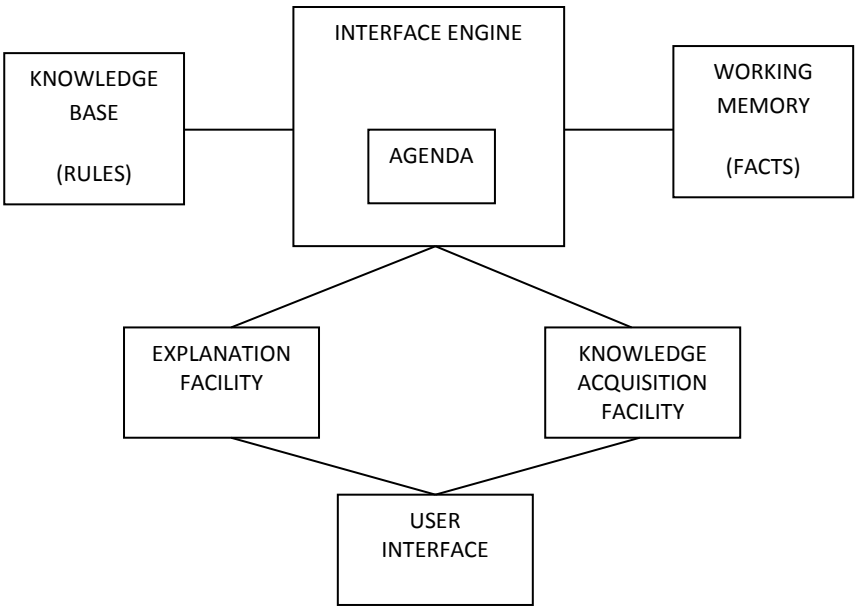
BAB II

TINJAUAN PUSTAKA

II.1. Sistem Pakar

Sistem pakar adalah sistem sistem komputer yang ditujukan untuk meniru semua aspek (*emulates*) kemampuan pengambilan keputusan (*Decision Making*) seorang pakar (Sistem Pakar; Rika Rosnelly; 2012 : 2). Sistem pakar memanfaatkan secara maksimal pengetahuan khusus selayaknya seorang pakar untuk memecahkan masalah. Penyelesaian masalah dapat diuji dan hasilnya akan sesuai dengan hasil yang dikerjakan oleh seorang pakar.

Menurut Rika Rosnelly (Sistem Pakar; Rika Rosnelly; 2012 : 2) adapun struktur sistem pakar dapat dilihat dari gambar II.1 :



Gambar II.1 : Struktur Sistem Pakar
Sumber : Rika Rosnelly (2012 : 13)

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

1. ***Knowledge Base*** (Basis Pengetahuan)

   Basis pengetahuan mengandung pengetahuan 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. Pada struktur sistem pakar diatas, knowledge base disini untuk menyimpan pengetahuan dari pakar berupa rule / aturan (if < kondisi > then <aksi> atau dapat juga disebut *condition-action rules*).

2. ***Interface Engine*** (Mesin Inferensi)

   Mesin Inferensi merupakan otak dari sebuah sistem pakar dan dikenal juga dengan sebutan control structure (struktur kontrol) atau rule intepreter (dalam sistem pakar berbasis kaidah). Mesin inferensi disini adalah processor pada sistem pakar yang mencocokkan bagian kondisi 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 kepercayaaan 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 acquisition facility***

   Meliputi proses pengumpulan,pemindahan dan perubahan dari kemampuan pemecahan masalaha 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 yang menyajikannya ke dalam bentuk yang dapat dimengerti oleh pemakai.

Antar Muka Pengguna, sistem pakar menggantikan seorang pakar dalam situasi tertentu, maka sistem harus menyediakan pendukung yang diperlukan oleh pemakai yang tidak memahami masalah teknis. Sistem pakar juga menyediakan komunikasi antar sistem dan pemakaianya (user) yang disebut sebagai antar muka. Antar muka yang efektif dan ramah penggunaan (user-friendly) penting sekali terutama bagi pemakai yang tidak ahli dalam bidang yang diterapkan pada sistem pakar. Sedangkan Basis pengetahuan, merupakan kumpulan pengetahuan bidang

tertentu pada tingkatan pakar dalam format tertentu. Pengetahuan ini diperoleh dari akumulasi pengetahuan pakar dan sumber-sumber pengetahuan lainnya. Pada sistem pakar ini basis pengetahuan terpisah dengan mesin inferensi. Pemisahan ini bermanfaat untuk pengembangan sistem pakar secara leluasa disesuaikan dengan perkembangan pengetahuan. Dan Mesin inferensi sesungguhnya adalah program komputer yang menyediakan metodologi untuk melakukan penalaran tentang informasi pada basis pengetahuan dan memeori kerja serta untuk merumuskan kesimpulan-kesimpulan.

Komponen ini menyajikan arahan-arahan tentang bagaimana menggunakan pengetahuan dari sistem dengan membangun agenda yang mengelola dan mengontrol langkah-langkah yang diambil untuk menyelesaikan masalah ketika dilakukan konsultasi. Memori kerja, merupakan bagian sistem pakar yang menyimpan fakta- fakta yang diperoleh saat dilakukan proses konsultasi. Fakta-fakta inilah yang nantinya akan diolah oleh mesin inferensi berdasarkan pengetahuan untuk menentukan suatu keputusan pemecahan masalah. Dan Fasilitas penjelasan dapat membrikan informasi kepada pemakai mengenai jalannya penalaran sehingga dihasilkan suatu keputusan. Tujuan adanya fasilitas penjelasan dalam sistem pakar antara lain membuat sistem menjadi lebih cerdas, menunjukkan adanya proses analisa dan yang tidak kalah pentingnya adalah memuaskan psikologis pemakai. Sedangkan Akuisisi pengetahuan adalah proses pengumpulan, perpindahan, dan transformasi dari keahlian/kepakaran pemecahan masalah yang berasal dari beberapa sumber pengetahuan ke dalam bentuk yang dimengerti oleh komputer.

Dengan demikian maka seorang pakar akan dengan mudah menambahkan pengetahuan ataupun kaidah baru pada sistem pakar. Untuk menjamin bahwa pengetahuan pada sistem pakar up to date dan valid, maka fasilitas akuisisi pengetahuan hanya bida diakses oleh pakar. Pengguna awam tidak berhak memakai fasilitas akusisi pengetahuan. (Rika Rosnelly; 2012 : 15)

II.2. Logika Fuzzy

Logika fuzzy pertama kali diperkenalkan oleh Prof. Lotfi A. Zadeh pada tahun 1962. Dasar logika fuzzy adalah teori himpunan fuzzy. Pada teori himpunan fuzzy, peranan derajat keanggotaan sebagai penentu keberadaan elemen dalam suatu himpunan sangatlah penting. Fuzzy secara bahasa diartikan sebagai kabur atau samar (Jurnal ELKHA; Helfi Nasution; 2012 : 4). Dalam fuzzy dikenal derajat keanggotaan yang memiliki rentang nilai 0 (nol) hingga 1(satu). Berbeda dengan himpunan tegas yang memiliki nilai 1 atau 0 (ya atau tidak). Logika Fuzzy merupakan suatu logika yang memiliki nilai kekaburan atau kesamaran (fuzzyness) antara benar atau salah. Dalam teori logika fuzzy suatu nilai bias bernilai atau salah secara bersama.

Nilai keanggotaan atau derajat keanggotaan atau membership function menjadi ciri utama dari penalaran degan logika fuzzy tersebut. Logika fuzzy dapat dianggap sebagai kotak hitam yang menghubungkan antara ruang input dengan ruang output. Kotak hitam tersebut berisi cara atau metode yang dapat digunakan untuk mengolah data input menjadi output dalam bentuk informasi yang baik.

Ada beberapa alasan mengapa orang menggunakan Logika fuzzy, yaitu :

1. Konsep logika fuzzy mudah dimengerti. Konsep matematis yang mendasari penalaran fuzzy sangat sederhana dan mudah dimengerti.
2. Logika fuzzy sangat fleksibel
3. Logika fuzzy memiliki toleransi terhadap data-data yang tidak tepat
4. Logika fuzzy mampu memodelkan fungsi-fungsi non linear yang sangat kompleks
5. Logika fuzzy dapat membangun dan mengaplikasikan pengalaman-pengalaman para pakar secara langsung tanpa harus melalui proses pelatihan
6. Logika fuzzy dapat bekerjasama dengan teknik-teknik kendali secara konvensional
7. Logika fuzzy didasarkan pada bahasa alami

II.2.1. Dasar Logika Fuzzy

Himpunan fuzzy memiliki 2 atribut, yaitu (Kecerdasan Buatan; T.Sutojo, S.Si,M.Kom; 2010 : 213).

1. Linguistik, yaitu penamaan suatu grup yang mewakili suatu kesadaran atau kondisi tertentu dengan menggunakan bahasa alami, seperti :

   MUDA, PAROBAYA, TUA

2. Numeris, yaitu suatu nilai (angka) yang menunjukkan ukuran dari suatu variabel seperti : 40, 25, 50, dan sebagainya.

Ada beberapa hal yang perlu diketahui dalam memahami sistem fuzzy, yaitu:

a. Variabel fuzzy

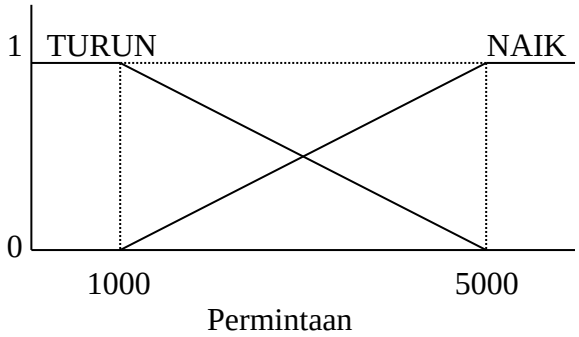
Variabel fuzzy merupakan variabel yang hendak dibahas dalam suatu sistem fuzzy.

b. Himpunan fuzzy

Himpunan fuzzy, yaitu suatu kelompok yang mewakili suatu keadaan tertentu suatu variabel fuzzy (Kecerdasan Buatan; T.Sutojo, S.Si,M.Kom; 2010 : 213).

Contoh (Gambar II.2) :

Variabel permintaan, terbagi menjadi 2 himpunan fuzzy, yaitu NAIK dan TURUN.



Gambar II.2 : Variable permintaan terbagi menjadi 2 himpunan fuzzy Yaitu himpunan NAIK dan himpunan TURUN

Sumber : T.Sutojo S.si M.kom (2010 : 213)

c. Semesta pembicaraan

Semesta pembicaraan adalah keseluruhan nilai yang diperbolehkan untuk diperasikan dalam suatu variabel fuzzy.semesta pembicaraan merupakan himpunan bilangan real yang senantiasa naik (bertambah) secara monoton dari kiri ke kanan. Nilai semesta pembicaraan dapat berupa bilangan posistif maupun negatif. Adakalahnya nilai semesta pembicaraan ini tidak dibatasi batas atasnya.

d. Domain

Domain himpunan fuzzy adalah keseluruhan nilai yang diizinkan dalam semesta pembicaraan dan boleh dioperasikan dalam suatu himpunan fuzzy.

II.2.2. Sistem Inferensi Fuzzy

Seorang pakar memiliki pengetahuan tentang cara kerja dari sistem yang bisa dinyatakan dalam sekumpulan IF-THEN rule. Dengan melakukan inferensi, pengetahuan tersebut bisa ditransfer ke dalam perangkat lunak yang selanjutnya memetakan suatu input menjadi output berdasarkan IF-THEN rule yang diberikan. Sistem fuzzy yang dihasilkan disebut Fuzzy Inference System (FIS).

Pendekatan logika fuzzy secara garis besar diimplementasikan dalam tiga tahapan yang dapat dijelaskan sebagai berikut :

1. Tahap pengaburan (fuzzification) yakni pemetaan dari masukan tegas ke himpunan kabur. FIS mengambil masukan-masukan dan menentukan

derajat keanggotaannya dalam semua fuzzy set menggunakan fungsi keanggotaan masing-masing fuzzy set.

2. Tahap inferensi, yakni pembangkitan aturan kabur. Ada beberapa tahap dalam membangkitkan aturan kabur adalah sebagai berikut : Operasi fuzzy logic dilakukan jika bagian antencendent lebih dari satu pernyataan. Hasil dari operasi ini adalah derajat kebenaran yang berupa bilangan tunggal yang kemudian akan diteruskan ke bagian consequent. Selanjutnya implikasi untuk proses medapatkan consequent atau keluaran sebuah IF-THEN rule berdasarkan derajat kebenaran antencendent. Setelah keluaran IF-THEN rule ditentukan berupa sebuah fuzzy set keluaran yang ada maka tahap selanjutnya adalah melakukan proses agregasi, yaitu proses mengkombinasikan keluaran semua IF-THEN rule menjadi sebuah fuzzy set tunggal.

3. Tahap penegasan (defuzzification), yakni transformasi keluaran dari nilai kabur ke nilai tegas. Masukkan defuzzifikasi adalah sebuah fuzzy set dan keluarannya adalah sebuah bilangan tunggal untuk diisikan ke sebuah variabel keluaran FIS.

II.3. VB.NET

Microsoft Visual Basic (sering disingkat sebagai VB saja) merupakan sebuah bahasa pemrograman yang bersifat event driven dan menawarkan *Integrated Development Environtment* (IDE) visual untuk membuat program

aplikasi berbasis sistem operasi *Microsoft Windows* dengan menggunakan model pemrograman *Common Object Model* (COM) (Sistem Informasi; Adelia,Jimmy Setiawan; 2011 : 114). Visual Basic merupakan turunan bahasa BASIC dan menawarkan pengembangan aplikasi computer berbasis graphic dengan cepat, akses ke basis data menggunakan *Data Access Object* (DAO), *Remote Data Object* (RDO), atau *Active X Data Object* (ADO), serta menawarkan pembuatan control *Active X* dan objek *Active X*.

Visual Basic merupakan turunan bahasa BASIC dan menawarkan pengembangan aplikasi computer berbasis grafik dengan cepa, akses ke basis data menggunakan *Data Access Object* (DAO), *Remote Data Objects* (RDO), atau *ActiveX Data Object* (ADO), serta menawarkan pembuatan kontrol *ActiveX* dan *object ActiveX*. Beberapa bahasa skrip seperti bahasa Visual Basic for Aplications (VBA) juga dapat menggunakan dan Visual Basic Scripting Edition (VBScript), mirip seperti halnya Visual Basic, tetap cara kerjanya yang berbeda. Para pemogrammer dapat membangun aplikasi dengan menggunakan komponen-komponen yang disediakan oleh *Microsoft Visual Basic*. Program-program yang ditulis dengan Visual Basic juga dapat menggunakan Windows API, tapi membutuhkan deklerasi fungsi eksternal tambahan.

II.4. Microsoft SQL Server

Menurut Adelia dan Jimmy Setiawan (Sistem Informasi; Adelia,Jimmy Setiawan; 2011 : 115) SQL (*Structured Query Language*) adalah sebuah bahasa

yang dipergunakan untuk mengakses data dalam basis data relasional. Bahasa ini secara de facto merupakan bahasa standar yang digunakan dalam manajemen basis data relasional. Saat ini hampir semua server basis data yang ada mendukung bahasa ini untuk melakukan manajemen datanya.

SQL teridiri dari 2 bahasa, yaitu *Data Definition Language* (DDL) dan *Data Manipulation Language* (DML). Implementasi DDL dan DML berbeda untuk tiap sistem manajemen basis sata (SMBD), namun secara umun implementasi setiap bahasa ini memiliki bentuk standar yang di tetapkan oleh ANSI. (Youness, 1991)

1) *Data Definition Language* (DDL)

   DDL digunakan untuk mendefinisikan, mengubah serta menghapus basis data dan objek-objek yang diperlukan dalam basis data, misalnya tabel, view, user, dan sebagainya. DDL biasanya digunakan oleh administrator basis data dalam pembuatan sebuah aplikasi basis data. Secara umum DDl yang digunakan adalah:

   1. *CREATE* untuk membuat objek baru.
   2. *USE* untuk menggunakan objek.
   3. *ALTER* untuk mengubah objek yang sudah ada.
   4. *DROP* untuk menghapus objek.

2) *Data Manipulation Laguage* (DML)

   DML digunakan untuk memanipulasi data yang ada dalam suatu label.

   Perintah-perintah yang umum dilakukan adalah :

   1. *SELECT* untuk menampilkan data.

2. *INSERT* untuk menambahkan data baru.
3. *UPDATE* untuk mengubah data yang sudah ada.
4. *DELETE* untuk menghapus data.

II.5. *Flowchart*

Flowchart adalah penggambaran secara grafik dari langkah-langkah dan urutan-urutan prosedur dari suatu program. *Flowchart* menolong *analysis* dan *programer* untuk memecahkan masalah kedalam segmen-segmen yang lebih kecil dan menolong dalam menganalisis alternatif-alternatif lain dalam pengoperasian. *Flowchart* biasanya mempermudah penyelesaian suatu masalah yang perlu dipelajari dan dievaluasi lebih lanju. *Flowchart* adalah bentuk gambar/diagram yang mempunyai aliran satu atau dua arah secara sekuensial. Flowchart digunakan untuk merepresentasikan komponen-komponen dalam bahasa pemogramman.

II.6. *Entity Relationship Diagram*

ERD adalah model konseptual yang mendeskripsikan hubungan antara penyimpanan. ERD digunakan untuk memodelkan struktur data dan hubungan antar data. Dengan ERD, model dapat di uji dengan mengabaikan proses yang dilakukan.

ERD pertama kali dideskripsikan oleh Peter Chen yang dibuat sebagai bagian dari perangkat lunak *CASE*. Komponen-komponen yang termasuk dalam ERD antara lain, adalah :

1) Entitas(Entity)

   Sebuah barang atau objek yang dapat dibedakan dari objek lain.

2) Relasi (Relationship)

   Asosiasi 2 atau lebih entitas dan berupa kata kerja.

3) Atribut (Attribute)

   Properti yang dimiliki setiap entitas yang akan disimpan datanya.

4) Kardinalitas (Kardinality)

   Angka yang menunjukan banyaknya kemunculan suatu obyek terkait dengan kemunculan obyek lain pada suatu relasi. Kardinalitas relasi yang terjadi antara dua himpunan entitas (misalnya A dan B) dapat berupa :

   1. Modalitas (Modality) adalah Partisipasi sebuah entitas pada suatu relasi, 0 jika partisipasi bersifat ”optional”/parsial, dan 1 jika partisipasi bersifat ”wajib”/total.
   2. Total constraint adalah constraint yang mana data dalam entitas yang memiliki constraint tersebut terhubung secara penuh ke dalam entitas dari realsinya.

II.7. Data Flow Diagram

Data flow diagram adalah sebuah grafik yang menjelaskan sebuah sistem dengan menggunakan bentuk-bentuk dan simbol-simbol untuk menggambarkan aliran data dari proses-proses yang saling berhubungan. Data flow diagram ini adalah salah satu alat pembuatan model yang sering digunakan, khusunya bila fungsi-fungsi sistem merupakan bagian yang lebih penting dan kompleks daripada data yang dimanipulasi oleh sistem. Dengan kata lain, data flow diagram adalah alat pembuatan model yang memberikan penekanan hanya pada fungsi sitem. Data flow diagram ini merupakan alat perencangan sistem yang berorientasi pada alur data dengan konsep dekomposisi dapat digunakan untuk penggambaran analisa maupun rancangan sistem yang mudah dikomunikasikan oleh profesional sistem kepada pemakai maupun pembuat program.

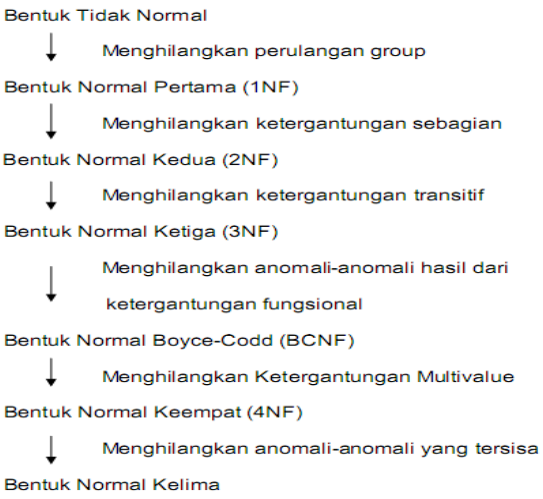
II.8. Normalisasi

Normalisasi adalah proses yang digunakan untuk menentukan pengelompokan atribut-atribut dalam sebuah relasi sehingga diperoleh relasi yang berstruktur baik. Dalam hal ini yang dimaksud dengan relasi yang berstruktur baik adalah relasi yang memenuhi dua kondisi berikut

1. Mengandung redundansi sedikit mungkin, dan
2. Memungkinkan baris-baris dalam relasi disispkan, dimodifikasi dan dihapus tanpa menimbulkan kesalahan atau ketidakkonsistenan.

Normalisasi sendiri dilakukan melalui sejumlah langkah. Setiap langkah berhubungan dengan bentuk normal (*normal form*) tertentu. Dalam hal ini yang disebut bentuk normal adalah suatu keadaan relasi yang dihasilkan oleh penerapan

aturan-aturan sederhana yang berhubungan dengan dependensi fungsional terhadap relasi tersebut. Gambar II.3 berikut ini akan memperlihatkan hubungan keenam bentuk normal tersebut.



Gambar II.3 : Langkah-langkah Dalam Normalisasi
Sumber: Adelia,Jimmy Setiawan (2011 ; 117)

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

UML Adalah perkakas untuk analisis danperancangan yang sesungguhnya digunakan untuk peyederhanaan permasalahan. UML merupakan metodologi kolaborasi antara metoda-metoda *Booch,* OMT (*Object Modeling Technique*),serta OOSE (*Object Oriented Software Engineering*) dan beberapa metoda lainya, merupakan metodologi yang paling sering di gunakan pada saat ini untuk mengadaptasi maraknya penggunaan bahasa pemrograman berorientasi objek.

Menurut prabowo Pudjo Widodo & Herlawati (2011:10) UML diaplikasikan untuk maksud tertentu, biasanya antara lain :

1. Merancang perangkat Lunak.
2. Sarana Komunikasi antaraperangkat lunak dengan proses bisnis.
3. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasikan sistem yang ada, proses-proses dan organisasinya.

Blok pembangunan utama UML adalah diagram. Beberapa diagram ada yang rinci (jenis *timmingdiagram*) dan lainya ada yang bersifat umum (misalnya diagram kelas). Para pengembang sistem berorientasi objek menggunakan bahasa model untuk menggambarkan, membangun dan mendokumentasikan sistem yang mereka rancang. UML memungkinkan para anggota team untuk bekerja sama dengan bahasa model yang sama dengan mengaplikasikan beragam sistem. Intinya UML merupakan alat komunikasi yang konsisten dalam mendukung para pengembang sistem saat ini.

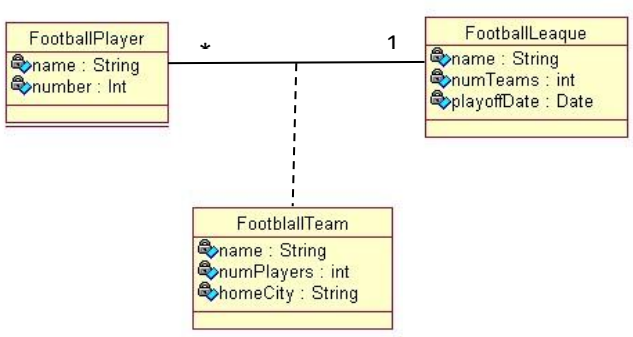
II.9.1. Diagram-Diagram UML

Beberapa literatur menyebutkan bahwa UML menyediakan sembilan jenis diagram, model-model ini dapat dikelompokan berdasarkan sifatnya yaitu statis atau dinamis. Jenis diagram itu antara lain :

1. **Diagram Kelas** : Bersifat statis. Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi. Diagram ini umum dijumpai pada pemodelan sistem berorientasi objek. Meskipun bersifat statis, sering pula diagram kelas memuat kelas-kelas aktif.

Terkadang hubungan antara dua elemen tidak sederhana misalnya suatu tim pemain bola (football player) berasosiasi dengan liga (league) lewat suatu regu. Jika hubungannya terlalu rumit, bisa dibuatkan hubungan asosiasi antar

kelas. Suatu asosiasi kelas memiliki nama dan atribut seperti kelas biasa, notasi untuk kelas asosiasi adalah dengan garis putus-putus mengenai garis asosiasi utama.

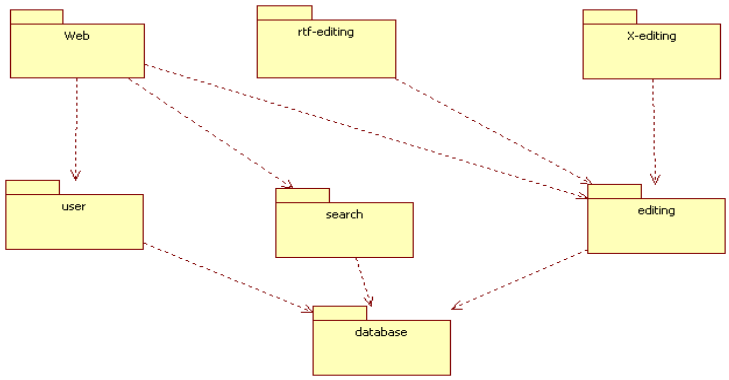


Gambar II.4 : Diagram Kelas Asosiasi

Sumber : Prabowo Pudjo Widodo & Herlawati (2011 : 69)

Ketika ditranslasikan menjadi kode, biasanya kelas asosiasi dianggap sebagai kelas bias, sehingga ada tiga kelas yang terbentuk, yang perlu diperhatikan dari gambar II.5 di atas adalah bahwa *FootballPlayer* tidak memiliki referensi langsung kepada *FootballLeague* tapi memiliki referensi terhadap *FootballTeam.* FootballTeam akan memiliki referensi terhadap *FootballLeague.*

2. **Diagram Paket (*Package Diagram*)** : Bersifat statis. Diagram ini memperlihatkan kumpulan kelas-kelas, merupakan bagian dari diagram komponen.



Gambar II.5 : Diagram Paket

Sumber : Prabowo Pudjo Widodo & Herlawati (2011 : 88)

Gambar di atas memiliki sedikit informasi, oleh karena itu perlu ditambahkan suatu dokumen tertulis yang menjelaskan dasar-dasar paket itu. Dokumen itu mungkin berisi daftar sebagai berikut :

a. *Web*

   Dibutuhkan keterampilan khusus : *HTML*, *css* dan *Struts*, teknologi presentasi. Seluruhnya memiliki ketergantungan.

b. *Database*

   Dibutuhkan suatu manajemen *database* dan keterampilan membuat model. Memiliki sifat bebas atau kurang tergantung.

c. *User*

Dibangun diluar lokais (*remote team*).

d. *Search*

Dibutuhkan suatu kemampuan dengan dengan teknologi *search engine.* Bersifat *subsistem* lokal.

e. *Editing*

Terdiri dari dasar *fiturediting* untuk penerbitan pertama.

f. *Rtf-editing*

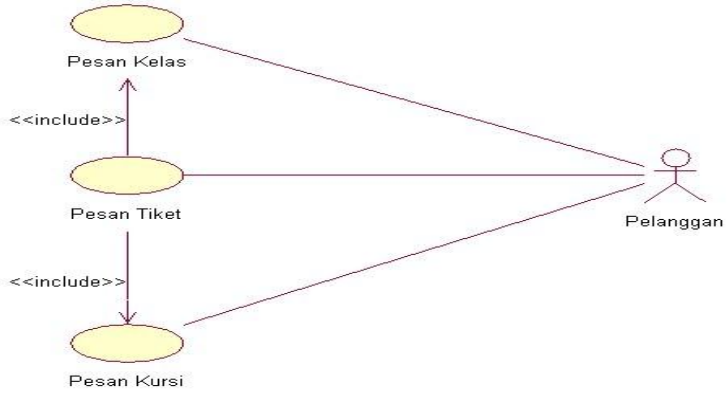
Terdiri dari dasar *fitur editing* untuk release berikutnya.

g. *X-editing*

Terdiri dari *fitur editing* atas permintaan klien khusus

Diagram paket pada *gambar II.6* diatas tidak memperlihatkan paket yang komplit. Diagram tersebut hanya bermanfaat dari sisi manajemen, sementara itu *programer* membuat diagram paket level bawah agar dihasilkan kode program yang benar.

3. **Diagram *Use-Case*** : Bersifat statis. Diagram ini memperlihatkan himpunan *use-case* dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini terutama sangat penting untuk mengorganisasi dan memodelkan perilaku suatu sistem yang dibutuhkan serta diharapkan pengguna.

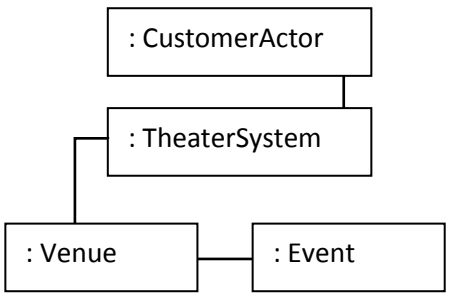


Gambar II.6 : Diagram *Use Case*

Sumber : Prabowo Pudjo Widodo & Herlawati (2011 : 30)

Gambar II.7 menggambarkan hubungan inklusi antara *usecase*pesan tiket dengan *use case*pesan kelas dan pesan kursi. Pesan tiket disebut *use case*pemanggil (*calling use case*) sedangkan pesan kelas dan pesan kursi disebut use case terpanggil (*called use case*). *Use case* pesan tiket belum lengkap karena harus pesan kelas dan pesan kursi terlebih dahulu.

4. **Diagram Interaksi dan *Sequence* (Urutan)** : Bersifat dinamis. Diagram urutan adalah diagram interaksi yang menekankan pada pengiriman pesan dalam suatu waktu tertentu.
5. **Diagram Komunikasi (*Communication Diagram*)** : Bersifat dinamis. Diagram sebagai pengganti diagram kolaborasi UML 1.4 yang menekankan organisasi struktural dari objek-objek yang menerima serta mengirim pesan.

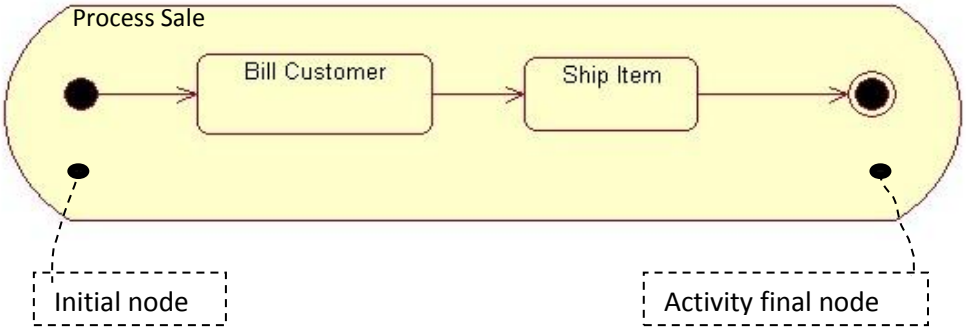


Gambar II.7 : Diagram Komunikasi

Sumber : Prabowo Pudjo Widodo & Herlawati (2011 : 196)

Diagram komunikasi dimaksudkan untuk melengkapi diagram urutan dengan memberikan tampilan visual pada pesan yang disampaikan antar objek. Dasar dari diagram komunikasi adalah diagram objek, seperti ditunjukan pada gambar II.8. Tiap objek dalam diagram komunikasi disebut garis hidup objek (object lifetime).

6. **Diagram *Statechart* (*Statechart Diagram*)** : Bersifat dinamis. Diagram status memperlihatkan keadaan-keadaan pada sistem, memuat status (*state*), transisi, kejadian serta aktifitas. Diagram ini terutama pentinguntuk memperlihatkan sifat dinamis dari antarmuka (*interface*), kelas, kolaborasi dan terutama penting pada pemodelan sistem-sistem yang reaktif.
7. **Diagram Aktivitas (*Activity Diagram*)**. Bersifat dinamis. Diagram aktivitas adalah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem. Diagram ini terutamapenting dalam pemodelan fungsi-fungsi suatu sistem dan memberi tekanan pada aliran kendaliantar objek.

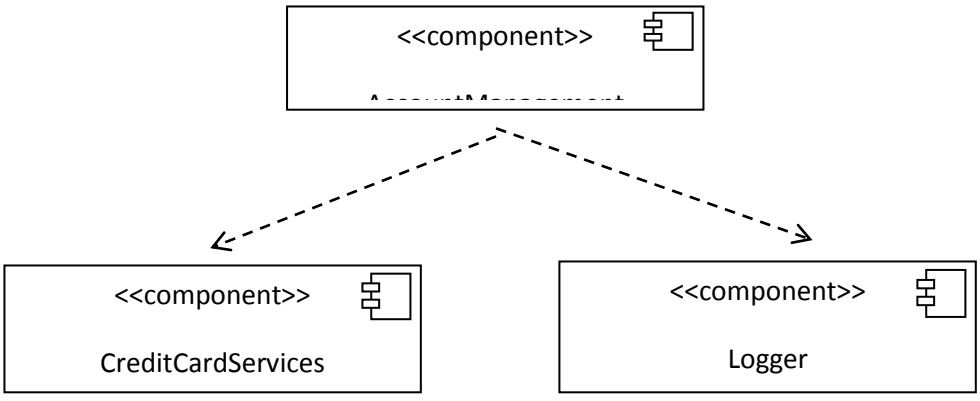


Gambar II.8 : Diagram Aktivitas

Sumber : Prabowo Pudjo Widodo & Herlawati (2011 : 146)

Seperti yang terlihat pada gambar II.9 di atas, tiap aktivitas dimulai dengan titik awal (*initial node*) dan diakhiri dengan aktivitas titik akhir (*final node*). Saat ini aktivitas mencapai aktivitas titk akhir, aktivitas itu terhenti. Titk awal digambarkan dengan titik hitam dan titik akhir dengan lingkaran tebal yang didalamnya terdapat titik hitam.

8. Diagram Komponen (*Component Diagram*) : Bersifat statis. Diagram komponen ini memperlihatkan organisasi serta ketergantungan sistem/perangkat lunak pada komponen-komponen yang telah ada sebelumnya. Diagram ini berhubungan dengan diagram kelas dimana komponen secara tipikal dipetakan kedalam satuatau lebih kelas-kelas, antarmuka-antarmuka serta kolaborasi-kolaborasi.



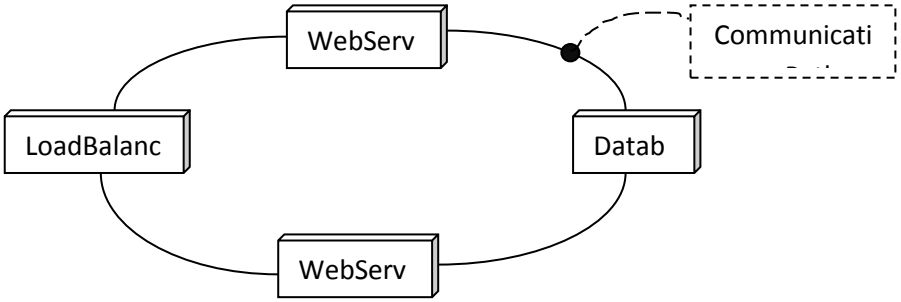
Gambar II.9 : Diagram Komponen

Sumber : Prabowo Pudjo Widodo & Herlawati (2011 : 96)

Gambar II.10 menggambarkan komponen *AccountManagement*memiliki ketergantungan dengan kedua komponen lainnya. Penggambaran

ketergantungan pada gambar di atas merupakan level tertinggi dan tidak merinci ketergantungan secara lebih besar

9. Diagram *Deployment* (*Deployment Diagram*) : Bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan (*run-time*). Memuat simpul-simpul beserta komponen-komponen.



Gambar II.10 : Diagram *Deployment*

Sumber : Prabowo Pudjo Widodo & Herlawati (2011 : 117)

Diagram ini sangat berguna saat aplikasi kita berlaku sebagai aplikasi yang dijalankan pada banyak mesin (*distributed computing*), (prabowo Pudjo Widodo & Herlawati 2011 : 6).