sistem menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

3. Lingkungan Luar Sistem

   Lingkungan luar dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkung luar sistem dapat bersifat menguntungkan dan dapat juga bersifat merugikan sistem tersebut.

4. Penghubung Sistem

   Penghubung merupakan media penghubung antara satu subsistem dengan subsistem lainnya. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem lainnya.

5. Masukan Sistem

   Masukan sistem adalah energi yang dimasukkan ke dalam sistem. Masukan dapat berupa masukan perawatan (*maintanceinput*) dan masukan sinyal (*signalinput*).

6. Keluaran Sistem

   Keluaran sistem adalah hasil energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan.

7. Pengolah Sistem

   Suatu sistem dapat mempunyai suatu bagian pengolah atau sistem itu sendiri sebagai pengolahnya. Pengolah akan mengubah masukan menjadi keluaran.

8. Sasaran Sistem

   Suatu sistem mempunyai tujuan (*goal*) atau sasaran (*objective*). Kalau suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak ada gunanya (Jurnal Saintikom ; Sulindawati ; 2010 : 135).

II.2. Sistem Pakar

Sistem Pakar (*Expert System*) adalah usaha untuk menirukan seorang pakar. Biasanya Sistem Pakar berupa perangkat lunak pengambil keputusan yang mampu mencapai tingkat performa yang sebanding seorang pakar dalam bidang

problem yang khusus dan sempit. Ide dasarnya adalah : kepakaran ditransfer dari seorang pakar (atau sumber kepakaran yang lain) ke komputer, pengetahuan yang ada disimpan dalam komputer, dan pengguna dapat berkonsultasi pada komputer itu untuk suatu nasehat, lalu komputer dapat mengambil inferensi (menyimpulkan, mendeduksi, dll.) seperti layaknya seorang pakar, kemudian menjelaskannya ke pengguna tersebut, bila perlu dengan alasan-alasannya. Sistem Pakar malahan terkadang lebih baik unjuk kerjanya daripada seorang pakar manusia.

Kepakaran (*expertise*) adalah pengetahuan yang ekstensif (meluas) dan spesifik yang diperoleh melalui rangkaian pelatihan, membaca, dan pengalaman. Pengetahuan membuat pakar dapat mengambil keputusan secara lebih baik dan lebih cepat daripada non-pakar dalam memecahkan problem yang kompleks. Kepakaran mempunyai sifat berjenjang, pakar top memiliki pengetahuan lebih banyak daripada pakar yunior.

Tujuan Sistem Pakar adalah untuk mentransfer kepakaran dari seorang pakar ke komputer, kemudian ke orang lain (yang bukan pakar). Proses ini tercakup dalam rekayasa pengetahuan (knowledge engineering) yang akan dibahas kemudian (Rizki Fitriani Maiza ; 2010 : 7).

II.2.1. Manfaat Sistem Pakar

Sangat banyak kemampuan dan mamfaat yang diberikan oleh Sistem Pakar, di antaranya :

1. Meningkatkan *output* dan produktivitas, karena Sistem Pakar dapat bekerja lebih cepat dari manusia.

2. Meningkatkan kualitas, dengan memberi nasehat yang konsisten dan mengurangi kesalahan.
3. Mampu menangkap kepakaran yang sangat terbatas.
4. Dapat beroperasi di lingkungan yang berbahaya.
5. Memudahkan akses ke pengetahuan.
6. Handal. Sistem Pakar tidak pernah menjadi bosan dan kelelahan atau sakit. Sistem Pakar juga secara konsisten melihat semua detil dan tidak akan melewatkan informasi yang relevan dan solusi yang potensial.
7. Meningkatkan kapabilitas sistem terkomputerisasi yang lain. Integrasi Sistem Pakar dengan sistem komputer lain membuat lebih efektif, dan mencakup lebih banyak aplikasi.
8. Mampu bekerja dengan informasi yang tidak lengkap atau tidak pasti. Berbeda dengan sistem komputer konvensional, Sistem Pakar dapat bekerja dengan inofrmasi yang tidak lengkap. Pengguna dapat merespon dengan: “tidak tahu” atau “tidak yakin” pada satu atau lebih pertanyaan selama konsultasi, dan Sistem Pakar tetap akan memberikan jawabannya (Rizki Fitriani Maiza ; 2010 : 8).

II.2.2. Komponen Sistem Pakar

Secara umum, Sistem Pakar biasanya terdiri atas beberapa komponen yang masing-masing berhubungan, diataranya :

1. Basis Pengetahuan

   Berisi pengetahuan yang dibutuhkan untuk memahami, memformulasi, dan memecahkan masalah

2. Mesin Inferensi (*Inference Engine*)

   Merupakan otak dari Sistem Pakar. Juga dikenal sebagai penerjemah aturan (rule interpreter). Komponen ini berupa program komputer yang menyediakan suatu metodologi untuk memikirkan (*reasoning*) dan memformulasi kesimpulan.

3. Papan Tulis (*Blackboard/Workplace*)

   Adalah memori/lokasi untuk bekerja dan menyimpan hasil sementara. Biasanya berupa sebuah basis data.

4. Antarmuka Pemakai (*User Interface*)

   Sistem Pakar mengatur komunikasi antara pengguna dan komputer. Komunikasi ini paling baik berupa bahasa alami, biasanya disajikan dalam bentuk tanya-jawab dan kadang ditampilkan dalam bentuk gambar/grafik. Antarmuka yang lebih canggih dilengkapi dengan percakapan (*voice communication*).

5. Subsistem Penjelasan (*Explanation Facility*)

   Kemampuan untuk menjejak (*tracing*) bagaimana suatu kesimpulan dapat diambil merupakan hal yang sangat penting untuk transfer pengetahuan dan pemecahan masalah. Komponen subsistem penjelasan harus dapat menyediakannya yang secara interaktif menjawab pertanyaan pengguna

6. Sistem Penghalusan Pengetahuan (*Knowledge Refining System*)

   Seorang pakar mempunyai sistem penghalusan pengetahuan, artinya, mereka bisa menganalisa sendiri performa mereka, belajar dari

pengalaman, serta meningkatkan pengetahuannya untuk konsultasi berikutnya (Rizki Fitriani Maiza ; 2010 : 8).

II.2.3. Pembangunan Sebuah Sistem Pakar

Mengembangkan Sistem Pakar dapat dilakukan dengan 2 cara:

1. Membangun sendiri semua komponen di atas, atau
2. Memakai semua komponen yang sudah ada kecuali isi basis pengetahuan.

Tahap-tahap pembangunnan yaitu:

1. Pemilihan Masalah
2. Rekayasa Pengetahuan (*Knowledge Engineering*)
3. Partisipan Dalam Proses Pengembangan
4. Akuisisi Pengetahuan (Rizki Fitriani Maiza ; 2010 : 9).

II.2.4. Mesin Inferensi (*Inference Engine*) Sistem Pakar

Inferensi digunakan dalam sistem pakar untuk memperoleh informasi terbaru dari informasi yang sudah ada. Diataranya :

1. *Forward Chaining*

   Adalah strategi inferensi yang dimulai dengan sekumpulan fakta, fakta baru yang diperoleh dengan menggunakan rule, dimana alasan yang digunakan sesuai dengan fakta yang ada, dan melanjutkan proses ini sampai goal diraih atau sampai tidak ada rule selanjutnya yang

mempunyai alasan yang sesuai dengan fakta yang ada maupun fakta yang diketahui

2. *Backwad Chaining*

   Adalah strategi inferensi yang diperoleh untuk membuktikan suatu hipotesis dengan dukungan informasi (Rizki Fitriani Maiza ; 2010 : 10).

II.3. Smartphone

Smartphone atau bisa disebut dengan telepon pintar/cerdas sudah menjadi sebuah kebutuhan bagi sekian orang di dunia ini sebagai penunjang aktivitas kerja maupun sekedar lifestyle atau gaya hidup. Telepon cerdas (*smartphone*) adalah telepon genggam yang mempunyai kemampuan tingkat tinggi, kadang-kadang dengan fungsi yang menyerupai komputer. Belum ada standar pabrik yang menentukan arti telepon cerdas. Bagi beberapa orang, telepon pintar merupakan telepon yang bekerja menggunakan seluruh perangkat lunak sistem operasi yang menyediakan hubungan standar dan mendasar bagi pengembang aplikasi. Bagi yang lainnya, telepon cerdas hanyalah merupakan sebuah telepon yang menyajikan fitur canggih seperti surel (surat elektronik), internet dan kemampuan membaca buku elektronik (*e-book*) atau terdapat papan ketik (baik sebagaimana jadi maupun dihubung keluar) dan penyambung VGA. Dengan kata lain, telepon cerdas merupakan komputer kecil yang mempunyai kemampuan sebuah telepon. Pertumbuhan permintaan akan alat canggih yang mudah dibawa kemana-mana membuat kemajuan besar dalam pemroses, ngingatan, layar dan sistem operasi

yang diluar dari jalur telepon genggam sejak beberapa tahun ini (Nekie Jocom ; 2013)

II.4. Metode *Certainty Factor*

Certainty factor (CF) menunjukkan ukuran kepastian terhadap suatu fakta atau aturan. Faktor kepastian ini merupakan bentuk penggabungan kepercayaan dan ketidakpercayaan dalam suatu bilangan tunggal. *Certainty Theory* ini diusulkan oleh Shortliffe dan Buchanan pada tahun 1975 untuk mengakomadasi ketidakpastian pemikiran (*inexact reasoning*) seorang pakar. Teori ini berkembang bersamaan dengan pembuatan sistem pakar MYCIN. Team pengembang MYCIN mencatat bahwa tim ahli sering kali menganalisa informasi yang ada dengan ungkapan seperti misalnya: mungkin, kemungkinan besar, hampir pasti. Untuk mengakomodasi hal ini tim MYCIN menggunakan *Certainty factor* (CF) guna menggambarkan tingkat keyakinan pakar terhadap masalah yang sedang dihadapi.

Beberapa *evidence* dapat dikombinasikan untuk menentukan CF dari suatu hipotesis. Untuk sistem ini, tingkat kepastian sistem terhadap kesimpulan yang diperoleh dihitung berdasarkan nilai probabilitas penyakit karena adanya evident/gejala tertentu. Faktor kepastian bukanlah suatu peluang, tetapi merupakan ukuran tingkat kepercayaan terhadap suatu evidensi. CF mempresentasikan tingkat kepercayaan bahwa suatu evidensi adalah benar.

Certainty Theory ini diusulkan oleh Shortliffe dan Buchanan pada tahun 1975 untuk mengakomadasi ketidakpastian pemikiran (*inexact reasoning*) seorang

pakar. Teori ini berkembang bersamaan dengan pembuatan sistem pakar MYCIN. Team pengembang MYCIN mencatat bahwa tim ahli sering kali menganalisa informasi yang ada dengan ungkapan seperti misalnya: mungkin, kemungkinan besar, hampir pasti. Untuk mengakomodasi hal ini tim MYCIN menggunakan *certainty factor* (CF) guna menggambarkan tingkat keyakinan pakar terhadap masalah yang sedang dihadapi. Secara umum, rule direpresentasikan dalam bentuk sebagai berikut:

IF E1 [AND/OR] E2 [AND/O R] … E_n

THEN H (CF = CF)

II.4.1. Model Perhitungan *Certainty Factor* dengan *Rule*

Certainty Factor (CF) menunjukkan ukuran kepastian terhadap suatu fakta atau aturan. Faktor kepastian ini merupakan bentuk penggabungan kepercayaan dan ketidakpercayaan dalam suatu bilangan tunggal. Berikut notasi faktor kepastian:

CF(Pk, G) = MB(Pk,G) - MD(Pk, G)

Beberapa *evidence* dapat dikombinasikan untuk menentukan CF dari suatu hipotesis. Untuk sistem ini, tingkat kepastian sistem terhadap kesimpulan yang diperoleh dihitung berdasarkan nilai probabilitas penyakit karena adanya evident/gejala tertentu (Bain Khusnul, 2010 : 13). Jika ada gejala dan penyakit sebagai *hipothesis* maka tingkat kepastian diformulasikan sebagai *CF(Pk,G)*:

$$MB(Pk,G)=\begin{cases}1 & ,P(Pk)=1\\ \dfrac{\max[P(Pk|G),P(Pk)]-P(Pk)}{\max[1,0]-P(Pk)} & ,yanglain\end{cases}$$

$$MD(Pk,G)=\begin{cases}1 & ,P(Pk)=1\\ \dfrac{\min[P(Pk|G),P(Pk)]-P(Pk)}{\min[1,0]-P(Pk)} & ,yanglain\end{cases}$$

Di mana:

P(Pk) : probabilitas kerusakan Pk

G : Gejala

CF : Certainty Factor (faktor Kepastian) dalam hipotesis H yang dipengaruhi oleh evidence (fakta) E.

MB : Measure of Belief (tingkat keyakinan), merupakan ukuran kepercayaan dari hipotesis H dipengaruhi oleh evidence (fakta) E.

MD : Measure of Disbelief (tingkat ketidakyakinan) merupakan ukuran ketidakpercayaan hipotesis H dipengaruhi oleh fakta E.

H : Hipotesa atau konklusi yang dihasilkan

Jika ada kaidah lain termasuk dalam hipotesis yang sama tetapi berbeda dalam faktor kepastian, maka perhitungan faktor kepastian dari kaidah yang sama dihitung dari penggabungan fungsi untuk faktor kepastian yang didefinisikan sebagai berikut:

$$CF_{combine}(CF_1,CF_2)=\begin{cases}CF_1+CF_2(1-CF_1) & kedua-duanya>0\\ \dfrac{CF_1+CF_2}{1-\min(|CF_1|,|CF_2|)} & salah\ satu<0\\ CF_1+CF_2(1-CF_1) & kedua-duanya<0\end{cases}$$

Di mana, $CF_{combine}$ digunakan bergantung pada apakah faktor kepastian positif atau negatif.

Evidensi tidak pasti diperoleh dengan menggali dari hasil wawancara dengan pakar. Nilai CF(Rule) didapat dari interpretasi term dari pakar menjadi nilai MD/MB tertentu. Suatu sistem yang menerapkan *inexact* reasoning, pertama-tama harus menemukan cara untuk mempresentasikan *uncertain evidence*. Pendekatan di atas mengubah bentuk formal dari suatu peluang P(E) dengan CF(E) yang berupa nilai MD/MB.

Faktor kepastian bukanlah suatu peluang, tetapi merupakan ukuran tingkat kepercayaan terhadap suatu evidensi. CF mempresentasikan tingkat kepercayaan bahwa suatu evidensi adalah benar (Bain Khusnul, 2010 : 13).

II.5. Database

Secara sederhana *database* (basis data/pangkalan data) dapat diungkapkan sebagai suatu pengorganisasian data dengan bantuan komputer yang memungkinkan data dapat diakses dengan mudah dan cepat. Pengertian akses dapat mencakup pemerolehan data maupun pemanipulasian data seperti menambah serta menghapus data. Dengan memanfaatkan komputer, data dapat disimpan dalam media pengingat yang disebut *harddisk.* Dengan menggunakan media ini, keperluan kertas untuk menyimpan data dapat dikurangi. Selain itu, data menjadi lebih cepat untuk diakses terutama jika dikemas dalam bentuk *database* (Agustinus Mujilan ; 2012 : 23)

II.6. ERD *(Entity Relationship Diagram)*

ERD adalah suatu diagram untuk menggambarkan desain konseptual dari model konseptual suatu basis data relasional. ERD juga merupakan gambaran yang merelasikan antara objek yang satu dengan objek yang lain dari objek di dunia nyata yang sering dikenal dengan hubungan antar entitas. Sebagai contoh jika membuat ERD dari sistem perpustakaan maka bahan sebagai objek ERD bisa berupa anggota, buku, peminjam, pengembalian dan sebagainya. ERD terdiri dari 4 komponen utama, yaitu (Robi Yanto ; 2016 : 32) :

Tabel II.1. Simbol ERD

Notasi	Keterangan
	**Entitas**, adalah suatu objek yang dapat diidentifikasi dalam lingkungan pemakai.
	**Relasi**, menunjukkan adanya hubungan di antara sejumlah entitas yang berbeda.
	**Atribut**, berfungsi mendeskripsikan karakter entitas (atribut yg berfungsi sebagai key diberi garis bawah)
	Garis, sebagai penghubung antara relasi dengan entitas, relasi dan entitas dengan atribut.

(Sumber : Robi Yanto ; 2016 : 32)

II.7. Kamus Data

Kamus data (*data dictionary*) mencakup definisi-definisi dari data yang disimpan di dalam basis data dan dikendalikan oleh sistem manajemen basis data. Struktur basis data yang dimuat dalam kamus data adalah kumpulan dari seluruh

definisi *field*, definisi tabel, relasi tabel dan hal-hal lainnya. Nama *field* data, jenis data (seperti teks atau angka atau tanggal), nilai-nilai yang valid untuk data, dan karakteristik-karakteristik lainnya akan disimpan dalam kamus data. Perubahan-perubahan pada struktur data hanya dilakukan satu kali, di dalam kamus data program-program aplikasi yang mempergunakan data tidak akan ikut terpengaruh (Raymond ; 2010 : 171).

II.8. Teknik Normalisasi

Normalisasi adalah teknik perancangan yang banyak digunakan sebagai pemandu dalam merancang basis data relasional. Pada dasarnya, normalisasi adalah proses dua langkah yang meletakkan data dalam bentuk tabulasi dengan menghilangkan kelompok berulang lalu menghilangkan data yang terduplikasi dari tabel rasional.

Teori normalisasi didasarkan pada konsep bentuk normal. Sebuah tabel relasional dikatakan berada pada bentuk normal tertentu jika tabel memenuhi himpunan batasan tertentu. Ada lima bentuk normal yang tekah ditemukan.

II.8.1. Bentuk-bentuk Normalisasi

1. **Bentuk normal tahap pertama (1” Normal Form)**

   Contoh yang kita gunakan di sini adalah sebuah perusahaan yang mendapatkan barang dari sejumlah pemasok. Masing-masing pemasok berada pada satu kota. Sebuah kota dapat mempunyai lebih dari satu pemasok dan masing-masing kota mempunyai kode status tersendiri.

2. **Bentuk normal tahap kedua (2nd normal form)**

   Definisi bentuk normal kedua menyatakan bahwa tabel dengan kunci utama gabungan hanya dapat berada pada 1NF, tetapi tidak pada 2NF. Sebuah tabel relasional berada pada bentuk normal kedua jika dia berada pada bentuk normal kedua jika dia berada pada 1NF dan setiap kolom bukan kunci yang sepenuhnya tergantung pada seluruh kolom yang membentuk kunci utama.

3. **Bentuk normal tahap ketiga (3rd normal form)**

   Bentuk normal ketiga mengharuskan semua kolom pada tabel relasional tergantung hanya pada kunci utama. Secara definisi, sebuah tabel berada pada bentuk normal ketiga (3NF) jika tabel sudah berada pada 2NF dan setiap kolom yang bukan kunci tidak tergantung secara transitif pada kunci utamanya.

4. **Boyce Code Normal Form (BCNF)**

   Setelah 3NF, semua masalah normalisasi hanya melibatkan tabel yang mempunyai tiga kolom atau lebih dan semua kolom adalah kunci. Banyak praktisi berpendapat bahwa menempatkan entitas pada 3NF sudah cukup karena sangat jarang entitas yang berada pada 3NF bukan merupakan 4NF dan 5NF.

5. **Bentuk Normal Tahap Keempat dan Kelima**

   Sebuah tabel relasional berada pada bentuk normal keempat (4NF) jika dia dalam BCNF dan semua ketergantungan multivalue

merupakan ketergantungan fungsional. Bentuk normal keempat (4NF) didasarkan pada konsep ketergantungan multivalue (MVD).

Sebuah tabel berada pada bentuk normal kelima (5NF) jika ia tidak dapat mempunyai dekomposisi lossless menjadi sejumlah tabel lebih kecil. Empat bentuk normal pertama berdasarkan pada konsep ketergantungan fungsional, sedangkan bentuk normal kelima berdasarkan pada konsep ketergantungan gabungan (*join dependence*) (Janner Simarmata, 2010 : 76).

II.9. Pengertian Visual Basic

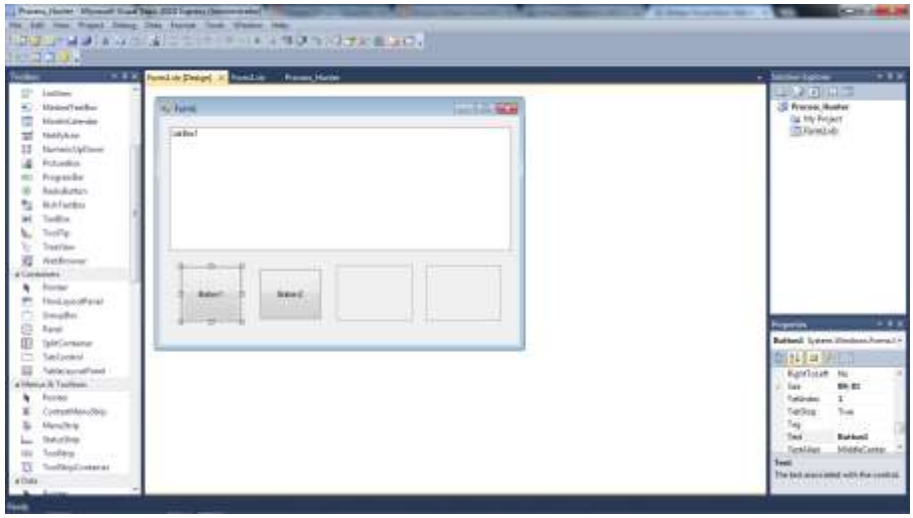
Visual basic dibuat oleh Microsoft, merupakan salah satu bahasa pemrograman beriorientasi objek yang mudah dipelajari. Selain menawarkan kemudahan, Visual Basic juga cukup andal untuk digunakan dalam pembuatan berbagai aplikasi, terutama aplikasi *database.* Visual basic merupakan bahasa pemrograman *event drive,* di mana program aplikasi yang dapat berupa kejadian atau *event*, misalnya ketika *user* mengklik tombol atau menekan enter. Jika kita membuat aplikasi dengan Visual Basic maka kita akan mendapatkan *file* yang menyusun aplikasi tersebut, yaitu :

1. File Project (*.vbp)

   File ini merupakan kumpulan dari aplikasi yang kita buat. File project bisa berupa file *.frm, *.dsr atau file lainnya.

2. File Form (*.frm)

File ini merupakan file yang berfungsi untuk menyimpan informasi tentang bentuk form maupun *interface* yang kita buat (Edy Winarno ; 2010 : 83).



Gambar II.1. Tampilan VB.Net
(Sumber : Edy Winarno ; 2010 : 83)

II.10. Pengertian SQL Server 2008

SQL Server 2008 adalah sebuah terobosan baru dari Microsoft dalam bidang *database*. SQL Server adalah DBMS (Database Management *System*) yang dibuat oleh Microsoft untuk ikut berkecimpung dalam persaingan dunia pengolahan data menyusul pendahulunya seperti IBM dan Oracle. SQl Server 2008 dibuat pada saat kemajuan dalam bidang *hardware* sedemikian pesat. Oleh karena itu sudah dapat dipastikan bahwa SQL Server 2008 membawa beberapa terobosan dalam bidang pengolahan dan penyimpanan data.

Microsoft merilis SQL Server 2008 dalam beberapa versi yang disesuaikan dengan segment-segment pasar yang dituju. Versi-versi tersebut adalah sebagai berikut. Menurut cara pemrosesan data pada prosesor maka Microsoft mengelompokkan produk ini berdasarkan 2 jenis yaitu :

1. Versi 32-bit(x86), yang biasanya digunakan untuk komputer dengan single prosesor (Pentium 4) atau lebih tepatnya prosesor 32 bit dan sistem operasi Windows XP.
2. Versi 64-bit(x64), yang biasanya digunakan untuk komputer dengan lebih dari satu prosesor (Misalnya Core 2 Duo) dan system operasi 64 bit seperti Windows XP 64, Vista, dan Windows 7.

Sedangkan secara keseluruhan terdapat versiversi seperti berikut ini:

1. Versi Compact, ini adalah versi "Tipis" dari semua versi yang ada. Versi ini seperti versi desktop pada SQL Server 2000. Versi ini juga digunakan pada handled drvice seperti Pocket PC, PDA, SmartPhone, Tablet PC.
2. Versi Express, ini adalah versi "Ringan" dari semua versi yang ada(tetapi versi ini berbeda dengan versi compact) dan paling cocok untuk latihan para pengembang aplikasi. Versi ini memuat Express Manager standar, integrasi dengan CLR dan XML (Wenny Widya ; 2012 : 3)

Gambar II.2. Tampilan SQL Server
(Sumber : Wenny Widya ; 2012 : 3)

II.11. UML (*Unified Modeling Language*)

Menurut Windu Gata (2013 : 4) Hasil pemodelan pada OOAD terdokumentasikan dalam bentuk *Unified Modeling Language* (UML). UML adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak.

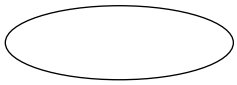
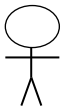
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut :

1. *Use case* Diagram

   Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram, yaitu :

Tabel II.2. Simbol *Use Case*

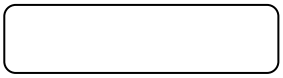
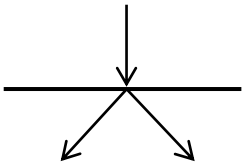
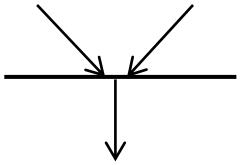
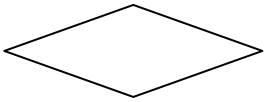
Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki control terhadap *use case*.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	Include, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	Extend, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(Sumber : Windu Gata, 2013 : 4)

2. Diagram Aktivitas (*Activity Diagram*)

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram*, yaitu :

Tabel II.3. Simbol *Activity Diagram*

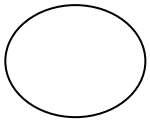
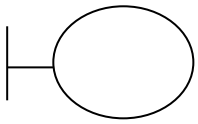
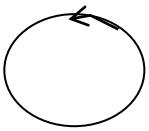
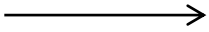
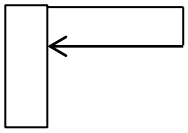
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true*, *false.*
	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

(Sumber : Windu Gata, 2013 : 6)

3. Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram*, yaitu :

Tabel II.4. Simbol *Sequence Diagram*

Gambar	Keterangan
	*Entity Class,* merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class,* berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak.
	*Control class,* suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	*Message,* simbol mengirim pesan antar *class*.
	*Recursive,* menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation, activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Windu Gata, 2013 : 7)

4. *Class Diagram* (Diagram Kelas)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng jawab entitas yang menentukan perilaku sistem.

Class diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class diagram* secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti.

Tabel II.5. *Multiplicity Class Diagram*

Multiplicity	**Penjelasan**
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

(Sumber : Windu Gata, 2013 : 9)