BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terdahulu

Menurut konsep dasar sistem pakar dari Ashari (2015) mengenai Penerapan Sistem Pakar Untuk Mengidentifikasi Masalah Kehamilan dan pada sistem pakar dalam mengidentifikasimasalah kehamilan dilakukan dengan metode *demspter shafer* dengan berdasarkan rule sesuai keahlian para pakardengan melakukan pengukuran kekuatan *evidence* dalammendukung suatu himpunan proposisi dan dilakukan olehsistem komputer sebagai hasil perancangan diagram atau alur sistem.Hasil penelitian yang di dapat adalah sistem pakar menggunakan metode *dempster shafer* mampu mengidentifikasi masalah kehamilan.

Menurut konsep dasar sistem pakar dari Istiqomah, dkk (2013) mengenaisistem pakar untuk mendiagnosa penyakit saluran pencernaan menggunakan metode *dempster shafer*,berdasarkan dari hasil penelitian yang sudah dilakukan, kesimpulan yang dapat diambil adalah perangkat lunak yang dihasilkan mampu mendiagnosa penyakit saluran pencernaan pada manusia berdasarkan gejala yang dimasukkan dan dapat memberikan data mengenai penyakit yang diderita berupa nama dan definisi penyakit, penyebab, solusi yang dilengkapi dengan nilai persentase dari penyakit tersebut.

II.2. Sistem Pakar

Menurut Istiqomah dan Fadlil, (2013 : 34), Sistem pakar (*Expert System*)merupakan solusi AI bagi masalah pemrograman pintar (*Intelligent*). Profesor Edward Feigenbaum dari *Stanford University* yang merupakan *pionir* dalam teknologi sistem pakar mendefenisikan sistem pakar sebagai sebuah program komputer pintar (*intelligentcomputer program*) yang memanfaatkan pengetahuan (*knowledge*) dan prosedur inferensi (*inference procedure*) untuk memecahkan masalah yang cukup sulit hingga membutuhkan keahlian khusus dari manusia.

Dengan kata lain, sistem pakar adalah sistem komputer yang ditujukan untuk meeniru semua aspek (*emulates*) kemampuan pengambilan keputusan (*decision making*) seorang pakar. Sistem pakar memanfaatkan secara maksimal pengetahuan khusus selayaknya seorang pakar untuk memecahkan masalah.

Secara umum, sistem pakar (*Expert System*) adalah sistem yang berusaha mengadopsi pengetahuan manusia ke komputer, agar komputer dapat menyelesaikan masalah seperti yang biasa dilakukan oleh para ahli.Dengan sistem pakar ini, orang awam pun dapat menyelesaikan masalah yang cukup rumit yang sebenarnya hanya dapat diselesaikan dengan bantuan para ahli.

II.2.1. Kelebihan Sistem Pakar

Menurut Rosnelly (2012),Sistem pakar memiliki beberapa fitur menarik yang merupakan kelebihannya, seperti :

1. Meningkatnya ketersediaan (*increased availability*). Kepakaran atau keahlian menjadi tersedia dalam sistem komputer. Dapat dikatakan bahwa sistem pakar merupakan produksi kepakaran secara masal (*massproduction*).
2. Mengurangi biaya (*reduced cost*). Biaya yang diperlukan untuk menyediakan keahlian per satu orang *user* menjadi berkurang.
3. Mengurangi bahaya (*reduced danger*). Sistem pakar dapat digunakan di lingkungan yang mungkin berbahaya bagi manusia.
4. Permanen (*permanence*). Sistem pakar dan pengetahuan yang terdapat di dalamnya bersifat lebih permanen dibandingkan manusia yang dapat merasa lelah, bosan, dan pengetahuannya hilang saat sang pakar meninggal dunia.
5. Keahlian *multiple* (*multiple expertise*). Pengetahuan dari beberapa pakar dapat dimuat ke dalam sistem dan bekerja secara simultan dan kontinyu menyelesaikan suatu masalah setiap saat. Tingkat keahlian atau pengetahuan yang digabungkan dari beberapa pakar dapat melebihi pengetahuan yang digabungkan dari beberapa pakar dapat melebihi pengetahuan satu orang pakar.
6. Meningkatkan kehandalan (*increased reliability*). Sistem pakar meningkatkan kepercayaan dengan memberikan hasil yang benar sebagai alternatif pendapat dari seorang pakar atau sebagai penengah jika terjadi konflik antara beberapa pakar. Namun hal tersebut tidak berlaku jika sistem dibuat oleh salah seorang pakar, sehingga akan selalu sama dengan pendapat pakar tersebut kecuali jika sang pakar melakukan kesalahan yang mungkin terjadi pada saat tertekan atau stres.

7. Penjelasan (*explanation*). Sistem pakar dapat menjelaskan detail proses penalaran (*reasoning*) yang dilakukan hingga mencapai suatu kesimpulan. Seorang pakar mungkin saja terlalu lelah, tidak bersedia atau tidak mampu melakukannya setiap waktu. Hal ini akan meningkatkan tingkat kepercayaan bahwa kesimpulan yang dihasilkan adalah benar.
8. Respon yang cepat (*fast response*). Respon yang cepat atau *real time* diperlukan pada beberapa aplikasi. Meskipun bergantung pada *hardware* dan *software* yang digunakan, namun sistem pakar relatif memberikan respon yang lebih cepat dibandingkan seorang pakar.
9. Stabil, tidak emosional, dan memberikan respon yang lengkap setiap saat (*steady, unemotional, and complete response at all times*). Karakteristik ini diperlukan pada situasi *real-time* dan keadaan darurat (*emergency*) ketika seorang pakar mungkin tidak berada pada kondisi puncak disebabkan oleh stress atau kelelahan.
10. Pembimbing pintar (*intelligent tutor*). Sistem pakar dapat berperan sebagai *intelligent tutor* dengan memberikan kesempatan pada *user* untuk menjalankan contoh program dan menjelaskan proses *reasoning* yang dilakukan.

Basis data cerdas (*intelligent database*).Sistem pakar dapat digunakan untuk mengakses basis data secara cerdas.

II.2.2. Konsep Umum Sistem Pakar

MenurutSinaga dan Sembiring (2016 : 96),Konsep-konsep dasar dari sebuah sistem pakar adalah :

1. Keahlian (*Expertise*)

Keahlian merupakan pengetahuan khusus yang dimiliki oleh seseorang melalui latihan, belajar, serta pengalaman-pengalaman yang dialami pada suatu bidang tertentu dalam jangka waktu yang cukup lama.Dengan pengetahuan tersebut seorang pakar dapat memberikan keputusan yang lebih baik dan cepat dalam menyelesaikan suatu permasalahan yang sulit.

2. Ahli atau pakar (*Expert*)

Seorang pakar harus memiliki kemampuan menyelesaikan permasalahan pada bidang tertentu yang ditanganinya, kemudianmemberikan penjelasan mengenai hasil dan kaitannya dengan permasalahan yang ada.Untuk meniru kepakaran seorang manusia, perlu dibangun sebuah sistem komputer yang menunjukan seluruh karakteristik tersebut.Namun hingga saat ini, pekerjaan dibidang sistem pakar terfokus pada aktifitas penyelesaian masalah dan memberikan penjelasan mengenai solusinya.

3. Memindahkan Keahlian (*Transfering Expertise*)

Tujuan dari sistem adalah memindahkan keahlian yang dimiliki oleh seorang pakar ke dalam sebuah sistem komputer.

4. Kesimpulan (*Inference*):

Keistimewaan dari sistem pakar adalah kemampuannya dalam memberikan saran, yaitu dengan menempatkan keahlian ke dalam basis pengetahuan (*Knowledge Base*) dan membuat program yang mampu mengakses basis pengetahuan sehinggga sistem dapat memberikan kesimpulan.Kesimpulan dibentuk di dalam komponen yang dinamakan mesin pengambil kesimpulan (*Inference Engine*), dimana berisi aturan-aturan untuk menyelesaikan masalah.

5. Aturan (*Rule*)

Umumnya sistem pakar adalah sistem berbasis aturan, yaitu pengetahuan yang terdiri dari aturan-aturan sebagai prosedur penyelesaian masalah.Pengetahuan tersebut digambarkan sebagai suatu urutan seri dari kaidah- kaidah yang sudah dibuat.

6. Kemampuan Penjelasan (*Explanation Capability*)

Keistimewaan lain dari sistem pakar adalah kemampuannya dalam memberikan saran atau rekomendasi serta menjelaskan mengapa tindakan tertentu tidak dianjurkan. Pemberian penerangan dan pendapat ini dilakukan dalam suatu subsistem yang dinamakan subsistem penjelasan (*explanation subsystem*).

II.2.3. Elemen Manusia Pada Sistem Pakar

Menurut Rosnelly (2012), Sistem pakar tidak lepas dari elemen manusia yang terkait di dalamnya. Personil yang terkait dengan sistem pakar ada 4 yaitu :

1. Pakar (*expert*)
2. Pembangun pengetahuan (*knowledge engineer*)
3. Pembangunan sistem (*system engineer*)
4. Pemakai (*user*)

Paling tidak terdapat dua komponen orang atau lebih yang berpartisipasi dalam pembangunan dan penggunaan sistem pakar, yakni sedikitnya seorang pembangun pengetahuan dan seorang pakar.

II.2.3.1. Pakar

Menurut Rosnelly (2012), Pakar adalah seorang individu yang memiliki pengetahuan khusus, pemahaman, pengalaman, dan metode-metode yang digunakan untuk memecahkan persoalan dalam bidang tertentu.

II.2.3.2. Pembangun/ Pembuat Pengetahuan

Menurut Rosnelly (2012), Pembuat pengetahuan memiliki tugas utama menerjemahkan dan merepresentasikan pengetahuan yang diperoleh dari pakar, baik berupa pengalaman pakar dalam menyelesaikan masalah maupun sumber terdokumentasi lainnya ke dalam bentuk yang bisa diterima oleh sistem pakar. Dalam hal ini pembangunan pengetahuan (*knowledge engineer*) menginterpretasikan dan merepresentasikan pengetahuan yang diperoleh dalam bentuk jawaban-jawaban atas pertanyaan-pertanyaan yang diajukan pada pakar atau pemahaman, penggambaran analogis, sistematis, konseptual yang diperoleh dari membaca beberapa dokumen cetak seperti *text book*, jurnal, makalah, dan sebagainya. Kurangnya pengalaman *knowledgeengineer* merupakan kesulitan utama dalam mengkonstruksi sistem pakar. Untuk mengatasi hal tersebut, perancang sistem pakar menggunakan *tools* komersial. (Seperti pada editor-editor khusus maupun *logic debuggers*) dan usahanya akan dipusatkan pada pembangunan mesin inferensi.

II.2.3.3. Pembangun/ Pembuat Sistem

Menurut Rosnelly (2012), Pembangunan sistem adalah orang yang bertugas untuk merancang antarmuka pemakai sistem pakar, merancang

pengetahuan yang sudah diterjemahkan oleh pembangun pengetahuan ke dalam bentuk yang sesuai dan dapat diterima oleh sistem pakar dan mengimplementasikannya ke dalam mesin inferensi. Selain hal tersebut, pembangun sistem juga bertanggung jawab apabila sistem pakar akan diintegrasikan dengan sistem komputerisasi lain. Alat pembangun (*tool builder*) dapat dipakai untuk menyajikan atau membangun *tool* yang spesifik. Penjual (*vendor*) dapat memberikan *tool* dan saran, staf pendukung dapat memberikan saran dan bantuan secara teknis dalam proses pembangunan sistem pakar.

II.2.3.4. Pengguna

Menurut Rosnelly (2012), Banyak sistem berbasis komputer mempunyai susunan pengguna tunggal. Hal ini berbedaa jauh dengan sistem pakar yang memungkinkan mempunyai beberapa kelas pengguna.Pengguna mungkin tidak terbiasa dengan komputer dan mungkin pada domain masalah.Bagaimanapun juga, banyak solusi permasalahan menjadi lebih baik dan kemungkinan lebih murah dan keputusan yang cepat bila menggunakan sistem pakar.Pakar dan pembangun sistem harus mengantisipasi kebutuhan-kebutuhan pengguna dan membuat batasan-batasan ketika mendesain sistem pakar.

II.2.4. Struktur Sistem Pakar

Menurut Rosnelly (2012), Komponen yang terdapat dalam struktur sistem pakar ini adalah *knowledge base* (*rules*), *inference engine*, *working memory*, *explanation facility*, *knowledge acquisition facility*, *user interface*.

1. *Knowledge Base* (Basis Pengetahuan)

Basis pengetahuan mengandung oengetahuan untuk pemahaman, formulasi, dan penyelesaian masalah.Komponen sistem pakar disusun atas dua elemen dasar, yaitu fakta dan aturan. Fakta merupakan informasi tentang objek dalam area permasalahan tertentu, sedangkan aturan merupakan informasi tentang cara bagaimana memperoleh fakta baru dari fakta yang telah diketahui.

2. *Inference Engine* (Mesin Inferensi)

   Mesin inferensi merupakan otak dari sebuah sistem pakar dan dikenal juga dengan sebutan *control structure* (struktu kontrol) atau *rule interpreter* (dalam sistem pakar berbasis kaidah). Komponen ini mengandung mekanisme pola pikir dan penalaran yang digunakan oleh pakar dalam menyelesaikan suatu masalah. Mesin inferensi disini adalah *processor* pada sistem pakar yang mencocokkan bagian kondisi dari *rule* yang tersimpan di dalam *knowledge base* dengan fakta yang tersimpan di *working memory*.

3. *Working Memory*

   Berguna untuk menyimpan fakta yang dihasilkan oleh *inference engine* dengan penambahan parameter berupa derajat kepercayaan atau dapat juga dikatakan sebagai global *database* dari fakta yang digunakan oleh *rule-rule* yang ada.

4. *Explanation Facility*

   Menyediakan kebenaran dari solusi yang dihasilkan kepada *user* (*reasoning chain*).

5. *Knowledge Acquistion Facility*

Meliputi proses pengumpulan, pemindahan dan perubahan dari kemampuan pemecahan masalah seorang pakar atau sumber pengetahuan terdokumentasi ke program *computer*, yang bertujuan untuk memperbaiki atau mengembangkan basis pengetahuan.

6. *User Interface*

   Mekanisme untuk memberi kesempatan kepada *user* dan sistem pakar untuk berkomunikasi.Antar muka menerima informasi dari pemakai dan mengubahnya ke dalam bentuk yang dapat diterima oleh sistem.Selain itu antarmuka menerima informasi dari sistem dan menyajikannya ke dalam bentuk yang dapat dimengerti oleh pemakai.

II.2.5. Karakteristik Sistem Pakar

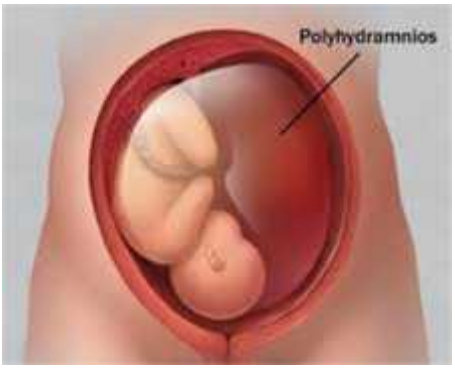
Menurut Rosnelly (2012), Sistem pakar umumnya dirancang untuk memenuhi beberapa karakteristik umum berikut ini :

1. Kinerja sangat baik (*high performance*). Sistem harus mampu memberikan respon berupa saran (*advice*) dengan tingkat kualitas yang sama dengan seorang pakar atau melebihinya.
2. Waktu respon yang baik (*adequate respon time*). Sistem juga harus mampu bekerja dalam waktu yang sama baiknya (*reasonable*) atau lebih cepat dibandingkan dengan seorang pakar dalam menghasilkan keputusan. Hal ini sangat penting terutama pada sistem waktu nyata (*real-time*).

3. Dapat diandalkan (*good reliability*). Sistem harus dapat diandalkan dan tidak mudah rusak/ *crash*.
4. Dapat dipahami (*understandable*). Sistem harus mampu menjelaskan langkah-langkah penalaran yang dilakukannya seperti seorang pakar.
5. Fleksibel (*flexibility*). Sistem harus menyediakan mekanisme untuk menambah, mengubah, dan menghapus pengetahuan.

II.3. Penyakit *Hidramnion*

Menurut Huda (2013 : 4), *Hidramnion*yaitu keadaan dimana banyaknya air ketuban melebihi 2000 cc. Menurut Merzalia (2012 : 41), Pada keadaan normal banyak air ketuban dapat mencapai 1000 cc untuk kemudian menurun lagi setelah minggu ke 38 sehingga hanya tinggal beberapa cc saja.Gejala yang muncul ialah ukuran uterus lebih besar dibanding yang seharusnya, identifikasi janin dan bagian janin melalui pemeriksaan *palpasi* sulit dilakukan, balotemen janin jelas dan detak jantung janin sulit di dengar.Penyebab dari hidramnion ini antara lain gangguan kesehatan janin, ibu hamil mengidap diabetes, seorang ibu mengandung anak kembar, infeksi kongenital, penumpukan cairan, janin mengidap anemia, seorang ibu hamil menggunakan obat terlarang dan metabolisme seorang ibu hamil tidak normal.



Gambar II.1. *Hydramnion*

Menurut Huda (2013: 9).,Hidramnion terjadi bila produksi air ketuban bertambah, bila pengaliran ketuban terganggu atau kedua-duanya. Dicurigai air ketuban dibentuk dari sel-sel amnion.Ditambah oleh air seni janin dan cairan otak pada anense falus.Air ketuban yang dibentuk, secara rutin dikeluarkan dan diganti dengan yang baru. Salah satu cara pengeluaran ialah ditelan oleh janin, diabsorpsi oleh usus kemudian dialirkan ke plasenta untuk akhirnya masuk peredaran darah ibu. Ekskresi air ketuban akan terganggu bila janin tidak bisa menelan seperti pada atresia esophagus atau tumortumor plasenta. Hidramnion dapat memungkinkan ketegangan rahim meningkat, sehingga membuat selaput ketuban pecah sebelum waktunya.

II.4. Metode*Dempster Shafer*

Menurut Sinaga dan Sembiring (2016 : 96), Secara umum teori *Dempster-Shafer* ditulis dalam suatu interval: [*Belief*,*Plausibility*]. *Belief* (Bel) adalah ukuran kekuatan *evidence* dalam mendukung suatu himpunan proposisi.Jika bernilai 0 maka mengindikasikan bahwa tidak ada *evidence*, dan jika bernilai 1 menunjukkan adanya kepastian.*Plausibility* (Pls) akan mengurangi tingkat

kepastian dari *evidence. Plausibility* bernilai 0 sampai 1. Jika yakin akan X’, maka dapat dikatakan bahwa Bel(X’) = 1, sehingga rumus di atas nilai dari Pls(X) = 0.

Berikut ini adalah rumus dari metode *dempster shafer* :

$$\mathbf{m_3(Z) = \frac{\sum_{X\cap Z} m_1(X)m_2(Y)}{1 - k}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

Dimana :

m_1 (X) :*mass function* dari *evidence* X

m_2 (Y) :*mass function* dari *evidence* Y

m_3 (Z) : *mass function* dari *evidence* Z

k : jumlah *conflictevidence*. (Ashari, 2015 : 12).

II.5. Basis Data (*Database*)

Menurut Urva dan Siregar (2015 :95), *Database* adalah sekumpulan tabel-tabel yang saling berelasi, relasi tersebut bisa ditunjukkan dengan kunci dari tiap tabel yang ada. Satu *database* menunjukkan satu kumpulan data yang dipakai dalam satu lingkup perusahaan atau instansi.*Database* mempunyai kegunaan dalam mengatasi penyusunan dan penyimpanan data, maka seringkali masalah yang dihadapi adalah:

1. Redundansi dan Inkonsistensi data
2. Kesulitan dalam pengaksesan data
3. Isolasi data untuk standarisasi
4. *Multi user*

5. Keamanan data

6. Integritas data

7. Kebebasan data.

II.6. Normalisasi

Menurut Triyono (2012 : 19), Normalisasi merupakan parameter digunakan untuk menghindari duplikasiterhadap tabel dalam basis data dan jugamerupakan proses mendekomposisikansebuah tabel yang masih memiliki beberapa anomali atau ketidakwajaran sehingga menghasilkan tabel yang lebih sederhana dan struktur yang bagus, yaitu sebuah tabel yang tidak memiliki *data redundancy* dan memungkinkan *user* untuk melakukan *insert*, *delete*, dan *update* pada baris (*record*) tanpa menyebabkan inkonsistensi data.

1. *First Normal Form (1 NF)*

   Sudah tidak ada *repeating group* yaitu pengulangan yang terjadi pada beberapa atribut atau kolom dalam sebuah tabel, dan juga setiap atribut harus bernilai tunggal. Atribut *multivalued, composite, derive* tidak tunggal. Setiap nilai dari atribut hanya mempunyai nilai tunggal.

2. *Second Normal Form (2 NF)*

   Untuk menjadikan tabel normal tingkat ke 2 maka sudah 1NF dan setiap atribut yang bukan *primary key* sepenuhnya secara *functional* tergantung pada semua atribut pembentuk *primary key*.

3. *Third Normal Form (3 NF)*

Tabel sudah 2NF dan tidak memiliki *transitive dependencies*, *Transitive dependency* adalah ketika ada atribut yang secara tidak langsung tergantung pada *primary key* dan atribut tersebut juga tergantung pada atribut lain yang bukan *primary key*.

4. *Boyce-codd Normal Form (BCNF)*

   Tabel dalam BCNF jika sudah 3NF dan semua *determinants* adalah *candidate keys*. Perbedaan 3NF dan BCNF adalah untuk *functional dependency* A □ B, 3NF memperbolehkan ketergantungan ada dalam relasi jika B adalah *PrimaryKey* dan A bukan merupakan *candidatekey*. Sedangkan BCNF menuntut untuk ketergantungan tetap ada dalam relasi, A harus menjadi *candidate key*.

5. *Fourth Normal Form* (4 NF)

   Relasi berada pada bentuk normal keempat apabila memenuhi syarat BCNF dan tidak mempunyai *multivalue dependency.*

6. *Fifth Normal Form* (5 NF)

   Tabel bentuk normal kelima sering disebut PJNF (*Projection Join Normal Form*), penyebutan PJNF karena untuk suatu relasi akan berbentuk normal kelima jika tabel tersebut dapat dipecah atau diproyeksikan menjadi beberapa tabel dan dari proyeksi-proyeksi itu dapat disusun kembali (*join*) menjadi tabel yang sama dengan keadaan semula. Jika penyusunan ini tidak mungkin dilakukan dikatakan pada relasi itu terdapat *join dependencies*dan dikatakan bersifat *lossy join*.

II.7. *Microsoft Visual Basic* 2010

Menurut Melisa (2015 : 45), *Microsoft Visual Basic* pertama kali diluncurkan pada tahun 1991 dengan nama *Thunder,* yang merupakan *development* pertama yang berbasis *visual* yang dibuat oleh *Microsoft*, untuk menandingi bahasa pemrograman lainnya seperti pemrograman *C, C++, Pascal*, dan bahasa pemrograman lainnya.

Menurut Melisa (2015 : 45), *Microsoft Visual Basic .NET* adalah sebuah alat untuk mengembangkan dan membangun aplikasi yang bergerak di atas sistem *.NET Framework*, dengan menggunakan bahasa *BASIC*. Dengan menggunakan alat ini, para *programmer* dapat membangun aplikasi *Windows Forms*, Aplikasi web berbasis *ASP.NET*, dan juga aplikasi *command-line*.Alat ini dapat diperoleh secara terpisah dari beberapa produk lainnya (seperti *Microsoft Visual C++* atau *Visual C#*), atau juga dapat diperoleh secara terpadu dalam *Microsoft Visual Studio.*

II.8. SQL *Server* 2008

SQL *Server* 2008 adalah sebuahRDBMS (*Relational Database Management System*) yang sangat powerful dan telahterbukti kekuatannya dalam mengolah data.Dalam versi terbarunya ini, SQL *Server* 2008memiliki banyak fitur yang bisa dihandalkanuntuk meningkatkan performa database.

Menurut Ruslan (2013 : 39), SQL *Server* 2008 memiliki suatu GUI(*Graphic User Interface*) yang kita gunakanuntuk melakukan aktivitas sehari hariberkaitan dengan *database*,seperti menulisT-SQL, melakukan *backup* dan *restore database*, melakukan *security database*terhadap aplikasi, dan sebagainya. Pada GUItersebut kita bisa melakukan settinganterhadap SQL *Server* untuk berkerja

lebihoptimal.*Setting*an juga bisa dilakukanmenggunakan *script* untuk memudahkan*developer* mengubah *Setting Options* padaSQL *Server* 2008.

II.9. *Unified Modeling Language* (UML)

Menurut Windu Gata (2013) 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.

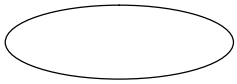
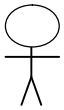
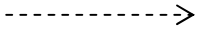
Menurut Urva dan Siregar (2015 : 93).UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak 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 dapat dilihat pada Tabel II.1 dibawah ini:

Tabel II.1. Simbol *Use Case*

Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar 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.

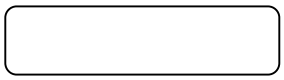
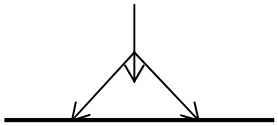
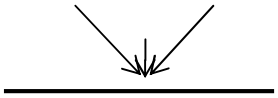
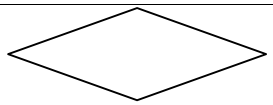
(Sumber:Urva dan Siregar; 2015, Hal : 94)

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* dapat dilihat pada Tabel II.2 dibawah ini:

Tabel II.2. Simbol *Activity Diagram*

Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.

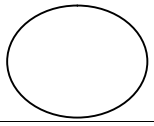
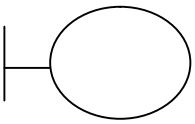
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan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

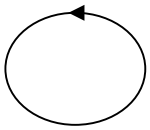
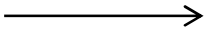
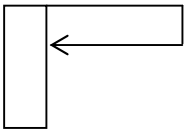
(Sumber : Urva dan Siregar; 2015, Hal : 94)

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* dapat dilihat pada Tabel II.3 dibawah ini :

Tabel II.3. Simbol *Sequence Diagram*

Gambar	Keterangan
	*EntityClass*, 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 : Urva dan Siregar; 2015, Hal : 95)

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 yang dapat dilihat pada Tabel II.4 dibawah ini:

Tabel II.4. *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 : Urva dan Siregar; 2015, Hal : 95)