BAB 2
TINJAUAN PUSTAKA

Landasan teori yang mendukung proses analisis sistem serta mendukung proses perancangan sistem pakar untuk diagnosa penyakit tanaman anggrek adalah sebagai berikut.

2.1 Tanaman Anggrek

Anggrek sudah dikenal dalam sejarah Cina sejak 3000 tahun yang lalu. Beberapa negara menobatkan bunga anggrek sebagai bunga kebangsaannya. Sebagai contoh, *Guatemala* menempatkan anggrek *Lysate skinneri varietas Alba* sebagai bunga nasional, sementara Republik Panama menobatkan *Presteria elata*. Lambang kerajaan Minnesota di benua Amerika dulu kala juga bunga anggrek. *Vanda Miss Joaquim* dijadikan sebagai bunga nasional Singapura. *Vanda* tersebut sebenarnya merupakan hasil persilangan, yakni antara *Vanda hookeriana* dengan *Vanda teres*. Indonesia juga mengangkat anggrek *Phalaenopsis* amabilis sebagai bunga nasional.

Anggrek termasuk dalam famili *Orchidaceae* (keluarga anggrek). Di dunia ini terdapat lebih kurang 25.000 jenis anggrek, dan sekitar 5.000 jenis di antaranya terdapat di Indonesia. Dari 5.000 jenis tersebut, di Pulau Sumatra terdapat 1.118 jenis, Pulau Jawa 731 jenis, Pulau Kalimantan (Borneo) ± 2.500 jenis, Pulau Sulawesi dan Maluku 817 jenis, dan Pulau Papua lebih dari 3.000 jenis.

Anggrek spesies Indonesia sangat indah, misalnya Vanda *tricolor*. Anggrek ini terdapat banyak di Jawa Barat, juga di hutan-hutan Kaliurang,Yogyakarta. Vanda *hookeriana*, yang berasal dari Sumatra, memiliki bunga berwarna ungu berbintik-bintik sangat menarik. Spesies Indonesia lainnya adalah *Larat* atau *Dendrobium phalaenopsis*, juga anggrek bulan atau *Phalaenopsis amabilis*, dan yang tidak kalah menarik adalah *Vanda daeri*. Anggrek-anggrek spesies Indonesia ini sangat dikenal luas dan merupakan anggrek yang bagus sebagai bahan persilangan.

Anggrek-anggrek spesies sudah terbukti merupakan bahan-bahan persilangan yang bermutu. Misalnya, *Vanda tricolor* disilangkan dengan *Vanda teres* menurunkan *Vanda Emma van Deventer*, yang memiliki bunga sangat indah. *Vanda Emma van Deventer* ini disilangkan lagi dengan *Vanda* sanderiana yang berasal

dari Filipina menghasilkan *Vanda Nellie Morley* yang bunganya juga sangat dikagumi orang. *Vanda hookeriana* dari Sumatra disilangkan dengan *Vanda teres* menghasilkan *Vanda Miss Yoaquim* yang bunganya sudah tak asing lagi di negara kita. *Vanda tricolor* disilangkan dengan *Vanda corulea* dari Muangthai menghasilkan *Vanda Gilbert triboulet* dengan bunga berwarna biru.[8]

2.2 Hama

Hama adalah organisme yang menyerang atau mengurangi kualitas dan kuantitas tanaman sehingga pertumbuhan dan perkembangannya terganggu.[9] Jenis-jenis hama :

1) Tinjau

   Tungau (*mites*) yang menyerang anggrek adalah Pseudoleptus vandergooti. Hewan ini berukuran sangat kecil, yakni sekitar 0,2 mm. Hama ini menyerang bagian daun dengan mengisap cairan sel sehingga lama kelamaan daun akan berkeriput. Tanaman yang terserang tungau akan terlihat daunnya berwarna kuning keperakan dan selanjutnya gugur. Pemberantasan hama ini menggunakan insektisida seperti *Dursban, Kelthane* dengan dosis sesuai anjuran pada labelnya[8].

2) Thrips

   Serangga berukuran 1-2 mm yang menyerang anggrek adalah *Dichromothrips smithi*. Hama ini hidup menempel pada daun-daun muda dan putik bunga. *Thrips* menggigit dan mengisap cairan sel-sel daun sehingga timbul bercak perak yang lama-kelamaan menjadi coklat. Pertumbuhan anggrek yang terserang thrips terhambat, daun pucuk keriput dan tidak mau membuka serta berwarna karat. Selain itu, bunga tumbuh tidak sempurna dan bahkan jadi rontok. Thrips dapat diberantas dengan menggunakan *Curacron, Confidor, Kel thane*, atau *Pe gasus* dengan dosis sesuai anjuran[8].

3) Kepik

   Kepik (*Mertila malayensis*) menyerang bagian daun. Serangannya ditandai oleh adanya bintik-bintik. Hama ini mengisap cairan sel-sel daun sehingga lama kelamaan daun berbintik-bintik, semakin bertambah hari semakin bertambah banyak. Akibatnya, daun terhambat pertumbuhannya, keriput, dan

bahkan rontok. Kepik dapat diberantas dengan *Supracide, Kelthane,* atau *Decis* dengan dosis sesuai label[8].

4) Ulat Grayak.

   Spodoptera litura Merupakan salah satu jenis hama penting yang merusak daun dibandingkan dengan hama perusak daun lainnya. Pengendaliannya menggunakan *Nuclear Polyhedrosis Virus* (NPV) adalah salah satu jenis virus patogen yang berpotensi sebagai agensia hayati dalam mengendalikan ulat grayak, karena bersifat spesifik, selektif, efektif untuk hama hama yang telah resisten terhadap insektisida dan aman terhadap lingkungan. NPV telah dikembangkan secara in vivo di laboratorium Balitkabi, untuk pengendalian hayati hama Lepidoptera. NPV diketahui efektif mengendalikan hama ulat grayak[10].

5) Kutu Kebul

   Hama kutu putih atau kutu kebul biasanya bergerombol sampai puluhan ribu ekor. Mereka merusak dengan cara mengisap cairan. Semua bagian tanaman bisa diserangnya dari buah sampai pucuk. Serangan pada pucuk menyebabkan daun kerdil dan keriput seperti terbakar[11]. pengendalian penyebaran hama tersebut. Pengendalian dilakukan dengan menentukan kontrol optimal dari model penyebaran hama kutu putih yaitu dengan penyemprotan pestisida dan kombinasi penyemprotan pestisida dan kontrol pengaturan jarak tanaman. Dengan metoda Minimum Pontryagin, solusi optimal kontrol ditentukan[12].

6) Hama pemakan daun

   Serangan berat organisme pengganggu pada tanaman menyebabkan daun rusak atau habis termakan sehingga dapat menurunkan produksi sampai mematikan tanaman. Hama ulat pemakan daun *Spodoptera* sp. dan *Plutella* sp. Salah satu cara pengendalian organisme pengganggu tanaman (OPT) adalah dengan menggunakan insektisida nabati. Beberapa jenis insektisida nabati yang berasal dari tumbuhan telah dikembangkan untuk mengendalikan hama ulat pemakan daun yaitu tanaman mimba (*Azadirachta indica*) dan tanaman cengkeh[13].

7) Kutu Perisai

Hama ini sangat kecil berwarna merah, sehingga kadang-kadang harus dilihat dengan kaca pembesar, pengendalian hama Digosok dengan kapas atau sikat dan air sabun. Jika serangan sudah para, disemprot dengan *insektisida suprecide* 40 EC dengan dosis 2 cc/liter air[14].

8) Siput Telanjang

Hewan ini *triploblasik selomata* yang bertubuh lunak termasuk semua hewan lunak dengan maupun tanpa cangkang. Pengendalian hama jika jumlah sedikit(2- 5 ekor) dapat dibunuh secara langsung, Jika Banyak maka menggunakan *mulustisida* atau terjebak dengan bubuk pursi[14].

9) Kumbang Gajah

Kumbang gajah (*Orchidiphilus arterrimus*) berukuran 3-7 mm, berwarna hitam, dan mempunyai belalai seperti gajah. Hama ini menyerang sejak stadia larva sampai dewasa. Serangannya terjadi terutama pada awal musim hujan. Kumbang gajah bersembunyi pada pangkal batang, ketiak daun, atau buah pada siang hari. Serangan kumbang gajah bisa memusnahkan seluruh isi kebun sehingga perlu diwaspadai apabila ditemukan 3 ekor kumbang gajah dalam setiap 10 pot. Pengendalian hama ini bisa dilakukan secara manual dengan membunuh kumbang serta memangkas dan membakar batang yang terserang. Bersihkan pot dan kebun dari kepompong dan telur kumbang gajah. Pengendalian secara kimiawi bisa dilakukan dengan menyemprotkan pestisida *Confidor* selama 7-10 hari[8].

10) Kumbang Penggerek

Kumbang Penggerek Pucuk (Oryctes rhinoceros), Pengendalian hama dengan insektisida kimiawi akan memberikan dampak positif dengan matinya hama tetapi menimbulkan dampak negatif seperti resistensi, resurgensi, dan letusan hama kedua. Selain itu juga mengganggu kesehatan manusia dan keseimbangan lingkungan, yang disebabkan oleh residu yang tinggi pada komponen produksi dan ekosistem [15].

11) Bekicot

Siput ini merupakan salah satu hama yang paling merusak yang menyerang daerah subtropis dan tropis, menyebabkan kerusakan besar pada pertanian, perkebunan komersial dan kebun rumah tangga[16]. Pengendalian bekicot

dilakukan dengan membuang semua bekicot yang berada di tanaman dan sekitar tanaman secara manual. Sanitasi kebun perlu dilakukan untuk menjaga kebersihan kebun, sehingga kehadiran hama dapat dicegah[17].

12) Ulat

Ulat yang sering menyerang daun muda, tunas, dan kuncup bunga adalah ulat jengkal (*Negeta chlorocrota Hps*.) yang berukuran 35 mm. Apabila didapati ulat pada tanaman, musnahkan secara manual. Jika hama ini tidak dikendalikan, daun tanaman akan habis. Serangan yang tinggi dapat diberantas dengan *Emcindo, Hapacin*, atau *Bassa* dengan dosis sesuai label[8].

2.3 Sistem Pakar

Sistem pakar merupakan sistem yang berusaha mengadopsi tentang pengetahuan manusia menuju ke komputer, komputer ini dapat menyelesaikan suatu masalah seperti yang biasa dilakukan oleh para ahli. Sistem pakar yang baik dirancang agar dapat menyelesaikan suatu permasalahan tertentu dengan meniru kerja dari para ahli. Sistem pakar yang baik dirancang agar dapat menyelesaikan suatu permasalahan tertentu dengan meniru kerja dari para ahli. Dengan sistem pakar, orang awam pun dapat menyelesaikan masalah yang cukup rumit yang sebenarnya hanya dapat diselesaikan dengan bantuan para ahli. Bagi para ahli, sistem pakar juga akan membantu aktivitasnya sebagai asisten yang sangat berpengalaman. Sistem pakar adalah salah satu cabang dari AI yang membuat penggunaan secara luas knowledge yang khusus untuk penyelesaian masalah tingkat manusia yang pakar. Seorang pakar adalah orang yang mempunyai keahlian dalam bidang tertentu, yaitu pakar yang mempunyai knowledge atau kemampuan khusus yang orang lain tidak mengetahui atau mampu dalam bidang yang dimilikinya. Ketika sistem pakar dikembangkan pertama kali sekitar tahun 70-an, sistem pakar hanya berisi *knowledge* yang eksklusif. Namun demikian sekarang ini istilah sistem pakar sudah digunakan untuk berbagai macam sistem yang menggunakan teknologi sistem pakar itu. Teknologi sistem pakar ini meliputi bahasa sistem pakar, program dan perangkat keras yang dirancang untuk membantu pengembangan dan pembuatan sistem pakar.

Sistem pakar mampu memecahkan masalah tanpa dipengaruhi oleh faktor dari luar seperti intimidasi, paksaan kejiwaan, faktor ekonomi maupun perasaan. Berikut ini beberapa keuntungan dari sistem pakar:

1. Memungkinkan orang awam bisa mengerjakan pekerjaan para ahli.
2. Bisa melakukan proses secara berulang secara otomatis.
3. Menyimpan pengetahuan dan keahlian para pakar.
4. Mampu mengambil dan melestarikan keahlian para pakar (terutama yang termasuk keahlian langka).
5. Mampu beroperasi dalam lingkungan yang berbahaya.
6. Memiliki kemampuan untuk bekerja dengan informasi yang tidak lengkap dan mengandung ketidakpastian.
7. Tidak memerlukan biaya saat tidak digunakan, sedangkan pada pakar manusia memerlukan biaya sehari-hari.
8. Dapat digandakan (diperbanyak) sesuai kebutuhan dengan waktu yang minimal dan sedikit biaya.
9. Dapat memecahkan masalah lebih cepat daripada kemampuan manusia dengan catatan menggunakan data yang sama.
10. Menghemat waktu dalam pengambilan keputusan.
11. Meningkatkan kualitas dan produktivitas.

2.3.1 Konsep Dasar Sistem Pakar

Konsep dasar sistem pakar mengandung keahlian, ahli/pakar, pengalihan keahlian, mengambil keputusan, aturan, kemampuan menjelaskan.

2.3.1.1 Keahlian

Keahlian adalah suatu kelebihan penguasaan pengetahuan di bidang tertentu yang diperoleh dari pelatihan, membaca atau dari pengalaman. Bentuk pengetahuan yang termasuk keahlian:

1. Fakta-fakta pada lingkup permasalahan tertentu.
2. Teori-teori pada lingkup permasalahan tertentu.
3. Aturan-aturan berkenaan dengan lingkup permasalahan tertentu.
4. *Meta –knowledge* (pengetahuan tentang pengetahuan)

2.3.1.2 Ahli / Pakar

Seorang ahli adalah seseorang yang mampu menjelaskan suatu tanggapan, mempelajari hal- hal baru seputar topik permasalahan, menyusun kembali pengetahuan jika dipandang perlu, memecahkan masalah dengan cepat dan tepat.

2.3.1.3 Pengalihan Keahlian

Tujuan dari sistem pakar adalah untuk mentransfer keahlian dari seorang pakar ke dalam komputer kemudian ke masyarakat. Proses ini meliputi 4 kegiatan, yaitu perolehan pengetahuan (dari para ahli atau sumber-sumber lainnya), representasi pengetahuan ke komputer, kesimpulan dari pengetahuan dan pengalihan pengetahuan ke user.

2.3.1.4 Mengambil Keputusan

Hal yang unik dari sistem pakar adalah kemampuan untuk menjelaskan dimana keahlian tersimpan dalam basis pengetahuan. Kemampuan komputer untuk mengambil kesimpulan dilakukan oleh komponen yang dikenal dengan mesin inferensi yaitu meliputi prosedur tentang pemecahan masalah.

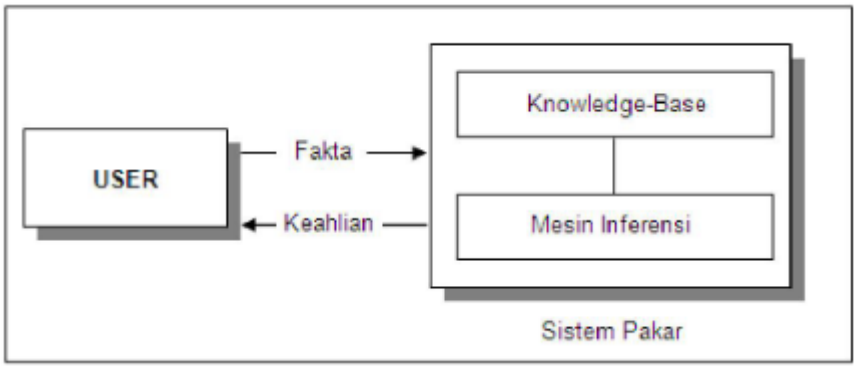
2.3.1.5 Aturan

Sistem pakar yang dibuat merupakan sistem yang berdasarkan pada aturan – aturan dimana program disimpan dalam bentuk aturan-aturan sebagai prosedur pemecahan masalah. Aturan tersebut biasanya berbentuk *IF – THEN*.

2.3.1.6 Kemampuan Menjelaskan

Keunikan lain dari sistem pakar adalah kemampuan dalam menjelaskan atau memberi saran/rekomendasi serta juga menjelaskan mengapa beberapa tindakan/saran tidak direkomendasikan.

Gambar 1 menggambarkan konsep dasar suatu sistem pakar *knowledge-base*. User menyampaikan fakta atau informasi untuk sistem pakar dan kemudian menerima saran dari pakar atau jawaban ahlinya. Bagian dalam sistem pakar terdiri dari dua komponen utama, yaitu *knowledge base* yang berisi *knowledge* dan mesin inferensi yang menggambarkan kesimpulan. Kesimpulan tersebut merupakan respons dari sistem pakar atas permintaan user.



Gambar 2. 1 Konsep Dasar Fungsi Sistem Pakar

2.3.2 Tujuan Sistem Pakar

Pengalihan keahlian dari para ahli ke komputer untuk kemudian dialihkan ke orang lain yang bukan ahli. Seorang ahli adalah seseorang yang mampu menjelaskan suatu tanggapan, mempelajari hal-hal baru seputar topik permasalahan(domain), menyusun kembali pengetahuan jika dipandang perlu, memecahkan aturan-aturan jika dibutuhkan dan menentukan relevan tidaknya keahlian mereka.

Proses pengalihan pengetahuan membutuhkan 4 aktivitas, yaitu :

1. Tambahkan pengetahuan (dari para ahli atau sumber-sumber lainnya).
2. Representasikan pengetahuan (ke komputer)
3. Lakukan inferensi pengetahuan.
4. Pengalihan pengetahuan ke user.

Pengetahuan yang disimpan ke dalam komputer disebut basis pengetahuan. Ada 2 tipe pengetahuan, yaitu: fakta dan prosedur (biasanya berupa aturan).

2.3.3 Permasalahan Yang Dapat Diselesaikan Dengan Sistem Pakar

Sistem pakar dapat diaplikasikan untuk memecahkan berbagai permasalahan. Umumnya kecepatan dalam memecahkan masalah pada suatu sistem pakar relatif lebih cepat dibandingkan dengan seorang pakar manusia. Hal ini sudah dibuktikan pada beberapa sistem pakar yang terkenal di dunia.

Berikut ini contoh berbagai masalah yang dapat diselesaikan dengan menggunakan sistem pakar:

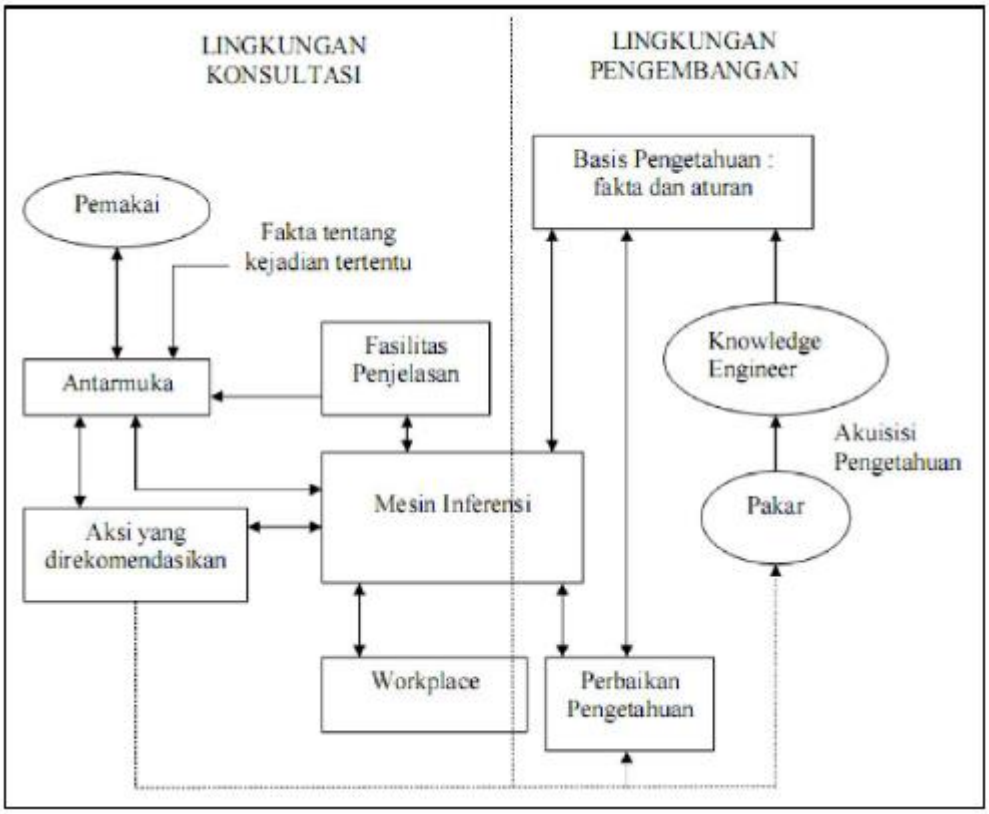
1. Interpretasi, pengambilan keputusan dari hasil observasi, termasuk: pengawasan, pengenalan ucapan, analisis citra, interpretasi sinyal dan beberapa analisis kecerdasan lainnya.
2. Prediksi, diantaranya: peramalan, prediksi demografis, peramalan ekonomi, prediksi lalu lintas, estimasi hasil, militer, pemasaran dan keuangan.
3. Diagnosis, diantaranya: medis, elektronis, mekanis dan diagnosis perangkat lunak.
4. Perancangan: perancangan sirkuit dan bangunan.
5. Perencanaan, seperti: perencanaan keuangan, komunikasi, produk dan manajemen proyek.
6. Monitoring: *Computer-Aided* Monitoring *System*.
7. Debugging: memberikan resep obat terhadap suatu kegagalan.
8. Instruksi, untuk diagnosis, debugging dan perbaikan kinerja. Kontrol, terhadap interpretasi, prediksi, perbaikan dan monitoring kelakuan sistem.

2.3.4 Struktur Sistem Pakar

Ada dua bagian utama sistem pakar:

1. Lingkungan pengembangan (*development environment*): digunakan untuk memasukkan pengetahuan pakar ke dalam lingkungan sistem pakar.

2. Lingkungan konsultasi (*consultation environment*):digunakan oleh user yang bukan pakar untuk memperoleh pengetahuan pakar.[18]



Gambar 2. 2 Struktur Sistem Pakar

2.4 Metode Forward Chaining

Algoritmaforward-chaining adalah satu dari dua metode utama reasoning (pemikiran) ketika menggunakan inference engine (mesin pengambil keputusan) dan bisa secara logis dideskripsikan sebagai aplikasi pengulangan dari modus ponens (satu set aturan inferensi dan argumen yang valid). Lawan dari forward-chaining adalah backward-chaining.

Forward-chaining mulai bekerja dengan data yang tersedia dan menggunakan aturan- aturan inferensi untuk mendapatkan data yang lain sampai sasaran atau kesimpulan didapatkan. Mesin inferensi yang menggunakan forward- chaining mencari aturan-aturan inferensi sampai menemukan satu dari antecedent (dalil hipotesa atau klausa IF -THEN) yang benar. Ketika aturan tersebut ditemukan maka mesin pengambil keputusan dapat membuat kesimpulan, atau konsekuensi (klausa THEN), yang menghasilkan informasi tambahan yang baru dari data yang disediakan. Mesin akan mengulang melalui proses ini sampai sasaran ditemukan.

Forward-chaining adalah contoh konsep umum dari pemikiran yang dikendalikan oleh data (data-driven) yaitu, pemikiran yang mana focus perhatiannya dimulai dari data yang diketahui. *Forward-chaining* bisa digunakan didalam agen untuk menghasilkan kesimpulan dari persepsi-persepsi yang datang, seringkali tanpa query yang spesifik[19].

2.5 Metode Certainty Factor

Faktor kepastian (*certainty factor*) diperkenalkan oleh Shortliffe Buchanan dalam pembuatan MYCIN (Wesley, 1984). *Certainty factor* (CF) merupakan nilai parameter klinis yang diberikan MYCIN untuk menunjukkan besarnya kepercayaan. Dalam menghadapi suatu masalah sering ditemukan jawaban yang tidak memiliki kepastian penuh. Ketidak pastian ini bisa berupa probabilitas atau kebolehjadian yang tergantung dari hasil suatu kejadian. Hasil yang tidak pasti disebabkan oleh dua faktor yaitu aturan yang tidak pasti dan jawaban user yang tidak pasti atas suatu pertanyaan yang diajukan oleh sistem. Hal ini sangat mudah dilihat pada system diagnosis penyakit, dimana pakar tidak dapat mendefinisikan tentang hubungan antara gejala dengan penyebabnya secara pasti, dan pasien tidak dapat merasakan suatu gejala dengan pasti pula. Pada akhirnya ditemukan banyak kemungkinan diagnosis[20].

Sistem pakar harus mampu bekerja dalam ketidakpastian. Sejumlah teori telah ditemukan untuk menyelesaikan ketidakpastian,termasuk diantaranya probabilitas klasik (*classical probability*), probabilitas Bayes (*Bayesian probability*), teori Hartley berdasarkan himpunan klasik (*Hartley theory based on classical sets*), teori Shannon berdasarkan pada probabilitas (*Shannon theory based on probability),* teori Dempster-Shafer (*Dempster-Shafer theory*), teori fuzzy Zadeh (*Zadeh.s fuzzy theory*) dan faktor kepastian (*certainty factor*). Dalam penelitian ini yang digunakan adalah faktor kepastian. Adapun faktor kepastian merupakan suatu metode yang digunakan untuk mengukur suatu keyakinan seseorang. Inputnya adalah berupa kepastian dari pakar serta kepastian dari user[20].

2.5.1 Probabilitas Dan Certainty Factor

Certainty factor didefinisikan sebagai persamaan berikut :

$$\mathbf{CF[h,e] = MB[h,e] – MD[h,e]...(1)}$$

Keterangan :

CF[h,e]= Faktor kepastian

MB[h,e] = Measure of belief, ukuran kepercayaan atau tingkat keyakinan terhadap hipotesis (h), jika diberikan evidence (e) antara 0 dan 1

MD[h,e] = Measure of disbelief, ukuran ketidakpercayaan atau tingkat keyakinan terhadap hipotesis (h), jika diberikan evidence (e) antara 0 dan 1. Adapun beberapa kombinasi certainty factor terhadap premis tertentu[1]:

Certainty factor dengan satu premis.

$$\mathbf{CF[h,e] = CF[e] * CF[rule]= CF[user] * CF[pakar]...(2)}$$

Certainty factor dengan lebih dari satu premis.

$$\mathbf{CF[A \wedge B] = Min(CF[a],CF[b]) * CF[rule] ...(3)}$$

$$\mathbf{CF[A \vee B] = Max(CF[a],CF[b]) * CF[rule] ...(4)}$$

Certainty factor dengan kesimpulan yang serupa.

$$\mathbf{CF\ gabungan\ [CF1, CF2] = CF1 + CF2 * (1 – CF1) ...(5)}$$

Kelebihan dari metode ini adalah cocok digunakan pada sistem pakar yang mengukur sesuatu yang pasti atau tidak pasti seperti mendiagnosis penyakit dan perhitungan dari metode ini hanya berlaku untuk sekali hitung, serta hanya dapat mengolah dua data sehingga keakuratannya terjaga[1].

Bentuk dasar rumus certainty factor, adalah sebuah aturan jika E maka H seperti ditunjukkan oleh persamaan berikut :

CF (H, e) = CF (E, e)*CF (H, E) Dimana :

CF (H, E) : certainty Factor hipotesis yang dipengaruhi oleh evidence e.

CF (E, e) : Certainty Factor evidence E yang dipengaruhi oleh evidence e.

CF (H, E) : Certainty Factor hipotesis dengan asumsi evidence diketahui dengan pasti, yaitu ketika CF (E, e) = 1. Jika Evidence pada antecedent diketahui dengan pasti maka persamaannya akan menjadi : CF (E, e) = CF (H, E) [20].

Nilai CF (rule) didapat dari interpretasi “term” dari pakar, yang diubah menjadi nilai CF tertentu sesuai tabel berikut:

Tabel 2. 1 Nilai Certainty Factor

Uncertain Term	CF
Pasti tidak	-1.0
Hampir Pasti tidak	-0.8
Kemungkinan besar tidak	-0.6

Mungkin tidak	-0.4
Tidak tahu	-0.2 to 0.2
Mungkin	0.4
Kemungkinan besar	0.6
Hampir pasti	0.8
Pasti	1.0

2.5.2 Perhitungan Certainty Factor Gabungan

Secara umum, rule direpresentasikan dalam bentuk sebagai berikut :

IF E1 AND E2…… AND En THEN H (CF rule) Atau IF E1 OR E2…… OR En THEN H (CF rule)

Dimana :

E1… En : fakta – fakta (evidence) yang ada.

H : Hipotesis atau konklusi yang dihasilkan.

CF rule : Tingkat keyakinan terjadinya hipotesis H akibat adanya fakta – fakta E1…. En [20].

2.5.3 Kelebihan dan Kekurangan Metode Certainty Factor.

Kelebihan metode certainty factor adalah :

1. Metode ini cocok dipakai dalam sistem pakar yang mengandung ketidakpastian.
2. Dalam sekali proses perhitungan hanya dapat mengolah 2 data saja sehingga keakuratan data dapat terjaga.

Sedangkan kekurangan metode certainty factor adalah :

1. Pemodelan ketidakpastian yang menggunakan perhitungan metode certainty factor biasanya masih diperdebatkan.
2. Untuk data lebih dari 2 buah, harus dilakukan beberapa kali pengolahan data. Selain itu ketika ada pembaharuan API dari LINE, developer hanya perlu memperbaharui SDK sehingga meminimalisir perubahan basis kode pada aplikasi.

API Messaging memungkinkan data dilewatkan antara server aplikasi bot dan *Platform LINE. LINE platform* akan mengirimkan informasi ke server kita setiap kali ada interaksi antara user dengan akun kita, secara tidak langsung *Messaging*

API ini sebagai jembatan untuk LINE *platform* dengan aplikasi *chatbot*. Ada dua jenis interaksi, yakni:

1. User mengirimkan pesan.
2. User melakukan sebuah aksi, seperti menjadikan akun bisnis sebagai teman.

Informasi dikirim dalam format JSON melalui protokol HTTPS ke URL yang sudah didaftarkan sebagai *webhook*. Struktur data JSON yang dikirim akan bervariasi tergantung dari jenis pesan atau operasi. Setiap request menyertakan kode access token yang digunakan untuk memverifikasi bahwa request yang diterima adalah benar-benar dari *platform* LINE. Jika *access token* tidak dikenali, maka request dianggap tidak valid [21].

2.6 LINE Webhook Maker

Webhook merupakan sebuah server yang digunakan untuk menerima request dan mengirimkan response kepada komponen-komponen lainnya yang melakukan request ke *server webhook* ini. Informasi dikirim dalam format JSON melalui protokol HTTPS ke URL yang sudah didaftarkan sebagai webhook. Struktur data JSON yang dikirim akan bervariasi tergantung dari jenis pesan atau operasi. Setiap request menyertakan kode *access token* yang digunakan untuk memverifikasi bahwa request yang diterima adalah benar-benar dari platform LINE. Jika *access token* tidak dikenali, maka request dianggap tidak valid. aplikasi chatbot pada platform LINE. Komponen lainnya yang dimaksudkan di sini adalah komponen seperti layanan LINE dan layanan LINE *webhook maker* agar dapat melakukan komunikasi antar satu layanan dengan layanan lainnya. Disini penulis menggunakan webhook pada layanan LINE *webhook maker* Adalah layanan berbasis web yang dapat desain chatbot dengan menggunakan fitur yang di sediakan di dalam web tersebut diantaranya ada *type text, type sticker, type image, type video, type audio, type location, type button, type confirm, type carousel* 1 *button, type carousel* 2 *button, type carousel* 3 *button*, dan sekaligus membuat *Webhook* URL untuk menghubungkan Bot pada aplikasi LINE dengan layanan LINE *webhook maker*. penulis mengunakan layanan webhook maker untuk mendesain tampilan chatbotnya atau alur berinteraksi mengunakan keyword[21]. LINE *WebHook Maker* ini akan membantu mengkoneksikan ke heroku.

2.7 Heroku

Heroku merupakan sebuah cloud platform yang dapat menjalankan beberapa bahasa pemrograman seperti Ruby, Node.Js, Python, Java, PHP dan lainnya. Heroku termasuk ke dalam kriteria Platform *As A Service* sehingga user dapat mengembangkan, menjalankan dan mengelola aplikasi tanpa kompleksitas membangun dan memelihara infrastruktur yang biasanya terkait dengan pengembangan dan peluncuran aplikasi.

Heroku memiliki beberapa manfaat yang sangat menguntungkan bagi penggunanya seperti heroku dapat menjalankan script app tanpa memerlukan konfigurasi yang rumit dan membuat pengembang aplikasi lebih fokus pada kode dan aplikasi mereka tanpa perlu memusingkan arsitektur dan *server*[22]. Heroku digunakan untuk Platform *As A Service.*

2.8 IntelliJ IDEA

Intellij IDEA adalah Java *Integrated Development Environment* (IDE) yang dikembangkan oleh JetBrains, untuk mengembangkan perangkat lunak komputer. Intellij IDE berfungsi dalam membantu perkodingan baik dari segi navigasi, penyokong produktivitas, hingga code editor yang cerdas[19]. Untuk editornya menggunakan Intellij IDEA.

2.9 PostgreSQL

PostgreSQL adalah sistem database objek-relasional open source yang kuat yang menggunakan dan memperluas bahasa SQL yang dikombinasikan dengan banyak fitur yang dengan aman menyimpan dan skala beban kerja data yang paling rumit. Asal usul PostgreSQL berasal dari tahun 1986 sebagai bagian dari proyek POSTGRES di University of California di Berkeley dan memiliki lebih dari 30 tahun pengembangan aktif pada platform inti[23].

2.10 Pemrograman Java

JAVA adalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam(Cellular). Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di Sun Microsystems saat ini merupakan bagian dari Oracle dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin atas bawah yang minimal.

Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam p-code (bytecode) dan dapat dijalankan pada berbagai Mesin Virtual Java (JVM). Java merupakan bahasa pemrograman yang bersifat umum /non-spesifik (general purpose), dan secara khusus didesain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan dibeberapa platform sistem operasi yang berbeda, java dikenal pula dengan slogannya, “Tulis sekali, jalankan dimanapun”. Saat ini java merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi berbasis web [24].

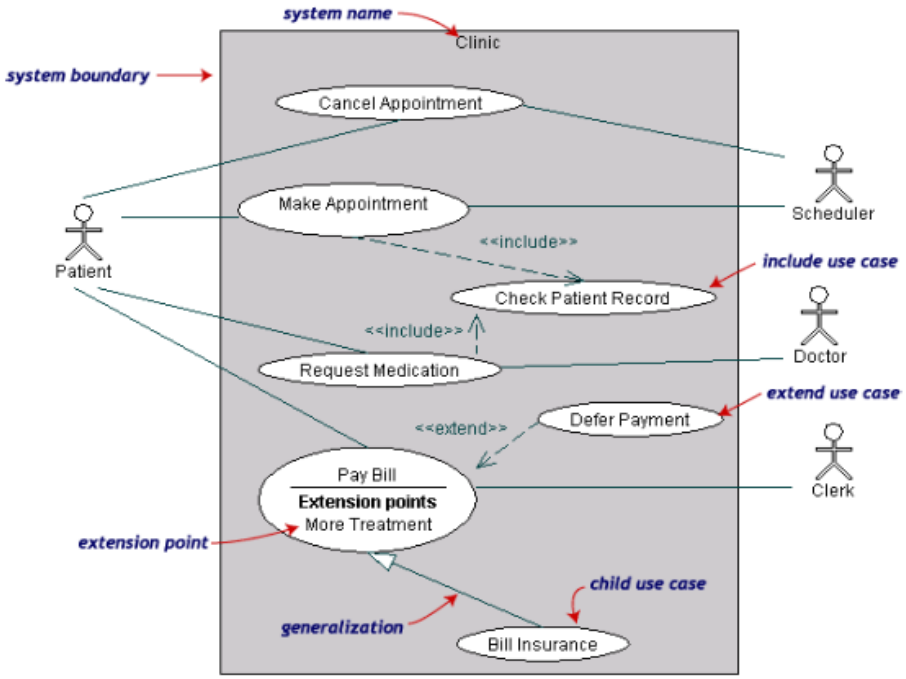
2.11 Unified Modeling Language

Unified Modelling Language (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan class dan operation dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa- bahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C[25].

Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan syntax/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML syntax mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya: Grady Booch OOD (*Object-Oriented Design*), Jim Rumbaugh OMT (*Object Modeling Technique*), dan Ivar Jacobson OOSE (*Object-Oriented Software Engineering*)[25].

2.12 Use Case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah use case merepresentasikan sebuah interaksi antara aktor dengan sistem. Use case merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, membuat sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. Use case diagram dapat sangat membantu bila kita sedang menyusun requirement sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang test case untuk semua feature yang ada pada sistem.Sebuah use case dapat meng-include fungsionalitas use case lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa use case yang di-include akan dipanggil setiap kali use case yang meng-include dieksekusi secara normal. Sebuah use case dapat di-include oleh lebih dari satu use case lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang common. Sebuah use case juga dapat meng-extend use case lain dengan behaviour-nya sendiri. Sementara hubungan generalisasi antar use case menunjukkan bahwa use case yang satu merupakan spesialisasi dari yang lain[25].



Gambar 2. 3 Contoh Use Case Diagram

2.13 Activity Diagram

Activity diagrams menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

Activity diagram merupakan state diagram khusus, di mana sebagian besar state adalah action dan sebagian besar transisi di-trigger oleh selesainya state sebelumnya (internal processing). Oleh karena itu activity diagram tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Sebuah aktivitas dapat direalisasikan oleh satu use case atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara use case menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas[25].

2.14 Sequence Diagram

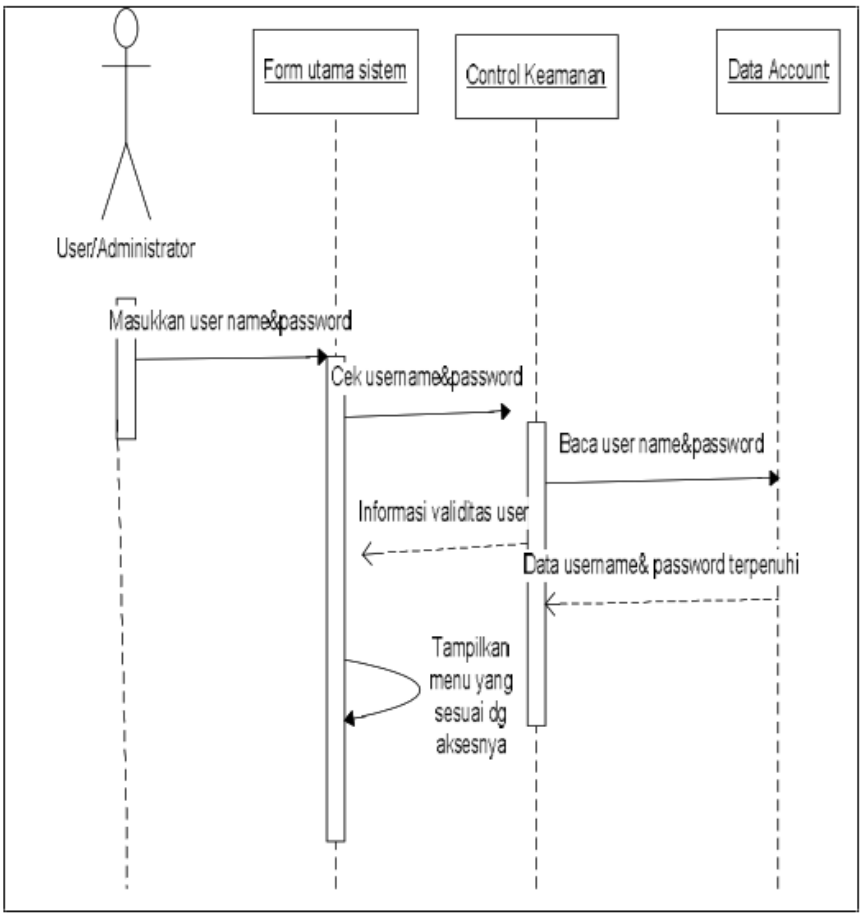
Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk user, *display*, dan sebagainya) berupa *message* yang

digambarkan terhadap waktu. *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan output tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan output apa yang dihasilkan[25].

Masing-masing objek, termasuk aktor, memiliki *lifeline* vertikal. Message digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya, message akan dipetakan menjadi operasi/metode dari class.Activation bar menunjukkan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah message. Untuk objek-objek yang memiliki sifat khusus, standar UML mendefinisikan icon khusus untuk objek boundary, controller dan persistent entity.[25]

Gambar 2. 4 merupakan contoh *sequence diagram :*



Gambar 2. 4 Contoh Sequence Diagram

2.15 Class Diagram

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. Class menggambarkan keadaan (*atribut/properti*) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (*metoda/fungsi*). Class diagram menggambarkan struktur dan deskripsi class, package dan objek serta hubungan satu sama lain seperti containment, pewarisan, asosiasi, dan lain-lain [25].

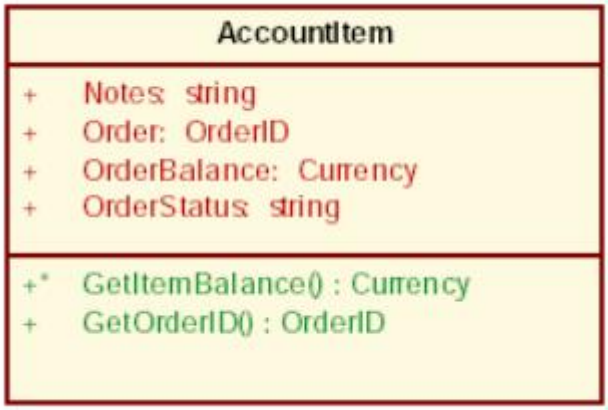
Class memiliki tiga area pokok, yaitu :

1. Nama (dan *stereotype*)
2. Atribut
3. Metode

Atribut dan metode dapat memiliki salah satu sifat berikut :

- *Private*, tidak dapat dipanggil dari luar class yang bersangkutan.
- *Protected*, hanya dapat dipanggil oleh class yang bersangkutan dan anak anak yang mewarisinya.
- *Public*, dapat dipanggil oleh siapa saja.

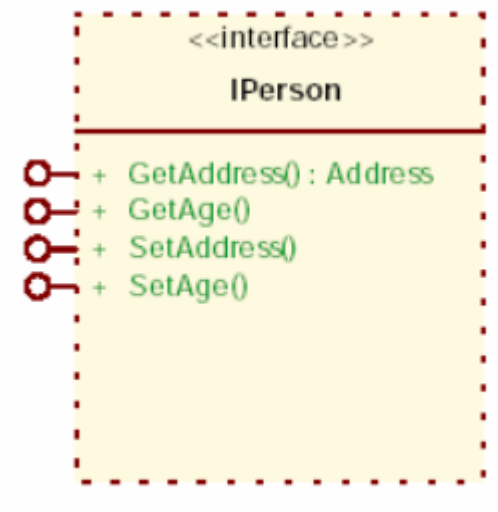
Gambar 2.5 merupakan contoh class diagram.



Gambar 2. 5 Contoh Class Diagram

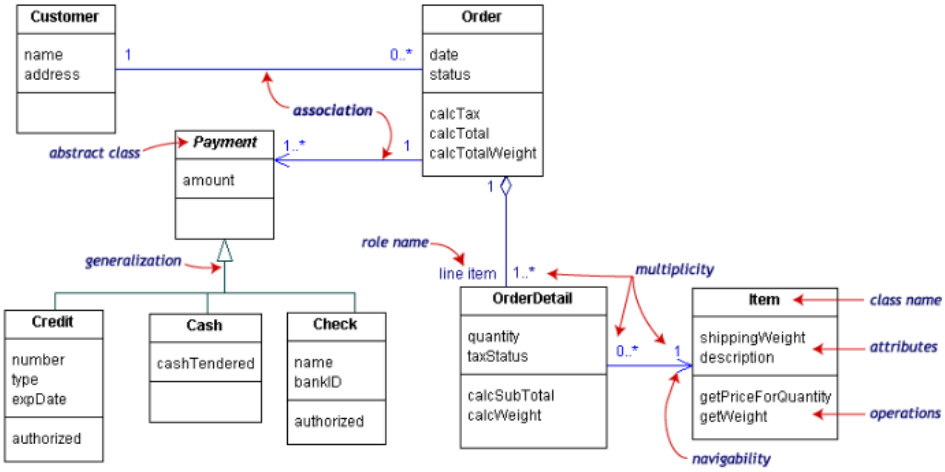
Class dapat merupakan implementasi dari sebuah *interface*, yaitu *class* abstrak yang hanya memilikimetoda. *Interface* tidak dapat langsung

diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah class. Dengan demikian *interface* mendukung resolusi metoda pada saat *run-time*[25]. Gambar 2.6 merupakan contoh class interface.



Gambar 2. 6 Contoh Class Interface

Gambar 2.7 merupakan contoh hubungan *class diagram*.



Gambar 2. 7 Contoh Hubungan Class Diagram

2.16 Black Box Testing

Pengujian black box juga disebut pengujian fungsional, yaitu teknik pengujian fungsional yang merancang kasus pengujian berdasarkan informasi dari spesifikasi. Dengan pengujian black box, penguji perangkat lunak tidak boleh (atau tidak)

memiliki akses ke kode sumber internal itu sendiri. Pengujian black box tidak memperhatikan mekanisme internal suatu sistem; ini hanya fokus pada output yang dihasilkan sebagai respons terhadap input dan kondisi eksekusi yang dipilih[26].