BAB 2
LANDASAN TEORI

2.1 Definisi Hutan

Hutan secara konsepsional yuridis dirumuskan di dalam Pasal 1 Ayat (1) Undang-undang Nomor 41 Tahun 1999 tentang Kehutanan. Menurut Undang-undang tersebut, Hutan adalah suatu kesatuan ekosistem berupa hamparan lahan berisi sumberdaya alam hayati yang didominasi pepohonan dalam persekutuan alam lingkungan, yang satu dengan yang lainnya tidak dapat dipisahkan.
Dari definisi hutan yang disebutkan, terdapat unsur-unsur yang meliputi :

a. Suatu kesatuan ekosistem
b. Berupa hamparan lahan
c. Berisi sumberdaya alam hayati beserta alam lingkungannya yang tidak dapat dipisahkan satu dengan yang lainnya.
d. Mampu memberi manfaat secara lestari.

Keempat ciri pokok dimiliki suatu wilayah yang dinamakan hutan, merupakan rangkaian kesatuan komponen yang utuh dan saling ketergantungan terhadap fungsi ekosistem di bumi. Eksistensi hutan sebagai subekosistem global menenpatikan posisi penting sebagai paru-paru dunia [1].

2.1.1 Hutan Indonesia

Indonesia adalah negara yang mempunyai area hutan yang sangat luas yang tersebar di tiap provinsi di Indonesia. Hutan di Indonesia sangat potensial, bahkan Indonesia disebut sebagai salah satu paru-paru dunia karena sebagian suplay oksigen ke seluruh dunia dihasilkan dari hutan Indonesia. Luas seluruh hutan di Indonesia adalah133.300.543,98 ha. Ini mencakup kawasan suaka alam, hutan lindung, dan hutan produksi.

Namun menurut data laju *deforestasi* (kerusakan hutan) periode 2003-2006 yang dikeluarkan oleh Departemen Kehutanan, laju *deforestasi* di Indonesia mencapai 1,17 juta hektar pertahun, ini disebabkan karena beberapa hal

diantaranya karena penjarahan, kegiatan industri, pembalakan liar, alih funsi hutan dan masih banyak lagi faktor lainnya.

Deforestasi (kerusakan hutan) memberikan dampak yang signifikan bagi masyarakat dan lingkungan alam di Indonesia, yaitu dapat mengakibatkan penurunan kualitas lingkungan yang pada akhirnya meningkatkan peristiwa bencana alam, seperti tanah longsor dan banjir. Dampak buruk lain akibat kerusakan hutan adalah terancamnya kelestarian satwa dan flora di Indonesia utamanya flora dan fauna endemic [3].

2.2 *Game*

Game merupakan salah satu media hiburan yang paling populer untuk semua kalangan usia. *Game* dapat dimainkan dengan aturan tertentu sehigga ada yang menang dan ada yang kalah, biasanya dalam konteks tidak serius atau dengan tujuan refreshing. Suatu cara belajar yang digunakan dalam menganalisa interaksi antara sejumlah pemain maupun perorangan yang menunjukkan strategi-strategi yang rasional. Sejak pertama kali ditemukan sampai saat sekarang, teknologi *game* telah mengalami kemajuan yang terbilang pesat. Hal ini dapat dilihat dengan berkembangnya jenis, produk, alat, dan jenis interaksi *game* dengan penggunaan yang semakin beragam.

2.2.1 Pengertian *Game*

Game (Permainan) merupakan bagian dari bermain dan bermain juga bagian dari permainan keduanya saling berhubungan. *Game* adalah kegiatan yang kompleks yang didalamnya terdapat peraturan, permainan dan budaya. Sebuah *Game* adalah sebuah sistem dimana pemain terlibat dalam konflik buatan, disini pemain berinteraksi dengan sistem dan konflik dalam permainan merupakan rekayasa atau buatan, dalam permainan terdapat peraturan yang bertujuan untuk membatasi perilaku pemain dan menentukan permainan. *Game* bertujuan untuk menghibur, biasanya *game* banyak disukai oleh anak – anak hingga orang dewasa. *Game* sebenarnya penting untuk perkembangan otak, untuk meningkatkan konsentrasi dan melatih untuk memecahkan masalah dengan tepat dan cepat

karena dalam *game* terdapat berbagai konflik atau masalah yang menuntut kita untuk menyelesaikannya dengan cepat dan tepat. Tetapi *game* juga bisa merugikan karena apabila kita sudah kecanduan *game* kita akan lupa waktu dan akan mengganggu kegiatan atau aktifitas yang sedang kita lakukan.

Berikut adalah Pengertian *Game* menurut para ahli :

1. Menurut David Parlett, *Game* adalah sesuatu yang memiliki "akhir dan cara mencapainya": artinya ada tujuan, hasil dan serangkaian peraturan untuk mencapai keduanya
2. Menurut Roger Caillois seorang sosiolog Perancis, dalam bukunya yang berjudul *Les jeux et les hommes* menyatakan *game* adalah aktivitas yang mencakup karakteristik berikut:
   a. *Fun* (bebas bermain adalah pilihan bukan kewajiban)
   b. *Separate* (terpisah).
   c. *Uncertain, non-productive, governed by rules* (ada aturan).
   d. *Fictitious* (pura-pura).
3. Menurut Clark C.Abt, *Game* adalah kegiatan yang melibatkan keputusan pemain, berupaya mencapai tujuan dengan;dibatasi oleh konteks tertentu (misalnya, dibatasi oleh peraturan).
4. Menurut Bernard Suits, *Game* adalah upaya sukarela untuk mengatasi rintangan yang tidak perlu.
5. Menurut Greg Costikyan, *Game* adalah sebentuk karya seni di mana peserta, yang disebut Pemain, membuat keputusan untuk mengelola sumberdaya yang dimilikinya melalui benda di dalam *game* demi mencapai tujuan.
6. Dari buku *Rules of Play* karya Katie Salendan Eric Zimmerman, *Game* adalah sistem tempat pemain melakukan konflik bohongan, ditentukan oleh aturan, yang memberi hasil terukur.
7. Menurut Hans Daeng (dalam Andang Ismail, 2009: 17) permainan adalah bagian mutlak dari kehidupan anak dan permainan merupakan bagian integral dari proses pembentukan kepribadian anak.
8. Menurut Andang Ismail(2009: 26) menuturkan bahwa permainan ada dua pengertian. Pertama, permainan adalah sebuah aktifitas bermain yang murni

mencari kesenangan tanpa mencari menang atau kalah. Kedua permainan diartikan sebagai aktifitas bermain yang dilakukan dalam rangka mencari kesenangan dan kepuasan, namun ditandai pencarian menang – kalah.

9. Menurut Kimpraswil (dalam As'adi Muhammad, 2009: 26) mengatakan bahwa definisi permainan adalah usaha olah diri (olah pikiran dan olah fisik) yang sangat bermanfaat bagi peningkatan dan pengembangan kinerja, dan prestasi dalam melaksanakan tugas dan kepentingan organisasi dengan lebih baik.
10. Menurut Joan Freeman dan Utami Munandar (dalam Andang Ismail, 2009: 27) mendefinisikan permainan sebagai suatu aktifitas yang membantu anak mencapai perkembangan yang utuh, baik fisik, intelektual, sosial, moral, dan emosional.
11. Menurut Agustinus Nilwan dalam bukunya "Pemrograman Animasi dan *Game* Profesional" terbitan Elex Media Komputindo, *game* merupakan permainan komputer yang dibuat dengan teknik dan metode animasi.Jika ingin mendalami pengunaan animasi haruslah memahami pembuatan *game*. Atau jika ingin membuat *game*, maka haruslah memahami teknik dan metode animasi, sebab keduanya saling berkaitan [4].

Game bertujuan untuk menghibur, biasanya *game* banyak disukai oleh anak anak hingga orang dewasa. *Game* sebenarnya penting untuk perkembangan otak, untuk meningkatkan konsentrasi dan melatih untuk memecahkan masalah dengan tepat dan cepat karena dalam *game* terdapat berbagai konflik atau masalah yang menuntut kita untuk menyelesaikannya dengan cepat dan tepat. Tetapi *game* juga bisa merugikan karena apabila kita sudah kecanduan *game* kita akan lupa waktu dan akan mengganggu kegiatan atau aktifitas yang sedang kita lakukan.

2.2.2 Sejarah Perkembangan *Game*

Game pertama kali diperkenalkan pada tahun 1971 sebagai medium hiburan yang bersifat komersial. Pada akhir tahun 1970an dan awal tahun 1980an, *video games* menjadi sebuah dasar yang penting bagi industri hiburan di Amerika Serikat, Jepang, dan Eropa. Sejak masa kejatuhan dalam bidang industri pada

tahun 1983 dan kebangkitannya dua tahun kemudian, industri *video game* telah mengalami perkembangan selama dua dekade dan menjadi industri yang berharga $10 miliar, yang menjadi saingan dengan industri '*motion picture*' sebagai industri hiburan yang paling berprofit besar. Pada tahun 1952, A.S. Douglas menulis untuk gelar PhD di University of Cambridge jurusan Human-Computer Interraction. Douglas menciptakan permainan komputer grafis yang diberi nama Tic-Tac-Toe. Permainan ini diprogram pada komputer EDSAC vacuum-tube yang memiliki tampilan tabung sinar katode.William Higinborham menciptakan *video game* pertama pada tahun 1958. Permainan ini diberi judul "*Tennis for Two*", diciptakan dan dimainkan di osiloskop Brookhaven National Laboratory. Pada tahun 1962, Steve Russell menciptakan SpaceWar! Permainan ini merupakan permainan pertama yang dimaksudkan untuk pemakaian komputer. Russel menggunakan MIT PDP-1 komputer mainframe untuk meciptakan permainan ini.Pada 1967, Ralph Baer menciptakan *video game* pertama yang dimaikan dengan televisi, sebuah permainan yang diberi nama Chase. Ralph Baer akhirnya menjadi bagian dari Sanders Associates, sebuah perusahaan elektronik militer.Ralph mencetuskan idenya untuk pertama kali pada 1951, saat sedang bekerja untuk Loral, sebuah perusahaan komputer.Pada 1971, Nolan Bushnell bersama dengan Ted Dabney, menciptakan permainan arcade yang pertama. Permainan ini diberi nama Computer Space, yang diciptakan berdasarkan pada permainan SpaceWar! Permainan arcade Pong diciptakan oleh Nolan Bushnell (dengan bantuan Al Alcorn) setahun kemudian pada tahun 1972. Nolan Bushnell dan Ted Dabney memulai dengan Atari Computer pada tahun yang sama. Pada tahun 1975, Atari kembali meluncurkan Pong sebagai home *video game*.Tahun 1972, hiburan pertama *video game* komersial dapat dimainkan di rumah, Odyssey, diluncurkan oleh Magnavox dan didesain oleh Ralph Baer. Permainan mesin ini aslinya didesain pada saat Ralph Baer masih berada di Sanders Associates pada tahun 1966, Baer mengurus untuk mendapatkan ijin legalnya setelah ditolak oleh Sanders Associates. Permainan Odyssey deprogram dengan dua belas permainan. Pada tahun 1976, Fairchild meluncurkan programmable home *game* yang pertama dan diberi nama Fairchild Video Entertainment System atau Channel F. Channel F

merupakan salah satu dari electronic system yang pertama menggunakan microchip yang diciptakan oelh Robert Noyce untuk Fairchild Semiconductor Corporation [5].

2.2.3 Klasifikasi *Game*

2.2.3.1 Berdasarkan Jenis *Platform* yang digunakan

1. *Arcade games*

   Arcade games atau kalau di Indonesia sering di sebut dengan ding-dong, biasanya berada di daerah/tempat khusus dan memiliki box atau mesin yang memang khusus di design untuk jenis *video games* tertentu dan tidak jarang bahkan memiliki fitur yang dapat membuat pemainnya lebih merasa “masuk” dan “menikmati”, seperti pistol, kursi khusus, sensor gerakan, sensor injakan dan stir mobil (beserta transmisinya tentunya).

2. *PC Games*

   PC games adalah *video games* yang dimainkan menggunakan *Personal Computers*.

3. *Console games*

   Console games adalah video *games* yang dimainkan menggunakan console tertentu, seperti Playstation 2, Playstation 3, XBOX 360, dan Nintendo Wii.

4. *Handheld games*

   Handheld games adalah *video games* yang dimainkan di *console* khusus *video game* yang dapat dibawa kemana-mana atau disebut juga dengan *portable console*, contohnya seperti Nintendo DS dan Sony PSP.

5. *Mobile games*

   Mobile games adalah *video games* yang dapat dimainkan atau khusus dimainkan untuk mobile phone atau PDA.

2.2.3.2 Berdasarkan *Genre* Permainannya

1. RTS (*Real Time Strategy*)

Real Time Strategy adalah genre suatu permainan komputer yang memiliki ciri khas berupa permainan perang yang tiap pemainnya memiliki suatu negara, negara tersebut di kelola dalam hal pengumpulan sumberdaya (alam, manusia, ekonomi), pengaturan strategi pasukan-pasukan tempur, diplomasi dengan negara tetanga, peningkatan ekonomi, pengembangan keyakinan, pengembangan pendidikan dari negara primitif menuju peradaban modern. RTS dibedakan dari *turn-based strategy* dimana dalam RTS permainan tidak mengenal giliran. Setiap pemain dapat mengatur/memerintah pasukannya dalam waktu apapun. Dalam RTS, tema permainan dapat berupa sejarah (misalnya seri Age of Empires), fantasi (misalnya Warcraft) dan fiksi ilmiah (misalnya Star Wars).

2. FPS (*First Person Shooter*)

Jenis permainan tembak-menembak dengan tampilan pada layar pemain adalah sudut pandang tokoh karakter yang dimainkan, tiap tokoh karakter mempunyai kemampuan yang berbeda-beda dalam tingkat akurasi menembak, reflek menembak,dll. Permainan ini dapat melibatkan banyak orang. Permainan ini bisa berupa misi melumpuhkan penjahat maupun alien, kadang juga sejumlah pemain dibagi beberapa tim yang bertugas melumpuhkan tim lainnya, sebelum dilumpuhkan. Ciri utama lain adalah penggunaan senjata genggam jarak jauh.

Contoh: Duke Nukem 3D, Quake, Blood, Unreal, Unreal Tournament, seri Half-Life, Counter-Strike, seri Halo, Perfect Dark, TimeSplitters, Call of Duty, System Shock, dan GoldenEye 007.

3. RPG (*Role Playing Game*)

Adalah sebuah permainan yang para pemainnya memainkan peran tokoh-tokoh khayalan dan berkolaborasi untuk merajut sebuah cerita bersama. Para pemain memilih aksi tokok-tokoh mereka berdasarkan karakteristik tokoh tersebut, dan keberhasilan aksi mereka tergantung dari sistem peraturan permainan yang telah ditentukan. Asal tetap mengikuti peraturan yang ditetapkan, para pemain bisa berimprovisasi membentuk arah dan hasil akhir permainan ini.

Dalam sebuah permainan RPG, jarang ada yang “kalah” atau “menang”. Ini membuat permain RPG berbeda dari jenis permainan papan lainnya seperti Monopoli atau Ular Tangga, permainan kartu, olah raga, dan permainan lainnya.

Seperti sebuah novel atau film, permainan RPG mempunyai daya tarik karena permainan-permainan ini mengajak para pemain untuk menggunakan imajinasi mereka. RPG biasa lebih mengarah ke kolaborasi sosial daripada kompetisi. Pada umumnya dalam RPG, para pemain tergabung dalam satu kelompok.

Ada 2 jenis *game* RPG, antara lain :

a. LARP (*Live-Action Role-Playing*)

   Permainan RPG di mana para pemain bisa melakukan gerakan fisik tokohnya oleh si pemain sendiri, biasanya para pemain memakai kostum dan menggunakan alat-alat yang sesuai dengan tokoh, dunia dan cerita yang dia mainkan.

b. MMORPG (*Massively Multiplayer Online Role-Playing Game*)

   Adalah permainan *role-playing game* (RPG) yang melibatkan ribuan pemain untuk bermain bersama dalam dunia maya yang terus berkembang pada saat yang sama melalui media internet dan jaringan.

   Contoh: World of Warcraft, The Lord of the Rings Online: Shadows of Angmar, Final Fantasy, Ragnarok, DoTA.

4. *Life Simulation Games*

Permainan simulasi kehidupan ini meliputi kegiatan individu dalam sebuah tokoh karakter. Dalam memainakn tokoh karakter tersebut pemain bertanggung jawab atas inteligen serta kemampuan fisik dari tokohnya tersebut. Tokoh karakter tersebut memerlukan kebutuhan layaknya manusia seperti kegiatan belajar, bekerja, belanja, bersosialisasi, memelihara hewan, memelihara lingkungan dan lain-lain. Lawan mainnya bisa berupa pemain lain yang memainkan karakter sebagai tetangga maupun komputer dengan kecerdasan buatan tingkat tinggi.

Contoh: SimLife, Second Life.

5. *Construction and Management Simulation Games*

Permainan yang mensimulasikan proyek membangun sebuah kota, pemain di haruskan membangun sebuah kota lengkap dengan fasilitas publik maupun fasilitas pemerintah seperti gedung, alat transportasi publik, taman, sekolah, rumah sakit, tempat beribadah, pabrik, bandara, stasiun, bank dan bangunan lainnya lainnya. saat membangun kota tersebut pemain juga harus memperhatikan

sumber daya ekonomi, kenyamanan para penduduknya dalam beraktifitas yang mungkin terganggu saat pembuatan jembatan penyebrangan dan lain-lain.

Contoh: SimCity, Caesar.

6. *Vehicle Simulation*

Jenis permainan ini mensimulasi pengoperasian beberapa kendaraan, kendaraan bisa berupa pesawat terbang, pesawat tempur, kereta, kendaraan perang, maupun kendaraan konstruksi.

Contoh: Flight Gear, Tram, Orbiter.

7. Action Games

Permainan jenis ini sangat berkaitan dengan tantangan fisik, seperti ketangkasan, reflek dari pemain. Dalam permainan ini pemain mengendalikan seorang tokoh karakter, tokoh karakter yang dimainkan bisa dihadapkan dengan tokoh karakter lain yang bertarung berhadapan bisa juga menjalankan sebuah misi yang banyak rintangan, mengumpulkan objek tertentu, mengalahkan musuh, maupun menyelamatkan karakter lainnya.

Contoh: Street Fighter.

8. *Adventure Games*

Permainan yang menggunakan tokoh karakter fiksi yang bertugas mengekplorasi memecahkan sebuah misteri atau kasus, memburu harta karun, maupun menyelamatkan tokoh karakter buatan. Banyak dari *game* ini diangkat dari sebuah novel populer maupun film biskop.

9. *Action-Adventure Games*

Permainan bertualngan yang dikombinasikan dengan aksi bertarung, menghadapi rintangan maupun memecahkan teka-teki.

Contoh: Tomb Raider, Indiana Jones.

10. *Manager Simulation*

Permainan yang mensimulasikan menjadi seorang manager dalam sebuah klub sepakbola.

Contoh: Championship Manager.

11. *Racing*

Adalah salah satu jenis permainan yang bernuansakan balap kendaraan. Disini kita bisa mengontrol sendiri kendaraan yang kita miliki maupun kita bisa menongtrolnya bersama-sama dengan *partner* kita.

Contoh: CTR (Crash Team Racing), Need For Speed.

12. *Sport*

Game bergenre ini bertemakan olahraga seperti pada umumnya, Kita dapat mengontrol manusia yang diibaratkan kita sendiri di dalam *game* tersebut. Yang membedakan adalah kita disini tidak benar-benar berolahraga, melainkan tokoh yang kita mainkanlah yang melakukan kegiatan tersebut.

Contoh: PES 2010, Winning Eleven

13. *Art*

Game yang bergenre ini sangat jarang sekali ditemukan, dikarenakan mungkin masih kurang peminatnya. Sehingga masih kurang perusahaan pembuat *game* yang mengeluarkannya.

14. *Education*

Game yang telah dirancang untuk media pembelajaran tentang topik tertentu, *game* ini dibangun dengan memadukan antara permainan yang menyenagkan dan menghibur dan pembelajaran, sehingga *game* ini mempunyai dua manfaat sekaligus yaitu menghibur dan memberikan pembelajaran bagi para penggunaya [5].

2.3 *Game Tower Defense*

Tower defence adalah sebuah *game real time strategy* yang merupakan MOD (modifikasi) dari *game* strategi populer Warcraft 3 yang diproduksi oleh Blizzard. Blizzard merupakan sebuah perusahaan pembuat *game* ternama yang telah berhasil meluncurkan berbagai *game* yang sering kali merebut hati para *gamer* di dunia.

Inti dari permainan *Tower Defence* ini adalah *gamer* ditugaskan untuk melindungi *goal point* (basis pertahanan) dari serbuan musuh yang dikendalikan oleh AI (*artificial intelligence*). Musuh ini akan datang secara bergelombang dan dalam jumlah yang banyak. *Gamer* melindungi target dengan cara mendirikan

tower-tower atau menara-menara pertahanan yang akan menembaki setiap musuh yang mendekati target.

Permainan ini dibagi dalam tingkatan-tingkatan yang dimulai dari tingkatan rendah, menegah, hingga tingkatan tinggi sesuai dengan tingkat keberhasilan *gamer* dalam mempertahankan *goal point* setiap tingkatannya. Dan semakin tinggi tingkatan yang dicapai, musuh yang menyerbu semakin kuat dan banyak. Serta uang yang didapatkan dari membunuh musuh-musuh ini pun semakin meningkat untuk digunakan mendirikan menara-menara pertahanan yang lebih canggih dan kuat.

Permainan ini menuntut ketangkasan, perhitungan, strategi, menempatkan berbagai macam *tower* dengan harga yang beragam. Adapun uang yang digunakan untuk membeli *tower* didapatkan dari hasil membunuh setiap musuh yang ada. Permaianan dinyatakan sukses apabila pemain dapat mempertahankan darah atau nyawanya sampai waktu yang ditentukan habis atau berdasarkan jumlah musuh yang dibunuh.

2.4 *Artificial Intelligence* (Kecerdasan Buatan)

Sebagian kalangan menerjemahkan *Artificial Intelligence* sebagai kecerdasan buatan, kecerdasan artifisial, intelijensia artifisial, atau intelejensia buatan. Pada bukunya [2], Suyanto sengaja tidak menerjemahkan istilah *Artificial Intelligence* ke bahasa Indonesia karena istilah tersebut sudah sangat akrab bagi orang Indonesia. Begitu juga dengan singkatan istilah tersebut, yaitu AI, sudah sangat melekat di berbagai media ilmiah maupun non ilmiah.

Para ahli mendefinisikan AI secara berbeda-beda tergantung pada sudut pandang mereka masing-masing. Ada yang focus pada logika berpikir manusia saja, tetapi ada juga yang mendefinisikan AI secara lebih luas pada tingkah laku manusia. Stuart Russel dan Peter Norvig mengelompokkan definisi AI, yang diperoleh dari beberapa *textbook* berbeda, ke dalam empat kategori [11], yaitu :

A. *Thinking humanly* : *the cognitive modeling approach*

Pendekatan ini dilakukan dengan dua cara sebagai berikut :

1. Melalui introspeksi : mencoba menangkap pemikiran-pemikiran kita sendiri pada saat kita berpikir. Tetapi, seorang psikolog Barat mengatakan *"how do you know that you understand?"* Bagaimana anda tahu bahwa anda mengerti? Karena pada saat anda menyadari pemikiran anda. Sehingga definisi ini terkesan mengada-ngada dan tidak mungkin dilakukan.
2. Melalui eksperimen-eksperimen psikologi.

B. *Acting humanly* : *the Turing test approach*

Pada tahun 1950, Alan Turing merancang suatu ujian bagi komputer berintelijensia untuk menguji apakah komputer tersebut mampu mengelabui seorang manusia yang menginterogasinya melalui *teletype* (komunikasi berbasis teks jarak jauh). Jika *interrogator* tidak dapat membedakan yang d2nterogasi adalah manusia atau komputer, maka komputer berintelijensia tersebut lolos dari *Turning test*. Komputer tersebut perlu memiliki kemampuan : *Natural Language Processing, Knowledge Representation, Automated Reasoning, Machine Learning, Computer Vision, Robotics*. *Turing test* sengaja menghindari interaksi fisik antara *interrogator* dan komputer karena simulasi fisik manusia tidak memerlukan intelijensia.

C. *Thinking rationally* : *the laws of thought approach*

Terdapat dua masalah dalam pendekatan ini, yaitu :

1. Tidak mudah untuk membuat pengetahuan informal dan menyatakan pengetahuan tersebut ke dalam *formal term* yang diperlukan oleh notasi logika, khususnya ketika pengetahuan tersebut memiliki kepastian kurang dari 100%.
2. Terdapat perbedaan besar antara dapat memecahkan masalah "dalam prinsip" dan memecahkannya "dalam dunia nyata".

D. *Acting rationally* : *the rational agent approach*

Membuat inferensi yang logis merupakan bagian dari suatu *rational agent.* Hal ini disebabkan satu-satunya cara untuk melakukan aksi secara rasional adalah dengan menalar secara logis. Dengan menalar secara logis, maka bisa didapatkan kesimpulan bahwa aksi yang diberikan akan mencapai tujuan atau tidak. Jika

mencapai tujuan, maka *agent* dapat melakukan aksi berdasarkan kesimpulan tersebut.

Thinking humanly dan *acting humanly* adalah dua definisi dalam arti yang sangat luas. Sampai saat ini, pemikiran manusia yang diluar rasio, yakni reflex dan intuitif (berhubungan dengan perasaan), belum dapat ditirukan sepenuhnya oleh komputer. Dengan demikian, kedua definisi ini dirasa kurang tepat untuk saat ini. Jika kita menggunakan definisi ini, maka banyak produk komputasi cerdas saat ini yang tidak layak disebut sebagai produk AI.

Definisi *thinking rationally* terasa lebih sempit daripada *acting rationally*. Oleh karena itu, definisi AI yang paling tepat untuk saat ini adalah *acting rationally* dengan pendekatan *rational agent*. Hal ini berdasarkan pemikiran bahwa komputer bisa melakukan penalaran secara logis dan juga bisa melakukan aksi secara rasional berdasarkan hasil penalaran tersebut.

2.4.1 *Fuzzy Logic* (Logika Fuzzy)

2.4.1.1 Pengertian

Fuzzy secara bahasa diartikan sebagai kabur atau samar-samar. Suatu nilai dapat bernilai besar atau salah secara bersamaan. 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).

Algoritma *Fuzzy* merupakan seuatu logika yang memiliki nilai kekaburan atau kesamaran (*fuzzyness*) antara benar atau salah. Dalam teori Algoritma *Fuzzy* satu nilai bias bernilai benar atau salah secara bersama. Namun berapa besar keberadaan dan kesalahan suatu tergantung pada bobot keanggotaan yang dimilikinya. Algoritma *Fuzzy* memiliki derajat keanggotaan dalam rentang 0 hingga 1. Berbeda dengan logika digital yang hanya memiliki dua nilai 1 atau 0. Algoritma *Fuzzy* digunakan untuk menterjemahkan suatu besaran yang diekspresikan menggunakan bahasa (*linguistic*), misalkan besaran kecepatan laju kendaraan yang diekspresikan dengan pelan, agak cepat, cepat, dan sangat cepat. Dan Algoritma *Fuzzy* menunjukan sejauh mana suatu nilai itu benar dan sejauh mana suatu nilai itu salah. Tidak seperti logika klasik (*crisp*) tegas, suatu nilai

hanya mempunyai 2 kemungkinan yaitu merupakan suatu anggota himpunan atau tidak. Derajat keanggotaan 0 (nol) artinya nilai bukan merupakan anggota himpunan dan 1 (satu) berarti nilai tersebut adalah anggota himpunan.

Algoritma *Fuzzy* adalah suatu cara yang tepat untuk memetakan suatu ruang *input* kedalam suatu ruang *output*, mempunyai nilai kontinyu. *Fuzzy* dinyatakan dalam derajat dari suatu keanggotaan dan derajat dari kebenaran. Oleh sebab itu sesuatu dapat dikatakan sebagian benar dan sebagian salah pada waktu yang sama.

Algoritma *Fuzzy* memungkinkan nilai keanggotaan antara 0 dan 1, tingkat keabuan dan juga hitam dan putih, dan dalam bentuk *linguistik*, konsep tidak pasti seperti "sedikit", "sedang" dan "banyak" [12].

2.4.1.2 Konsep Dasar Logika Fuzzy

Teori himpunan *fuzzy* diperkenalkan pertama kali oleh Lotfi A. Zadeh pada tahun 1965. Ada beberapa alasan mengapa orang menggunakan logika *fuzzy*, antara lain :

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 *nonlinear* 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.

Dalam logika *fuzzy* dikenal berhingga keadaan dari nilai “0” sampai ke nilai “1”. Logika *fuzzy* tidak hanya mengenal dua keadaan tetapi juga mengenal sejumlah keadaan yang berkisar dari keadaan salah sampai keadaan benar.

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

1. Variabel *Fuzzy*

   Variabel *Fuzzy* merupakan variabel yang hendak dibahas dalam suatu sistem *Fuzzy*.
2. Himpunan *Fuzzy*

   Himpunan *Fuzzy* adalah himpunan yang tiap elemennya mempunyai derajat keanggotaan tertentu terhadap himpunannya.

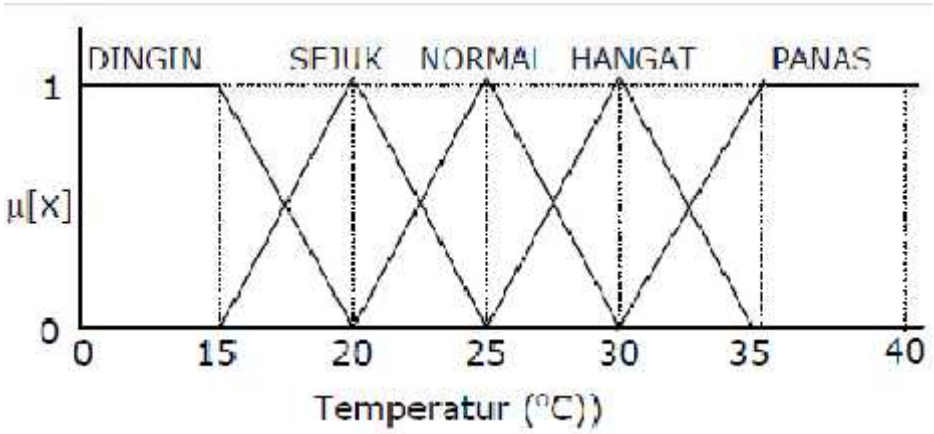
   Himpunan *Fuzzy* memiliki dua atribut, yaitu :

   a. *Linguistik*, yaitu penamaan suatu grup yang mewakili suatu keadaan atau kondisi tertentu dengan menggunakan bahasa alami.

   b. *Numeris*, yaitu suatu nilai (angka) yang menunjukkan ukuran dari suatu variabel.

   Contoh Himpunan *Fuzzy* :

   Misalkan Variabel temperatur, terbagi menjadi 5 himpunan *fuzzy*, yaitu: DINGIN, SEJUK, NORMAL, HANGAT, dan PANAS.



Gambar 2. 1 Variabel Temperatur

Pada himpunan tegas (*crisp*), nilai keanggotaan suatu item x dalam satu himpunan A, yang sering ditulis dengan μA[x], memiliki 2 kemungkinan, yaitu:

a. Satu (1), yang berarti bahwa suatu item menjadi anggota dalam suatu himpunan, atau
b. Nol (0), yang berarti bahwa suatu item tidak menjadi anggota dalam suatu himpunan.

Contoh :

Jika diketahui:

S = { 1, 2, 3, 4, 5, 6} adalah semesta pembicaraan.

A = {1, 2, 3}

B = {3, 4, 5}

bisa dikatakan bahwa:

Nilai keanggotaan 2 pada himpunan A, μA[2]= 1, karena 2 anggota A.

Nilai keanggotaan 3 pada himpunan A, μA[3]= 1, karena 3 anggota A.

Nilai keanggotaan 4 pada himpunan A, μA[4]= 0, karena 4 bukan anggota A.

Nilai keanggotaan 2 pada himpunan B, μB[2]= 0, karena 2 anggota B.

Nilai keanggotaan 3 pada himpunan B, μB[3]= 1, karena 3 anggota B.

3. Semesta Pembicaraan

Semesta pembicaraan adalah suatu keseluruhan nilai yang diperbolehkan untuk dioperasikan dalam suatu variabel *Fuzzy*. Nilai semesta pembicaraan dapat berupa bilangan positif atau bilangan negatif.

4. *Domain*

Domain himpunan *Fuzzy* adalah keseluruhan nilai yang diijinkan dalam semesta pembicaraan dan boleh dioperasikan dalam suatu himpunan *Fuzzy*. Nilai *domain* dapat berupa bilangan positif maupun bilangan negatif.

2.4.1.3 Karakteristik Logika *Fuzzy*

Logika *Fuzzy* memiliki beberapa karakteristik yaitu himpunan *Fuzzy* dan fungsi keanggotaan[10].

1. Himpunan *Fuzzy*

Pada logika *boolean*, sebuah individu dipastikan sebagai anggota dari salah satu himpunan saja, sedangkan pada himpunan *fuzzy* sebuah individu dapat masuk pada dua himpunan yang berbeda. Seberapa besar eksistensinya dalam himpunan tersebut dapat dilihat pada nilai keanggotaannya.

Himpunan *fuzzy A* pada semesta *X* dinyatakan sebagai himpunan pasangan berurutan (*set of ordered pairs*) baik diskrit maupun kontinu.

$$A = \{(x, \mu_A(x)) | x \in X\}$$

Dimana $\mu_A(x)$ fungsi keanggotaan himpunan *fuzzy A*. Fungsi keanggotaan memetakan setiap $x \in X$ pada suatu nilai antara [0,1] yang disebut derajat keanggotaan (*membership grade* atau *membership value*).

2. Fungsi Keanggotaan (*Membership Function*)

Fungsi keanggotaan adalah suatu kurva yang menunjukkan pemetaan titik-titik *input* data ke dalam nilai keanggotaannya (disebut juga dengan derajat keanggotaan) yang memiliki interval antara 0 sampai 1. Salah satu cara yang dapat digunakan untuk mendapatkan nilai keanggotaan adalah dengan melalui pendekatan fungsi. Derajat keanggotaan dalam himpunan (*degree of membership*) dilambangkan dengan μ.

Dalam sistem *fuzzy* banyak dikenal bermacam-macam fungsi keanggotaan (*membership function*). Beberapa fungsi keanggotaan yang sering digunakan adalah:

a. Fungsi keanggotaan Linear

$$\mu[x] = \begin{cases} 0; & x \le a \text{ atau } x \ge c \\ (x - a) / (b - a) & a \le x \le b \\ 1; & x \ge b \end{cases}$$

b. Fungsi keanggotaan segitiga

$$\mu[x] = \begin{cases} 0; & x \le a \text{ atau } x \ge c \\ (x - a) / (b - a); & a \le x \le b \\ (c - x) / (c - b); & b \le x \le c \end{cases}$$

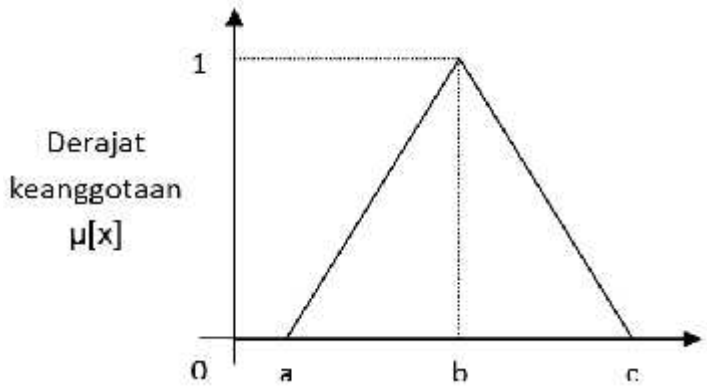
c. Fungsi Keanggotaan Trapesium

$$\mu[x] = \begin{cases} 0 & x \le a \text{ atau } x \ge d \\ (x - a) / (b - a) & a < x < b \\ 1 & b \le x \le c \\ (d - x) / (d - c) & c < x < d \end{cases}$$

Dalam Aplikasi *Game Tower Defense* Selamatkan Hutan Indonesia, fungsi keanggotaan yang dipakai adalah Representasi Kurva Segitiga dan Representasi Kurva Bahu.

a. Representasi Kurva Segitiga

Kurva segitiga pada dasarnya merupakan gabungan antara dua garis.



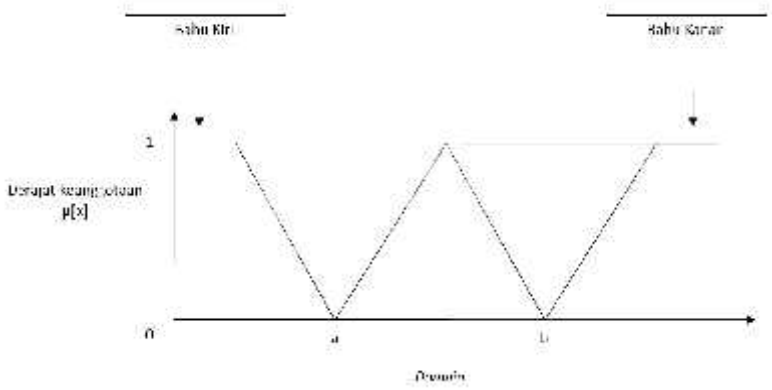
Gambar 2. 2 Fungsi Keanggotaan Kurva Segitiga

Fungsi keanggotaan:

$$\mu[x] = \begin{cases} 0; & x \le a \text{ atau } x \ge c \\ (x - a) / (b - a); & a \le x \le b \\ (c - x) / (c - b); & b \le x \le c \end{cases}$$

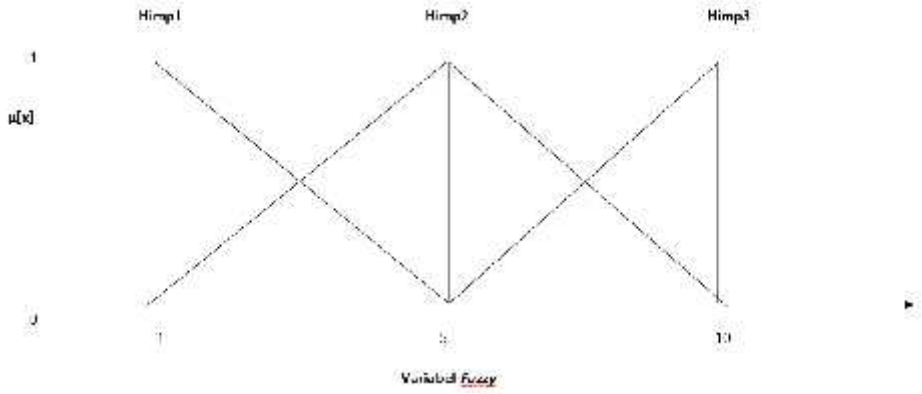
b. Representasi Kurva Bahu

Kurva bahu merupakan daerah yang terletak di tengah-tengah suatu variabel yang direpresentasikan dalam bentuk segitiga, pada sisi kanan dan sisi kirinya akan naik dan turun. Himpunan *fuzzy* bahu digunakan untuk mengakhiri variabel suatu daerah *fuzzy*.



Gambar 2. 3 Fungsi Keanggotaan Kurva Bahu

Fungsi keanggotaan pada kurva segitiga dan fungsi keanggotaan pada kurva bahu dapat dilihat pada gambar 2.4



Gambar 2. 4 Himpunan *Fuzzy* untuk Suatu Variabel

2.4.1.4 Operator Dasar Zadeh.

1. Operator *AND*

   Operator ini berhubungan dengan operasi interseksi pada himpunan. *Fire strength* sebagai hasil operasi dengan operator *AND* diperoleh dengan mengambil nilai keanggotaan terkecil antar elemen pada himpunan-himpunan yang bersangkutan.

$$«A \cap B = \min(«_A[x], «_B[y])$$

2. Operator *OR*

   Operator ini berhubungan dengan operasi *union* pada himpunan. *Fire strength* sebagai hasil operasi dengan operator OR diperoleh dengan mengambil nilai keanggotaan terbesar antara elemen pada himpunan-himpunan yang bersangkutan.

$$«A \cup B = \max(«_A[x], «_B[y])$$

3. Operator *NOT*

   Operator ini berhubungan dengan operasi komplemen pada himpunan. *Fire strength* sebagai hasil operasi dengan operator *NOT* diperoleh dengan mengurangkan nilai keanggotaan elemen pada himpunan yang bersangkutan dari 1.

$$«A^1 = 1 - «_A[x]$$

2.4.1.5 *Fuzzifikasi*

Fuzzifikasi adalah pengubahan seluruh variabel *input/output* ke bentuk himpunan *fuzzy*. Rentang nilai variabel *input* dikelompokkan menjadi beberapa himpunan *fuzzy* dan tiap himpunan mempunyai derajat keanggotaan tertentu.

Bentuk *fuzzifikasi* yang dipakai pada sistem ini adalah bentuk segitiga dan bentuk bahu. Bentuk *fuzzifikasi* menentukan derajat keanggotaan suatu nilai

rentang *input/output*. Derajat keanggotaan himpunan *fuzzy* dihitung dengan menggunakan rumus fungsi keanggotaan dari segitiga *fuzzifikasi*[10].

2.4.1.6 *Inferensi Fuzzy* (Logika Pengambilan Keputusan)

Setelah fungsi keanggotaan untuk variabel masukan dan keluarannya ditentukan, basis aturan pengendalian dapat dikembangkan untuk menghubungkan aksi keluaran pengendali terhadap kondisi masukannya. Tahap ini disebut sebagai tahap *inferensi*, yakni bagian penentuan aturan dari sistem logika *fuzzy*. Sejumlah aturan dapat dibuat untuk menentukan aksi pengendali *fuzzy*[10].

Pada basis aturan, aturan *If-Then* tersebut dapat menghubungkan banyak variabel masukan dan keluaran. Masukan *x* dipetakan menjadi keluaran *y*. Aturan *if-then* diinterpretasikan sebagai implikasi *fuzzy*. Terdapat banyak sekali model interpretasi implikasi yang telah dikembangkan. Pada aplikasi *Game* ini, metode yang digunakan adalah Metode MAMDANI.

Metode Mamdani sering juga dikenal dengan nama Metode Max-Min. Metode ini diperkenalkan oleh Ebrahim Mamdani pada tahun 1975. Untuk mendapatkan *output*, diperlukan empat tahapan :

1. Pembentukan Himpunan *Fuzzy*

   Pada Metode Mamdani, baik variabel *input* maupun variabel *output* dibagi menjadi satu atau lebih himpunan *fuzzy*.

2. Aplikasi Fungsi Implikasi

   Pada Metode Mamdani, fungsi implikasi yang digunakan adalah Min.

3. Komposisi Aturan

   Tidak seperti penalaran monoton, apabila sistem terdiri dari beberapa aturan, maka inferensi diperoleh dari kumpulan dan korelasi antar aturan. Pada Pembangunan Aplikasi *Game Tower Defense* Selamatkan Hutan Indonesia ini metode yang digunakan dalam melakukan inferensi *system fuzzy*, yaitu metode *Maximum*.

Pada metode ini, solusi himpunan *fuzzy* diperoleh dengan cara mengambil nilai maksimum aturan, kemudian menggunakannya untuk memodifikasi daerah *fuzzy*, dan mengaplikasikannya ke *output* dengan menggunakan operator OR (*union*). Secara umum dapat dituliskan :

$$\mu_{sf}[x_i] \leftarrow \max(\mu_{sf}[x_i], \mu_{kf}[x_i])$$

dengan :

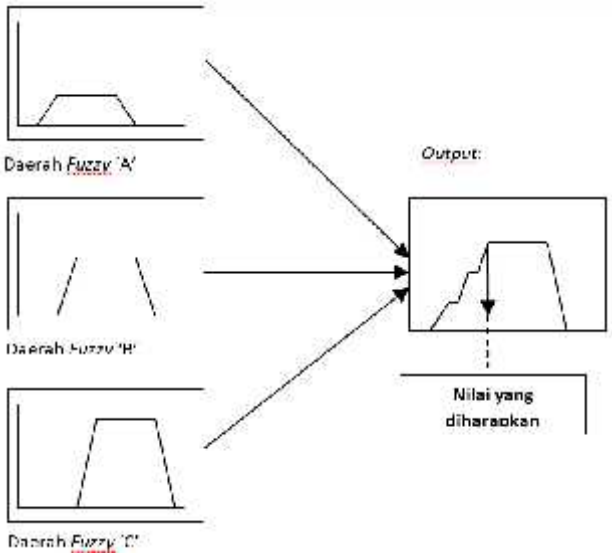
$\mu_{sf}[x_i]$ = nilai keanggotaan solusi *fuzzy* sampai aturan ke-i;

$\mu_{kf}[x_i]$ = nilai keanggotaan konsekuen *fuzzy* sampai aturan ke-i;

Apabila digunakan fungsi implikasi MIN, maka metode komposisi ini sering disebut dengan nama MAX-MIN atau MIN-MAX atau MAMDANI.

4. Penegasan (*defuzzy*)

Input dari proses *defuzzifikasi* adalah suatu himpunan *fuzzy* yang diperoleh dari komposisi aturan-aturan *fuzzy*, sedangkan *output* yang dihasilkan merupakan suatu bilangan pada *domain* himpunan *fuzzy* tersebut. Sehingga jika diberikan suatu himpunan *fuzzy* dalam *range* tertentu, maka harus dapat diambil suatu nilai crisp tertentu sebagai *output* seperti terlihat pada gambar 2.5



Gambar 2. 5 Proses *Defuzzifikasi*

2.5 Berorientasi Objek

Analisis dan desain berorientasi objek adalah cara baru dalam memikirkan suatu masalah dengan menggunkan model yang dibuat menurut konsep sekitar dunia nyata. Dasar pembuatan adalah objek, yang merupakan kombinasi antara struktur data dab perilaku dalam suatu entitas. Model berorientasi objek bermanfaat untuk memahami masalah, komunikasi dengan ahli aplikasi, pemodelan suatu organisasi, meyiapkan dokumentasi serta perancangan program dan basis data. Pertama-tama suatu model analisis dibuat untuk menggambarkan aspek dasar dari domain aplikasi, dimana model tersebut berisi objek yang terdapat dalam domain aplikasi termasuk deskripsi dari keterangan objek dan perilakunya.

Secara spesifik, pengertian berorientai objek berarti bahwa mengorganisasi perangkat lunak sebagai kumpulan dari objek tertentu yang memiliki struktur data dan perilakunya. Hal ini yang membedakan dengan pemograman konvensional dimana struktur data dan perilaku hanya berhubungan secara terpisah. Terdapat beberapa cara untuk menentukan karateristik dalam pendekatan berorientasi objek, tetapi secara umum mencakup empat hal, yaitu identifikasi, klasifikasi, *polymorphism* (polimorfisme) dan *inheritance* (pewarisan) [8].

2.5.1 Karakteristik dari Objek

Identitas berarti bahwa data diukur mempunyai nilai tertentu yang membedakan entitas dan disebut objek. Suatu paragraf dari dokumen, satu *windows* dari *workstation*, dan raja putih dari buah catur adalah contoh dari objek. Objek dapat kongkrit, seperti halnya arsip dalam sistem, atau konseptual seperti kebijakan penjadualan dalam *multiprocessing* pada sistem operasi. Setiap objek mempunyai sifat yang melekat pada identitasnya. Dua objek dapat berbeda walaupun bila semua atributnya identik.

Klasifikasi berarti bahwa suatu kegiatan mengumpulkan data (atribut) dan perilaku (operasi) yang mempunyai struktur data sama ke dalam satu grup yang disebut kelas. Paragraf, window, buah catur adalah contoh dari kelas. Kelas merupakan abstraksi yang menjelaskan sifat penting pada suatu aplikasi dan

mengabaikan yang lain. Setiap kelas menunjukan suatu kumpulan infinite yang mungkin dari objek. Suatu objek dapat dikatakan sebagai instans dari kelas, setiap instans dari kelas mempunyai nilai individu untuk setiap nama atribut dan operasi, tetapi memiliki bersama atribut dan operasi dengan instans lain dalam kelas [8].

2.5.2 Karakteristik Metodologi Berorientasi Objek

Metodologi pengembangan sistem berorientasi objek mempunyai tiga karateristik utama [8], yaitu:

1. *Class*

   kumpulan atas definisi data dan fungsi-fungsi dalam suatu unit untuk suatu tujuan tertentu. Sebagai contoh *'class of dog*' adalah suatu unit yang terdiri atas definisi-definisi data dan fungsi-fungsi yang menunjuk pada berbagai macam perilaku/turunan dari anjing. Sebuah *class* adalah dasar dari modularitas dan struktur dalam pemrograman berorientasi objek. Sebuah *class* secara tipikal sebaiknya dapat dikenali oleh seorang non-programmer sekalipun terkait dengan domain permasalahan yang ada, dan kode yang terdapat dalam sebuah *class* sebaiknya (relatif) bersifat mandiri dan independen (sebagaimana kode tersebut digunakan jika tidak berorientasi objek). Dengan modularitas, struktur dari sebuah program akan terkait dengan aspek-aspek dalam masalah yang akan diselesaikan melalui program tersebut. Cara seperti ini akan menyederhanakan pemetaan dari masalah ke sebuah program ataupun sebaliknya.

2. *Object*

   Membungkus data dan fungsi bersama menjadi suatu unit dalam sebuah program komputer. Objek merupakan dasar dari modularitas dan struktur dalam sebuah program komputer berorientasi objek.

3. *Abstract*

   Kemampuan sebuah program untuk melewati aspek informasi yang diproses olehnya, yaitu kemampuan untuk memfokus pada inti. Setiap objek dalam sistem melayani sebagai model dari "pelaku" abstrak yang dapat melakukan kerja, laporan dan perubahan keadaannya, dan berkomunikasi

dengan objek lainnya dalam sistem, tanpa mengungkapkan bagaimana kelebihan ini diterapkan. Proses, fungsi atau metode dapat juga dibuat abstrak, dan beberapa teknik digunakan untuk mengembangkan sebuah pengabstrakan.

4. *Encapsulation*

   Encapsulation (pengkapsulan) merupakan dasar untuk pembatasan ruang lingkup program terhadap data yang diproses. Data dan prosedur atau fungsi dikemas dalam bersama-sama dalam satu objek, sehingga prosedur atau fungsi lain dari luar tidak dapat mengaksesnya. Data terlindung dari prosedur atau objek lain kecuali prosedur yang berada dalam objek itu sendiri.

5. *Inheritance*

   Inheritance (pewarisan) adalah teknik yang menyatakan bahwa anak dari objek akan mewarisi atribut dan metoda dari induknya langsung. Atribut dan metode dari objek induk diturunkan kepada anak objek, demikian seterusnya. Pendefinisian objek dipergunakan untuk membangun suatu hirarki dari objek turunannya, sehingga tidak perlu membuat atribut dan metode lagi pada anaknya, karena telah mewarisi sifat induknya.

6. *Polymorphism*

   Polymorphism (polimorfisme) yaitu konsep yang menyatakan bahwa sesuatu yang sama dapat mempunyai bentuk dan perilaku berbeda. Polimorfisme mempunyai arti bahwa operasi yang sama mungkin mempunyai perbedaan dalam kelas yang berbeda.

2.6 UML (*Unified Modeling Language*)

UML (*Unified Modeling Language*) adalah bahasa spesifikasi standar untuk mendokumentasikan, menspesifikasikan, dan membangun system (Flowler,2006). *Unified Modeling Language* (UML) adalah himpunan struktur dan teknik untuk pemodelan desain program berorientasi objek (OOP) serta aplikasinya. UML adalah metodologi untuk mengembangkan sistem OOP dan sekelompok perangkat *tool* untuk mendukung pengembangan sistem tersebut. UML mulai diperkenalkan oleh Object Management Group, sebuah organisasi yang telah mengembangkan model, teknologi, dan standar OOP sejak tahun 1980-an. Sekarang UML sudah

mulai banyak digunakan oleh para praktisi OOP. UML merupakan dasar bagi perangkat (*tool*) desain berorientasi objek dari IBM.

UML adalah suatu bahasa yang digunakan untuk menentukan, memvisualisasikan, membangun, dan mendokumentasikan suatu sistem informasi. UML dikembangkan sebagai suatu alat untuk analisis dan desain berorientasi objek oleh Grady Booch, Jim Rumbaugh, dan Ivar Jacobson. Namun demikian UML dapat digunakan untuk memahami dan mendokumentasikan setiap sistem informasi. Penggunaan UML dalam industri terus meningkat. Ini merupakan standar terbuka yang menjadikannya sebagai bahasa pemodelan yang umum dalam industri peranti lunak dan pengembangan System [9].

UML menyediakan 10 macam diagram untuk memodelkan aplikasi berorientasi objek, yaitu :

1. *Use Case Diagram* untuk memodelkan proses bisnis.
2. *Conceptual Diagram* untuk memodelkan konsep yang ada di dalam aplikasi.
3. *Sequence Diagram* untuk memodelkan pengiriman pesan (*message*) antar *objects*.
4. *Collaboration Diagram* untuk memodelkan interaksi antar *objects*.
5. *State Diagram* untuk memodelkan perilaku *objects* di dalam sistem.
6. *Activity Diagram* untuk memodelkan perilaku *Use Cases* dan *objects* dalam Sistem.
7. *Class Diagram* untuk memodelkan struktur kelas.
8. *Object Diagram* untuk memodelkan struktur *object*.
9. *Component Diagram* untuk memodelkan komponen *object*.
10. *Deployment Diagram* untuk memodelkan distribusi aplikasi.

2.7 *Java*

Java adalah bahasa pemrograman yang disusun oleh James Gosling yang dibantu oleh rekan-rekannya seperti Patrick Naugton, Chris Warth, Ed Frank, dan Sheridan di suatu perusahaan perangkat lunak yang bernama Susn Microsystems, pada tahun 1991. Bahasa pemrograman ini mula-mula diinisialisasi dengan nama “Oak”, namun oada tahun 1995 diganti namanya menjadi “Java”.

Alasan utama pembentukan bahasa Java adalah untuk membuat aplikasi-aplikasi yang dapat diletakkan diberbagai macam perangkat elektronik, seperti microwave oven dan remote control, sehingga Java harus bersifat portabel atau yang sering disebut dengan *platform independent* (tidak bergantung pada platform). Itulah yang menyebabkan dalam dunia pemrograman Java, dikenal adanya istilah '*write once, run everywhere*', yang berarti kode program hanya ditulis sekali, namun dapat dijalankan di bawah platform manapun, tanpa harus melakukan perubahan kode program [9].

2.8 *Tools* Yang Digunakan

2.8.1 Greenfoot

Greenfoot merupakan sebuah IDE berbasis java yang dikhususkan untuk pembuatan *game* sederhana. Greenfoot diprakarsai oleh Michael Kolliung pada tahun 2003. *Prototype* yang pertama dibangun oleh Poul Henriksson dan Michael Kolling pada tahun 2003/2004. semenjak tahun 2005 pembuatan greenfoot ini dilanjutkan dengan melibatkan anggota kelompok dari University of Kent dan Deakin University. Pada awalnya, greenfoot bertujuan untuk menarik minat anak-anak untuk belajar pemrograman, greenfoot ini cocok untuk anak usia 13 tahun ke atas. Cara penggunaannya pun cukup mudah. Selain berbentuk teks seperti editor lainnya, pada greenfoot ini terdapat perangkat alat bantu seperti *class browser*, *editor* dan *compiler*. Greenfoot juga mendukung bahasa Java secara utuh. Dengan adanya bantuan perangkat-perangkat tadi, konsep OOP pada greenfoot lebih mudah dimengerti. Didalam greenfoot terdapat *project* yang dinamakan *scenario*. Di dalam *project* tersebut, kita bisa membuat dua macam tipe *class* dari *library* greenfoot. Yaitu *class world* dan *actor*. World merupakan latar dari *game* yang akan dibuat. Untuk membuat *class world*, caranya adalah dengan mengklik *new sub class* di bagian world. Maka akan mucul sebuah kotak dialog. Kemudian kita diminta untuk memberi nama untuk *class world* tersebut. Setelah itu kita dapat memilih gambar sebagai latar belakang *class world* yang akan kita buat. Di dalam greenfoot sendiri telah tersedia beberapa gambar untuk latar belakang, seperti

gambar tanah, pasir, batu, air, dll. *Class actor* adalah sebuah kelas yang berfungsi sebagai karakter pada *game* yang akan kita buat. Untuk membuat *actor* caranya adalah dengan meng-klik icon aktor, kemudian pilih *new subclass*. caranya hampir mirip dengan pembuatan *class world*. Pada *class actor* juga telah disediakan beberapa gambar, sepertii gambar *animal*, *people*, *symbol*, All[13].

2.8.2 Adobe Photoshop

Adobe Photoshop adalah *software* yang dibuat oleh perusahaan Adobe System Incorporated, yang di khususkan untuk pengeditan foto atau gambar dan pembuatan *effect*. Perangkat lunak ini banyak digunakan oleh Photografer Digital dan perusahaan iklan sehingga dianggap sebagai pemimpin pasar (*market leader*) untuk perangkat lunak pengolah gambar. Meskipun pada awalnya Photoshop dirancang untuk menyunting gambar untuk cetakan berbasis-kertas, Photoshop yang ada saat ini juga dapat digunakan untuk memproduksi gambar untuk *World Wide Web*. Beberapa versi terakhir juga menyertakan aplikasi tambahan, Adobe Image Ready, untuk keperluan tersebut.

Adobe Photoshop merupakan salah satu program aplikasi yang ditujukan untuk menyunting dan memanipulasi *image* (*image-editing*). Dengan Adobe Photoshop Anda dapat dengan mudah membuat dan menyunting *image* dengan kualitas yang tinggi yang siap untuk dicetak, ditempatkan di situs Web dan untuk keperluan lainnya.

Adobe Photoshop merupakan program aplikasi pengolah *image* atau gambar Bitmap. Image atau gambar Bitmap yang sering disebut Raster, merupakan gambar yang dibentuk dari grid-grid warna. Grid ini adalah elemen dasar dari sebuah *image* atau gambar yang disebut pixel atau *picture elements*. Pada saat Anda menyunting *image* atau gambar, proses yang terjadi sebenarnya adalah Anda hanya menyunting lokasi pixel-pixel ini. Gambar bitmap ini biasanya digunakan pada fotografi atau *digital painting* sehingga mode ini sangat bergantung pada tingkat kepadatan (resolusi) grid pixel-nya. Semakin tinggi resolusi sebuah *image* atau gambar, maka pixel yang dikandungnya akan semakin banyak dan semakin rapat sehingga *image* atau gambar akan mempunyai detail

yang lebih baik atau nyata. Namun perlu diperhatikan bahwa semakin besar resolusi sebuah image atau gambar akan berakibat ukuran filenya semakin besar.

Adobe Photoshop mempunyai tiga mode warna yang digunakan, yaitu RGB, CMYK, dan *index color*. Struktur *image* atau gambar yang dihasilkan monitor dengan image atau gambar cetak mempunyai perbedaan. Layar komputer atau monitor mempunyai elemen pembentukan warna Red, Green dan Blue (RGB), sedangkan warna yang dihasilkan oleh media cetak mempunyai empat kali proses pewarnaan yaitu Cyan, Magenta, Yellow, dan Black (CMYK) Photoshop juga memiliki hubungan erat dengan beberapa perangkat lunak penyunting media, animasi, dan authoring buatan-Adobe lainnya. File format asli Photoshop, .PSD, dapat diekspor ke dan dari Adobe ImageReady. Adobe Illustrator, Adobe Premiere Pro, After Effects dan Adobe Encore DVD untuk membuat DVD profesional, menyediakan penyuntingan gambar non-linear dan layanan *special effect* seperti *background*, tekstur, dan lain-lain untuk keperluan televisi, film, dan situs web. Sebagai contoh, Photoshop CS dapat digunakan untuk membuat menu dan tombol (*button*) DVD [14].

2.8.2.1 *Format File*

Photoshop memiliki kemampuan untuk membaca dan menulis gambar berformat raster seperti .png, .gif, .jpeg, dan lain-lain. Photoshop juga memiliki beberapa format file khas:

1. .PSD (*Photoshop Document*) format yang menyimpan gambar dalam bentuk layer, termasuk teks, mask, opacity, blend mode, channel warna, channel alpha, clipping paths, dan setting duotone. Kepopuleran photoshop membuat format file ini digunakan secara luas, sehingga memaksa programer program penyunting gambar lainnya menambahkan kemampuan untuk membaca format PSD dalam perangkat lunak mereka.
2. .PSB' adalah versi terbaru dari PSD yang didesain untuk file yang berukuran lebih dari 2 GB
3. .PDD adalah versi lain dari PSD yang hanya dapat mendukung fitur perangkat lunak Photshop Deluxe.

2.8.2.2 *Tool*

Tool dalam Adobe Photoshop adalah alat yang dapat membantu pengguna dalam mengedit. Adobe Photoshop memilikit 59 *tool* yang dapat dipakai oleh pengguna. *Tool - tool* tersebut terdiri dari berbagai macam *tool* dengan kegunaan yang spesifik.

Beberapa *Tools* yang ada di Photoshop diantaranya *Move Tool, History Brush Tool, Eraser Tool, Path Selection Tool, Direct Selection tool, Pen Tool, Shape Tool, Brush Tool, Audio Annotation Tool, Eyedropper Tool, Measure Tool, Text Tool, Hand Tool, 3D Object Rotate Tool, 3D Rotate,* dan *Camera Tool.*

2.8.3 Corel Draw

Corel Draw adalah editor grafik vektor yang dibuat oleh Corel, sebuah perusahaan perangkat lunak yang bermarkas di Ottawa, Kanada. Versi terakhirnya versi 16 yang dinamai X6 dirilis pada tanggal Maret 2012. Corel Draw pada awalnya dikembangkan untuk dijalankan pada sistem operasi Windows 2000 dan yang lebih baru. Versi Corel Draw untuk Linux dan Mac OS pernah dikembangkan, tetapi dihentikan karena tingkat penjualannya rendah.

Versi CorelDRAW X6 memiliki tampilan baru serta beberapa aplikasi baru yang tidak ada pada CorelDRAW versi sebelumnya. Beberapa aplikasi terbaru yang ada, di antaranya *Quick Start, Table, Smart Drawing Tool, Save as Template,* dan lain sebagainya [15].

2.8.3.1 Fasilitas Dasar Corel Draw

1. *Quick Start*

   Pertama kali perangkat lunak CorelDRAW diaktifkan, sistem akan menampilkan kotak dialog *welcome*. Pada kotak dialog ini, lembar proses yang yang dapat diaktifkan, yaitu:

   - *quick start*: aplikasi ini dapat membantu untuk mempermudah mempersiapkan ruang perancangan grafis baru atau pengubahan rancangan grafis yang pernah dibuat sebelumnya.

- *what's new*: aplikasi yang menginformasikan fasilitas baru yang disediakan pada perangkat lunak CorelDRAW X4.
- *learning tools*: alat yang mempermudah pemakai perangkat lunak CorelDRAW mempelajari fasilitas yang disediakan.
- *galeri*: galeri digunakan untuk mengunjungi situs komunitas pemakai CorelDRAW. Pada situs tersebut terdapat berbagai rancangan grafis profesional yang menggunakan perangkat lunak CorelDRAW.
- *updates*: aplikasi yang digunakan untuk mengunjungi situs resmi perangkat lunak CorelDRAW untuk berbagai keperluan. Salah satunya, yaitu untuk memperbarui model sistem CorelDRAW.

2. *Hints*

   Hints berada pada sisi kanan kotak dialog utama sistem. Fasilitas ini memberikan petunjuk apa dan bagaimana memproses suatu objek gambar/teks.

3. *Menu Bar*

   Menu Bar berada pada bagian atas kotak dialog utama sistem. Sistem menu CorelDRAW X menggunakan standar sistem operasi Windows yang sangat memudahkan pemakaian.

4. *Standard Toolbar*

   Standard toolbar terletak di bawah menu bar. Pada *standard toolbar*, sistem meletakkan simbol proses cepat. Misalnya terdapat simbol seperti folder untuk membuka suatu data grafis CorelDRAW.

5. *Property Bar*

   Property bar adalah fasilitas tambahan yang muncul setelah memilih salah satu alat dalam fasilitas *toolbox*. Tujuannya untuk mempermudah pemakaian alat fasilitas *toolbox* terpilih. *Property bar* biasanya berada di bawah *standard toolbar.*

6. *Toolbox*

   Toolbox biasanya terletak di bagian paling kiri. Sistem meletakkan sebagian memberi tanda lipatan pada sisi kanan bawah alat fasilitas *toolbox* yang memiliki subalat. Untuk menampilkan daftar sub alat tersebut dapat dilakukan dengan mengklik dan menahan simbol alat yang bersangkutan.

7. *Status Bar*

   Sistem meletakkan berbagai informasi pada baris status yang terdapat pada sisi bawah kotak dialog utama. Informasi tersebut berkenaan dengan objek gambar/teks dan/atau alat proses terpilih.

8. *Color Palette*

   Color palette atau kotak warna terletak di bagian paling kanan kotak dialog utama. Untuk memberi warna pada objek gambar/teks cukup melakukan klik seperti biasa. Sedangkan untuk memberi warna garis, klik kanan pada warna yang dipilih dalam kotak warna.

9. *Dialog Box*

   Sistem CorelDRAW akan meletakkan sejumlah pilihan proses dari fasilitas yang kompleks pada suatu kotak dialog. Fasilitas yang bersangkutan dapat diatur melalui kotak dialog tersebut. Misalnya, pengubahan format objek gambar dari vektor ke bitmap melalui menu bitmaps submenu *convert to bitmap.*

10. *Docker*

    Sistem meletakkan sejumlah kotak dialog fasilitas yang (dianggap) sering digunakan dalam bentuk tetap pada sisi kanan kotak dialog. Format tersebut disebut dengan *docker*. Fasilitas ini dapat ditampilkan melalui menu *window submenu docker*.

11. *Fixed/Floating Toolbar*

    Salah satu keistimewaan dari sistem CorelDRAW X4, yaitu fasilitas *toolbox* kotak warna, menu bar, standard *toolbar*, dan/atau property bar yang dapat digeser dan ditempatkan di sembarang lokasi.

2.8.3.2 Operasi Dasar

1. Objek Garis

   Objek garis dalam Corel Draw terdiri dari *freehand tool, bezier too, artistic media tool, pen tool, polyline tool, 3 point curve tool, connector tool* dan *dimension tool.*

2. Objek Gambar

Objek gambar Corel Draw terdiri dari *rectangle tool, ellipse tool, object tool, perfect shape tool dan table tool*.

3. *Smart Drawing*

   Menggambar pintar adalah aplikasi baru yang ada pada CorelDRAW X4. *Smart drawing tool* yang ada pada *smart tool* dapat digunakan untuk membentuk objek gambar dari sketsanya.

4. Objek Teks

   Objek teks merupakan unsur yang paling sering digunakan pada rancangan grafis. Ada yang membuatnya sebagai logo, teks artistik, bahkan mengubah objek teks menjadi objek gambar.

   Objek teks dapat dibuat melalui *text tool* yang terdapat pada *toolbox*. Setelah itu, objek teks dapat diproses melalui *pick tools*. Objek teks dapat diproses sehingga membentuk bayangan, menjadi format 3D, dan lain-lain.

5. Objek 2D Menjadi 3D

   CorelDRAW memiliki fasilitas untuk mengubah objek taks/gambar 2D menjadi 3D. Fasilitas yang dapat digunakan untuk membentuk objek 2D menjadi 3D yaitu *extrude, bevel, contour* dan efek bayangan.

6. Warna

   Pilihan warna terdapat dalam kotak warna. Namun, ada beberapa cara untuk memberi warna pada objek gambar/teks, yaitu dengan *fill & outline, eyedropper & paintbucket tool* dan *smart fill*

7. Objek Artistik

   Objek artistik dapat berupa teks atau gambar. Dengan menggunakan alat tertentu, suatu objek teks/gambar dapat terlihat berbeda dari biasa. Objek artistik terdiri dari garis artistik (Objek Teks), garis artistik (objek gambar), efek distorsi dan *smudge brush* (objek baru).

2.9 Pengujian Perangkat Lunak

Pengujian perangkat lunak merupakan elemen kritis dari jaminan kualitas perangkat lunak dan merepresentasikan kajian pokok dari spesifikasi, desain, dan pengkodean. Pentingnya pengujian perangkat lunak dan implikasinya yang

mengacu pada kualitas perangkat lunak tidak dapat terlalu ditekan karena melibatkan sederetan aktivitas produksi di mana peluang terjadinya kesalahan manusia sangat besar dan arena ketidakmampuan manusia untuk melakukan dan berkomunikasi dengan sempurna maka pengembangan perangkat lunak diiringi dengan aktivitas jaminan kualitas.

Meningkatnya *visibilitas* (kemampuan) perangkat lunak sebagai suatu elemen sistem dan "biaya" yang muncul akibat kegagalan perangkat lunak, memotivasi dilakukannya perencanaan yang baik melalui pengujian yang teliti. Pada dasarnya, pengujian merupakan satu langkah dalam proses rekayasa perangkat lunak yang dapat dianggap sebagai hal yang merusak daripada membangun.

2.9.1 Pengujian *White-Box*

Penguhian *White-box* adalah pengujian yang meramalkan cara kerja perangkat lunak secara rinci, karenanya *logikal path* (jalur logika) perangkat lunak akan di-*test* dengan menyediakan *test case* yang akan mengerjakan kumpulan kondisi atau pengulangan secara spesifik. Secara sekilas dapat diambil kesimpulan *white box testing* merupakan petunjuk untuk mendapatkan program yang benar secara 100%.

Pengujian *White-box* merupakan salah satu metode perancangan *test case* yang menggunakan struktur kontrol dari perancangan prosedural untuk mendapatkan *test case*. Dengan rnenggunakan metode *white box*, analis sistem akan dapat memperoleh *test case* yang:

1. Menjamin seluruh *independent path* di dalam modul yang dikerjakan sekurang-kurangnya sekali.
2. Mengerjakan seluruh keputusan logika.
3. Mengerjakan seluruh *loop* yang sesuai dengan batasannya.
4. Mengerjakan seluruh struktur data *internal* yang menjamin *validitas*.

2.9.1.1 Uji Coba *Basis Path*

Uji coba basis *path* adalah teknik uji coba *white box* yang diusulkan Tom Mc Cabe. Metode ini memungkinkan perancang *test case* mendapatkan ukuran kekompleksan *logical* dari perancangan prosedural dan menggunkan ukuran ini sebagai petunjuk untuk mendefinisikan *basis set* dari jalur pengerjaan. *Test case* yang didapat digunakan untuk mengerjakan *basis set* yang menjamin pengerjaan setiap perintah minimal satu kali selama uji coba.

Terdapat beberapa proses yang harus di lakukan dalam uji coba *basis path* yaitu diantaranya :

1. Notasi Diagram Alir

   Sebelum metode *basis path* diperkenalkan, terlebih dahulu akan dijelaskan mengenai notasi sederhana dalam bentuk diagram alir (grafik alir). Diagram alir menggambarkan aliran kontrol logika yang menggunakan notasi.

2. Kompleksitas Siklomatis

   Kompleksitas siklomatis adalah metriks perangkat lunak yang memberikan pengukuran kuantitatif terhadap kompleksitas logis suatu program. Bila metriks ini digunakan dalam konteks metode pengujian *basis path*, maka nilai yang terhitung untuk kompleksitas siklomatis menentukan jumlah jalur independen dalam basis set suatu pemrograman memberi batas atas bagi jumlah pengujian yang harus dilakukan untuk memastikan bahwa semua statemen telah dieksekusi sedikitnya satu kali.

   Jalur independen adalah jalur yang memlalui progarm yang mengintroduksi sedikitnya satu rangkaian statemen proses baru atau suatu kondisi baru. Bila dinyatakan dengan terminologi grafik alir, jalur independen harus bergerak sepanjang paling tidak satu *edge* yang tidak dilewatkan sebelum jalur tersebut ditentukan.

3. Melakukan *Test Case*

   Metode uji coba *basis path* juga dapat diterapkan pada perancangan prosedural rinci atau program sumber. Pada bagian ini akan dijelaskan langkah-langkah uji coba *basis path*.

4. Matriks Grafis

Prosedur untuk mendaptakan grafik alir dan menentukan serangkaian *basis path*, cocok dengan mekanisasi. Untuk mengembangkan peranti perangkat lunak yang membantu pengujian *basis path*, struktur data yang disebut matriks grafis dapat sangat berguna.

Matriks garfis adalah matriks bujur sangkar yang ukurannya sama dengan jumlah simpul pada grafik alir. Masing-masing baris dan kolom sesuai dengan simpul yang diidentifikasikan dan *entry* matriks sesuai dengan edge diantara simpul.

2.9.2 Pengujian *Black Box*

Pengujian *black-box* berfokus pada persyaratan fungsional perangkat lunak. Dengan demikian, pengujian *black-box* memungkinkan perekayasa perangkat lunak mendapatkan serangkaian kondisi *input* yang sepenuhnya menggunakan semua peryaratan fungsional untuk suatu program. Pengujian *black-box* bukan merupakan alternatif dari teknik *white-box* tetapi merupakan pendekatan komplementer yang kemungkinan besar mampu mengungkap kesalahan daripada metode *white-box.*

Pengujian *black-box* berusaha menemukan kesalahan dalam kategori sebagai berikut :

1. Fungsi-fungsi yang tidak benar atau hilang.
2. Kesalahan *interface.*
3. Kesalahan dalam struktur data atau akses database eksternal.
4. Kesalhan kinerja.
5. Inisialisasi dan kesalahan terminal.

Tidak seperti pengujian *white box* yang dilakukan pada awal proses pengujian, pengujian *black box* cenderung diaplikasikan selama tahap akhir pengujian. Karena pengujian *black Box* memperhatikan struktur kontrol, maka perhatian berfokus pada *domain* informasi. Pengujian didesain untuk menjawab pertanyaan-pertanyaan berikut :

a. Bagiamana validitas fungsional diuji ?
b. Kelas *input* apa yang akan membuat *test case* menjadi baik ?

c. Apakah sistem sangat sensitif terhadap harga *input* tertentu ?
d. Bagaimana batasan dari suatu data diisolasi ?
e. Kecepatan data apa dan *volume* data apa yang dapat ditolerir oleh sistem ?
f. Apa pengaruh kombinasi tertentu dari data terdahap operasi sistem ?

Dengan mengaplikasikan teknik *black box*, maka dapat menarik serangkaian *test case* yang memenuhi kriteria berikut ini :

1. *Test case* yang mengurangi, dengan harga lebih dari saatu, jumlah *test case* tambahan yang harus didesain untuk mencapai pengujian yang dapat dipertanggungjawabkan.
2. *Test case* yang memberi tahu sesuatu mengenai kehadiran atau ketidakhadiran kelas kesalahan daripada memberi tahu kesalahan yang berhubungan hanya dengan pengujian. spesifik yang ada.

2.9.3 Pengujian *Alpha* dan Pengujian *Beta*

Apabila Perangkat Lunak dibuat untuk pelanggan maka dapat dilakukan *aceeptance test* sehingga memungkinkan pelanggan untuk memvalidasi seluruh keperluan. *Test* ini dilakukan karena memungkinkan pelanggan menemukan kesalahan yang lebih rinci dan membiasakan pelanggan memahami Perangkat Lunak yang telah dibuat.

2.9.3.1 Pengujian *Alpha*

Dilakukan pada sisi pengembang oleh seorang pelanggan. Pengujian ini dilakukan terhadap perangkat lunak untuk memastikan bahwa perangkat lunak dapat berjalan dengan benar sesuai dengan kebutuhan dan tujuan yang diharapkan. Pengujian *Alpha* berfokus pada persyaratan fungsional dari perangkat lunak.

2.9.3.2 Pengujian *Beta*

Pengujian *beta* merupakan pengujian yang dilakukan secara objektif, Dimana pengujian dulakukan secara langsung terhadap pengguna, biasanya menggunakan kuisioner mengenai tanggapan pengguna atas perangkat lunak yang telah dibangun. Metode penilaian pengujuan yang digunakan adalah metode *kuantitatif* berdasarkan data dari pengguna.

2.9.4 *Skala Likert*

Skala Likert adalah suatu skala psikometrik yang umum digunakan dalam kuesioner. *Skala Likert* merupakan skala yang paling banyak digunakan dalam riset berupa survey seperti mengukur sikap, pendapat, dan persepsi seseorang atau sekelompok orang tentang fenomena sosial. Dengan *Skala Likert*, variabel yang akan diukur dijabarkan menjadi indikator variabel. Kemudian indikator tersebut dijadikan sebagai titik tolak untuk menyusun item-item instrumen yang dapat berupa pertanyaan atau pernyataan. Nama skala ini diambil dari nama Rensis Likert, yang menerbitkan suatu laporan yang menjelaskan penggunaannya. Sewaktu menanggapi pertanyaan dalam *skala Likert*, responden menentukan tingkat persetujuan mereka terhadap suatu pernyataan dengan memilih salah satu dari pilihan yang tersedia. Biasanya disediakan lima pilihan skala dengan format seperti:

1. Sangat tidak setuju
2. Tidak setuju
3. Netral
4. Setuju
5. Sangat setuju

Penulis kuesioner harus memutuskan apakah memasukkan titik tengah atau tidak sesuai dengan pernyataan yang diberikan kepada responden. Meskipun penggunaan respon kategori tengah tidak mempengaruhi *reliabilitas* dan *validitas* dalam penelitian ini, namun direkomendasikan bahwa penilaian. Pengembang kuesioner untuk memasukkan alternatif tengah. Ahli lain bahwa menyediakan kategori tengah memungkinkan responden untuk menunjukkan respon yang netral dan lebih *diskriminatif* dalam respon mereka, membuat nilai skala yang lebih handal dan skala yang lebih disukai oleh responden.

Selain pilihan dengan lima skala seperti contoh di atas, kadang digunakan juga skala dengan tujuh atau sembilan tingkat. Suatu studi empiris menemukan bahwa beberapa karakteristik statistik hasil kuisioner dengan berbagai jumlah pilihan tersebut ternyata sangat mirip.

Prosedur dalam membuat *skala Likert* adalah sebagai berikut:

1. Peneliti mengumpulkan item-item yang cukup banyak, relevant dengan masalah yang sedang diteliti, dan terdiri dari item yang cukup jelas disukai dan tidak disukai.
2. Kemudian item-item itu dicoba kepada sekelompok responden yang cukup representatif dari populasi yang ingin diteliti.
3. Responden di atas diminta untuk mengecek tiap item, apakah ia menyenangi (+) atau tidak menyukainya (-). Respon tersebut dikumpulkan dan jawaban yang memberikan indikasi menyenangi diberi *skor* tertinggi. Tidak ada masalah untuk memberikan angka 5 untuk yang tertinggi dan *skor* 1 untuk yang terendah atau sebaliknya. Yang penting adalah konsistensi dari arah sikap yang diperlihatkan. Demikian juga apakah jawaban "setuju" atau "tidak setuju" disebut yang disenangi, tergantung dari isi pertanyaan dan isi dari item-item yang disusun.
4. Total *skor* dari masing-masing individu adalah penjumlahan dari *skor* masing-masing item dari individu tersebut.
5. Respon dianalisis untuk mengetahui item-item mana yang sangat nyata batasan antara skor tinggi dan skor rendah dalam skala total. Misalnya, responden pada *upper* 25% dan *lower* 25% dianalisis untuk melihat sampai berapa jauh tiap item dalam kelompok ini berbeda. Item-item yang tidak menunjukkan beda yang nyata, apakah masuk dalam skor tinggi atau rendah juga dibuang untuk mempertahankan konsistensi internal dari pertanyaan.