BAB 2

LANDASAN TEORI

2.1 Pertolongan Pertama

Pertolongan pertama merupakan pemberian pertolongan dengan segera kepada korban yang mengalami sakit, cedera atau kecelakaan secara mendadak sebelum korban dibawa ketempat rujukan (fasilitas kesehatan terdekat seperti puskesmas, rumah sakit, dll). Tujuan dari pertolongan pertama yaitu : menyelamatkan jiwa penderita / korban, mencegah cacat, serta memberikan rasa nyaman terhadap korban. Teknik tata cara melakukan pertolongan pertama dapat dipelajari disekolah, universitas, instansi perusahaan dan juga bisa didapat dari kader – kader PMI, BASARNAS dll. Bentuk dari ilmu dan teknik tata cara yang harus diketahui oleh para pelaku pertolongan pertama yang pelakunya adalah anak – anak berupa :

1. Ilmu pengetahuan mengenai dasar – dasar pertolongan pertama.

   Ilmu ini berisi tentang pengertian dari pertolongan pertama, tujuan dari tindakan pertolongan, serta kewajiban para pelaku pertolongan pertama.

2. Alat perlindungan diri (APD) dan peralatan pertolongan pertama

   Para pelaku pertolongan pertama harus mengutamakan keselamatanya sendiri terlebih dahulu sebelum membantu korban. Jadi, para pelaku pertolongan pertama harus mengetahui alat perlindungan dirinya (APD) agar tidak tertular oleh kuman atau penyakit yang mungkin bisa didapat dari tubuh korban. Hal penting yang harus diingat adalah darah dan semua cairan tubuh bisa mengeluarkan kuman ataupun penyakit. APD minimal dapat berupa :

   a. Sarung tangan lateks berfungsi sebagai alat perlindungan diri dari kuman yang ada pada tubuh korban yang menderita.
   b. Masker penolong, berfungsi sebagai alat mencegah penularan penyakit melalui udara.

c. Kaca mata pelindung berfungsi sebagai alat untuk melindungi mata dari percikan darah, mencegah cedera akibat benturan atau kelilipan pada mata saat melakukan pertolongan.

Selain APD dibutuhkan peralatan pendukung lainnya berupa :

a. Kasa steril, berfungsi sebagai penutup bagian yang terluka
b. Pembalut gulung / perban, berfungsi sebagai penutup dan penahan bagian yang terluka.
c. Pembalut perekat / plester, berfungsi sebagai penutup luka dan perekat pembalut agar gulugan balutan luka tidak mudah terbuka atau terurai.
d. Gunting pembalut, berfungsi sebagai : alat pemotong pembalut.
e. Bidai, berfungsi sebagai alat stabilisasi tubuh.
f. Pinset, berfungsi sebagai alat pembantu dalam mengambil benda – benda kecil yang tertancap pada tubuh korban, dan dapat digunakan untuk membantu mengambil alat pendukung yang diperlukan pada saat melakukan pertolongan.
g. Senter, berfungsi sebagai pembantu pencahayaanbagi penolong.
h. Kapas, berfungsi sebagai penyerap cairan luka, dan penyerap cairan antiseptic yang dapat diberikan pada korban disaat melakukan proses pertolongan
i. Cairan antiseptic, berfungsi sebagai cairan pembunuh kuman yang menempel pada luka agar luka cepat mengring dan tidak bernanah atau mengalami infeksi.
j. Dll yang biasanya sudah tersedia di dalam kotak atau tas P3K.

3. Ilmu pengetahuan anatomi faal tubuh manusia

Anatomi adalah uraian fungsi organ tubuh atau susunan tubuh manusia. Bagian tubuh manusia dilindungi oleh kulit dan diperkuat oleh rangka.

Umumnya tubuh manusia terbagi menjadi 5 bagian yaitu :

a. Kepala

Kepala terdiri dari : tengkorak, wajah, dan rahang bawah.

b. Leher
c. Batang tubuh

Terdiri dari : dada, perut, punggung dan panggul.

d. Anggota gerak atas

Sendi bahu, lengan atas, siku, lengan bawah, pergelangan tangan, dan tangan.

e. Anggota gerak bawah

Sendi panggul, tungkai atas (paha), lutut, tungkai bawah, pergelangan kaki, dan kaki.

Dalam tubuh manusia juga terdapat batang rongga yang terbagi menjadi 5 buah rongga, yaitu :

a. Rongga tengkorak
b. Rongga tulang belakang
c. Rongga dada
d. Rongga perut
e. Rongga panggul.

Dalam tubuh manusia juga terdapat banyak sistem, yang terpenting diantaranya adalah : sistem pernapasan, sistem peredarah darah, sistem saraf, sistem otot, sistem rangka, sistem pencernaan, sistem kemih, sistem kelenjar buntu, kulit, panca indra, dan sistem reproduksi.

4. Penilaian terhadap korban

Pada saat sampai dilokasi kejadian, hal yang pertama kali dilakukan adalah mengenali lingkungan atau menilai keadaan sekitar. Apakah aman atau tidak bagi dirinya. Jika ragu lebih baik minta bantuan kepada orang dewasa atau orang sekitar yag lebih ahli. Penilaian dini juga penting dilakukan untuk menentukan korban karena penyakit atau cedera. Jika diketahui berasal dari penyakit maka dapat meminta bantuan dari orang lain. Setelah melakukan penilaian langkah selanjutnya adalah :

a. Memeriksa respon. Untuk memeriksa respon dilakukan dengan empat tingkatan yaitu: awas, suara, nyeri, tidak respon atau disingkat dengan ASNT.
b. Memastikan jalan napas terbuka dengan baik, jika korban tidak merespon, tindakan selanjutnya melakukan teknik angkat dagu dan tekan dahi.

c. Untuk menilai pernapasan dilakukan dengan cara : lihat, dengar, rasakan
d. Menilai denyut nadi. Untuk menilai denyut nadi bagi korban sadar dapat dilakukan dengan meraba nadi pergelangan tangan (radial). Sedangkan bagi korban yang tidak sadar, nadi yang diperiksa adalah di bagian leher (carotis).
e. Langkah terakhir adalah menghubungi bantuan dari orang ahli, dapat meminta bantuan melalui telepon, dll.

Setelah melakukan penilaian dini maka tahapan selanjutnya yaitu melakukan pemeriksaan fisik. Tanda yang harus diketahui saat pemeriksaan fisik adalah :

a. Adakah perubahan bentuk pada bagian tubuh korban.
b. Adakah luka terbuka (yang terlihat) pada tubuh korban.
c. Apakah korban merasakan nyeri saat bagian tubuh nya di raba atau ditekan.
d. Apakah ada bengkak yang terjadi pada korban.

Untuk pemeriksaan lebih lanjut kondisi korban, perlu dilakukan pemeriksaan lebih lengkap dari ujung kepala sampai ujung kaki. Rincian nya sebagai berikut :

a. Kepala : telinga, hidung, mata, mulut.
b. Leher
c. Dada
d. Perut
e. Punggung
f. Panggul
g. Anggota gerak atas dan bawah

5. Pengetahuan tentang luka

Luka adalah rusaknya jaringan lunak baik didalam maupun luar bagian tubuh. Luka yang paling jelas terlihat pada kulit. Luka dapat dikelompokan menjadi dua: yaitu luka terbuka, dan luka tertutup.

Luka terbuka yaitu kerusakan jaringan lunak yang menyebabkan darah keluar. Contoh nya : luka lecet, luka sayat, luka robek, luka tusuk, luka sobek, amputasi. Cara penanganan luka terbuka yaitu dengan membersihkan

luka tersbut menggunakan air atau dengan cairan antiseptic, dan menutup luka tersebut dengan penutup luka. Penutup luka mempunyai peranan penting karena mempunyai fungsi sebagai pencegah masuknya kuman kedalam luka dan dapat membantu menghentikan perdarahan pada luka.

Sedangkan Luka tertutup yaitu bila darah tidak terlihat, contohnya : memar, benjol,dan remuk. Penangganan yang dapat dilakukan pada saat terjadinya luka tertutup yaitu dengan cara member kompres dingin pada memar, menekan bagian terluka dengan kompres dingin, dan apabila luka tertutup ini terjadi pada bagian alat gerak tubuh maka penangganan dapat dilakukan denga cara menaikan atau meninggikan bagian terluka lebih tinggi dari pada jantung [1].

2.2 *Game*

Merupakan sebuah aktifitas yang bertujuan untuk bersenang – senang ataupun aktifitas sekedar pengisi waktu luang. *Game* mempunyai aturan cara bermain tertentu sehingga nantinya akan ada hasil yang kalah atau menang. *Game* dapat dilakukan secara perseorangan maupun kelompok.

2.2.1 Pengertian *Game*

Pengertian *Game* menurut para ahli adalah :

1. Menurut Agustianus Nilwan dalam bukunya yaitu Pemograman Animasi dan *Game* professional, *Game* merupakan *Game* 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 [5].
2. Menurut Jhon von Neumann dan Oskar Morgenstern tahun 1994, *Game* adalah: *Game* yang terdiri atas sekumpulan peraturan yang membangun situasi bersaing dari dua sampai beberapa orang atau kelompok dengan mmilih strategi yang akan dibangun untuk memaksimalkan kemenangan sendiri atau pun untuk meminimalkan kemenangan lawan. Peraturan – peraturan menentukan kemungkinan tindakan untuk setiap pemain,

sejumlah keterangan diterima setiap pemain sebagai kemajuan bermain, dan sejumlah kemenangan atau kekalahan dalam berbagai situasi.

3. Menurut David Parlett, *Game* adalah sesuatu yang memiliki akhir dan cara mencapainya, artinya ada awal, tujuan, hasil dan serangkaian peraturan untuk mencapai keduanya.
4. Menurut Clark C. Abt, *Game* adalah kegiatan yang melibatkan keputusan pemain, untuk berupaya mencapai tujuan dengan dibatasi oleh konteks tertentu. Misalnya peraturan – peraturan *Game*.

Berdasarkan pendapat para ahli tersebut, maka *Game* dapat diartikan sebagai permainan yang dibuat dengan komputer menggunakan teknik dan metode animasi, dimana *Game* harus mempunyai awal dan akhiran yang harus dicapai (target) yang dibatasai oleh aturan – aturan dalam permainan. Hasil akhir dari permainan dapat berupa kemenangan atau kekalahan, serta *Game* juga dapat dimainkan secara perorangan maupun kelompok.

2.2.2 *Game* Komputer

Game komputer atau yang biasa disebut *Game* digital adalah *Game* yang dimainkan dengan menggunakan suatu komputer. *Game* komputer saat ini sudah berevolusi dari sistem grafis sederhana menjadi sangat komplek dan mutakhir. Contohnya adalah *Game online.* video *Game*, *Game console*

2.2.2.1 *Game* Online

Game online adalah jenis *Game* video atau *Game* komputer dengan menggunakan jaringan internet sebagai media interaksinya. *Game online* terdiri dari dua unsur utama yaitu : *server* dan *client*. *Server* adalah penyedia layanan *gaming* yang merupakan basis agar *client – client* yang terhubung dapat memainkan *Game* sambil melakukan komunikasi jarak jauh denga baik. Suatu *server* pada prinsipnya hanya melakukan administrasi *Game* dan menghubungkan *client – client.* Sedangkan *client* adalah pengguna *Game* yang menggunakan kemampuan *server*. Contoh *Game online* adalah Point blank, perfect word, ragnarok, warcraf dota, dll.

2.2.2.2 Video *Game*

Game yang menggunakan interaksi antarmuka pengguna melalui gambar yang dihasilkan oleh piranti video. *Video Game* juga sering diartikan sebagai *Game* yang berbasis elektronik dan visual. Sistem elektronik yang digunakan untuk menjalankan *Game* disebut *patform,* contohnya adalah komputer pribadi (*personal komputer)* ataupun konsol *Game* (*Game console).*

2.2.2.3 Konsol *Game*

Console *Game* adalah sebuah mesin elektronik yang dirancang khusus untuk memainkan *Game* video. Perangkat keluarannya bisa berupa monitor komputer atau televisi dan alat masukan utamanya berupa sebuah pengendali. Konsol *Game* yang pertama kali dibuat adalah *Atari,* kemudian dilanjutkan dengan *ninetendo yang sukses pada tahun 1985 – 1989.* Konsol *Game* modern saat ini adalah *playstation* buatan *sony.*

2.2.3 Klasifikasi *Game*

Sejarah *Game* tidak lepas dari yang namanya *platform,* dari yang awalnya hanya berupa mesin *Arcade* (mesin dingdong) hingga *mobile Games*. Berikut adalah klasifikasi *Game* berdasarkan jenis *platform* yang digunakan [8]:

1. *Arcade Games,* adalah *Game* yang sering disebut dengan *Game* dingdong di Indonesia, biasanya *Game* ini berada di 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 transmisi tentunya).
2. *Pc Games*, yaitu *video Game* yang dimainkan menggunakan personal komputer.
3. *Console Game*, yaitu video *Game* yang dimainkan menggunakan *console* tertentu misalnya : palystation 1-3, XBOX 360, Nintendo, dll.
4. *Handeld Games* adalah *Game* yang dimainkan di *console* khusus *video Game* yang juga dapat dibawa kemana – mana, contoh : Ninetendo DS dan Sony PSP.

5. *Mobile Games,* adalah *Game* yang dapat dimainkan pada *mobile phone* atau PDA.

Klasifikasi *Game* berdasarkan pembatasan usia dari pemainnya, menurut ESRB (*Entertainmet Software Rating Board)* adalah sebagai berikut :

1. *Early Childhood*

   Konten *Game* yang ditujukan untuk anak – anak kecil

2. *Everyone*

   Konten *Game* yang umumnya cocok untuk segala usia. Mungkin berisi kartun minimal, fantasi atau kekerasan ringan, dan bahasa yang ringan untuk pengguna.

3. *Everyone (10+)*

   Konten *Game* yang cocok untuk usia 10 tahun ke atas. Munkin berisi kartun lebih, fantasi, kekerasan ringan, bahasa ringan, dan tema sugestif minimal.

4. *Teen*

   Konten *Game* yang cocok untuk usia 13 tahun ke atas. Mungkin berisi kekerasan, tema sugestif, humor kasar, darah minimal, simulasi perjudian, dan tidak jarang menggunakan bahasa yang kuat.

5. *Mature*

   Konten *Game* yang cocok untuk usia 17 tahun ke atas.

   Mungkin berisi kekerasan intens, darah, konten seksual, dan bahasa yang kuat.

6. *Adults Only*

   Konten yang ditujukan untuk orang dewasa usia 18 tahun ke atas. Mungkin berisi adegan kekerasan yang berkepanjangan, konten seksual grafis, dan perjudian dengan mata uang yang rill.

7. *Rating Pending*

   ESRB belum memberikan *rating* (peringkat) akhir atau sedang menunggu penilaian akhir. Biasanya simbol ini hanya muncul dalam iklan, pemasaran, dan bahan promosi yang berkaitan dengan *Game* yang dapat dapat membawa rating ESRB, dan harus diganti dengan rating *Game* setelah ditetapkan.

2.2.4 Sudut Pandang *Game* (*Game View Point*)

Sebuah *Game* mempunyai sudut pandang *Game* yang telah disesuaikan berdasarkan genre *Game* yang diambil. Berikut beberapa macam sudut pandang *Game* yang biasa digunakan [9]:

1. *Side Scroller Game*

   Saat pertama kali muncul *Game* ini berbentuk 2D, sekarang sudah banyak yang dibuat dengan efek 3D. pada *Game* jenis ini pemain diharuskan bergerak searah di alur yang disediakan. Pemain diharuskan untuk berjalan, meloncat, merunduk, serta menghindari rintangan – rintangan. Contoh *Game* nya adalah *: Mario bros* dan *Price of Persia*.

2. *Top Down*

   Sudut pandang yang memungkinkan karakter utama bermanuver ke empat arah namun cara *Game*nya sendiri bergeser dari bawah ke atas, dan biasanya *Game* yang menggunakan susut pandang jenis ini adalah *shooter Game*.

3. *Isometric*

   Sudut pandang *Game* yang memungkinkan *Game* terlihat diatara sisi *side scrolling* dan juga *top down*, dan biasanya diterapkan pada *Game* dengan genre RTS (*Real Time Strategy).*

2.2.5 Jenis *Game* (*Genre Game*)

Game dari tahun-tahun semakin banyak saja jenisnya, teknologi yang dipakaipun semakin canggih. Karena banyak nya *Game* yang bermunculan dipasaran maka, *Game-Game* tersebut dikelompokkan menjadi beberapa *genre*. Pada dasarnya, *genre-genre Game* yang beredar di pasaran dapat dikelompokkan sebagai berikut [9]:

1. *Advertising Games(adverGame)*

   Game yang dikembangkan secara eksplisit maupun menyeluruh memasukan muatan promosi untuk tujuan periklanan. Penerapan *adverGame* ini dapat terlihat dari karakter dengan brand tertentu. Contohnya: *Game biskuat milk,*

2. *Action Game*

   Action Game adalah tipe *Game* dengan fitur utama berupa banyaknya aksi di mana pemain harus memiliki keterampilan reaksi yang cepat untuk menghindari musuh atau menghindari rintangan. Pengembang *Game* tipe ini perlu memastikan *Game* yang dibuat dioptimasi sehingga pemain memiliki pengalaman bermain yang baik, yang tidak terganggu oleh *delay* proses yang lama, contohnya: *Metal Gear Solid.*

3. *Adventure Game*

   Adventure *Game* adalah tipe *Game* yang umumnya membuat pemain harus berjalan mengelilingi suatu tempat yang terkondisi, seperti sebuah istana, gua yang berkelok, dan planet yang jauh. Pemain melakukan navigasi suatu area mencari pesan - pesan rahasia, memperoleh obyek yang memiliki kemampuan yang bervariasi, bertempur dengan musuh, dan lain-lain. Untuk membuat *Game* ini, diperlukan perencanaan yang akurat sehingga memiliki alur cerita yang menarik bagi pemain, contohnya : *Tomb Raider, Prince of Persia*, *Hitman*, dan lain-lain. Dalam penelitian *Game* yang akan dibangun akan memakai tipe *Game* bergenre *adventure* seperti ini.

4. *Fighting Game*

   Fighting Game tipe *Game* yang intinya harus menjatuhkan lawan tandingnya, entah itu dengan pukulan, tendangan, combo, maupun dengan jurus spesial, contohnya : *Tekken, Mortal Kombat,One piece, Naruto*, dan lain-lain.

5. Selain dari tipe jenis *Game* diatas juga terdapat jenis *Game* berupa FPS (*First Person Shooter), RPG (Role Playing Game), platform Game, puzzle Game, racing Game, simulation Game, sandbox Game, RTS (Real Time Strategy), dan sport Game.*

2.2.6 *Game* Edukasi

Game edukasi adalah *Game* digital yang dirancang untuk pengayaan pendidikan (mendukung pengajaran dan pembelajaran), menggunakan teknologi multimedia interaktif [6].

Menurut Hurd dan Jenuings, perancang yang baik haruslah memenuhi krtieria dari *education Game* itu sendiri. Berikut ini adalah beberapa kriteria dari sebuah *Game* edukasi yaitu [7]:

1. Nilai Keseluruhan *(Overall Value)*

   Nilai keseluruhan dari suatu *Game* terpusat pada desain dan panjang durasi *Game*. Aplikasi ini dibangun dengan desain yang menarik dan interaktif. Untuk penentuan panjang durasi, aplikasi ini menggunakan fitur timer.

2. Dapat Digunakan (*Usability*)

   Mudah digunakan dan diakses adalah poin penting bagi pembuat *Game*. Aplikasi ini merancang sistem dengan *interface* yang *user friendly* sehingga user dengan mudah dapat mengakses aplikasi.

3. Keakuratan *(Accuracy)*

   Keakuratan diartikan sebagaimana kesuksesan model atau gambaran sebuah *Game* dapat dituangkan kedalam percobaan atau perancangannya. Perancangan aplikasi ini harus sesuai dengan model *Game* pada tahap perencanaan.

4. Kesesuaian *(Appropriateness)*

   Kesesuaian dapat diartikan bagaimana isi dan desain *Game* dapat diadaptasikan terhadap keperluan *user* dengan baik, aplikasi ini menyediakan menu dan fitur yang diperlukan *user* untuk membantu pemahaman user dalam menggunakan aplikasi.

5. Relevan *(Relevance)*

   Relevan artinya : dapat mengaplikasikan isi *Game* ke target user. Agar dapat relevan terhadap *user*, sistem harus membimbing mereka dalam pencapaian tujuan pembelajaran. Karena aplikasi ini ditujukan untuk anak – anak, maka desain antarmuka harus sesuai denga nuansa anak – anak, yaitu menampilkan warna – warna yang ceria

6. Objektifitas *(Objectives)*

   Objektifitas menentukan tujuan *user* dan kriteria dari kesuksesan atau kegagalan. Dalam aplikasi ini objektivitas adalah usaha untuk mempelajari hasil dari *Game*.

7. Umpan Balik *(Feedback)*

   Untuk membantu pemahaman *user* bahwa *Game* (*performance*) mereka sesuai dengan objek *Game* atau tidak, *feedback* harus disediakan. Aplikasi ini menyajikan animasi dan efek suara yang mengindikasikan kesuksesan atau kegagalan *Game*.

2.3 Kecerdasan Buatan / AI (Artificial Inteligence)

Kecerdasan buatan atau *Artificial Inteligence* yang disingkat dengan AI adalah cabang ilmu yang berusaha memahami kecerdasan manusia, membangun entitas – entitas cerdas yang sesuai dengan pemahaman manusia. Entitas – entitas cerdas yang dibangun AI ternyata sangat menarik dan mempercepat prose pemahaman terhadap kecerdasan manusia. Hingga saat ini AI terus dipelajari dan dikembangkan sehingga banyak bidang studi baru yang berawal dari AI, seperti ; *computional intelligence, soft computing, fuzzy systems, evolutionary computation*, dll yang semakin fokus pada bidang kajian dan permasalahan tertentu [10].

AI berhubungan dengan penggunaan komputer untuk melaksanakan tugas atau menyelesaikan masalah yang memerlukan kecerdasan atau kepintaran. Pada umumnya komputer memerlukan aplikasi atau software untuk melaksanakan tugas atau menyelesaikan masalah, maka aplikasi atau software yang berdasarkan teknologi AI diperlukan. Intelegensi perlu dilihat dari berbagai sudut pandang, termasuk pemahaman, inferensi, daya penjelasan, pembelajaran, komunikasi dan sebagainya.

Dari penjelasan tersebut maka dalam pembangunan *Game* ini juga diperlukan sebuah kecerdasan buatan / AI agar aplikasi *Game* yang dibangun dapat berjalan dengan baik.

2.3.1 Definisi AI

Para ilmuan memiliki dua cara pandang yang berbeda tentang AI yang Pertama adalah memandang AI sebagai bidang ilmu yang hanya fokus pada proses berfikir. Sedangkan yang kedua adalah memandang AI sebagai bidang ilmu yang fokus pada tingkah laku. Stuart Russel dan Petter Norvig mengelompokan definisi AI menjadi empat kategori [11]:

1. *Thinking Humanly* (Berpikir seperti Manusia)

   Pendekatan ini dilakukan dengan dua cara sebagai berikut :

   a. Melalui intropeksi mencoba menyangka pemikiran – pemikiran manusia pada saat berfikir. 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, ternyata pemikiran tersebut sudah lewat dan digantikan kesadaran anda. Sehingga, definisi ini terkesan mengada - ada dan tidak mungkin dilakukan.
   b. Melalui eksperimen-eksperimen psikologi.

2. *Acting Humanly* (Beraksi seperti Manusia)

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

3. *Thinking Rationally* (Berpikir Rasional)

   Terdapat dua masalah dalam pendekatan ini, yaitu :

   a. 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%.

b. Terdapat perbedaan besar antara dapat memecahkan masalah “dalam prinsip” dan memecahkannya “dalam dunia nyata”.

4. *Acting Rationally* (Beraksi Rasional)

   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.

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.

2.3.2 Penerapan AI

Semakin pesatnya perkembangan teknologi menyebabkan adanya perkembangan dan perluasan lingkup yang membutuhkan kehadiran AI. Karakteristik cerdas sudah mulai dibutuhkan di berbagai disiplin ilmu, tidak hanya pada bagian disiplin ilmu komputer / informatika, tetapi juga sudah ke berbagai disiplin ilmu lain. Pentingnya kehadiran AI menjadikan ruag lingkup utama penerapan AI, sehingga penerapan AI dapat terbagi menjadi beberapa bagian yaitu :

1. Sistem Pakar (*Expert System*)

   Disini komputer digunakan sebagai sarana untuk menyimpan pengetahuan para pakar. Dengan demikian komputer akan memiliki keahlian untuk meyelesaikan permasalahan dengan meniru keahlian yang dimiliki oleh para pakar.

2. Pengolahan Bahasa Alami (*Natural Language Processing*)

Dengan pengolahan pengenalan bahasa alami ini diaharapkan user dapat berkomunikasi dengan komputer dengan menggunakan bahasa sehari – hari.

3. Pengenalan Ucapan (*Speech Recocnition*)

   Melalui pengenalan ucapan diaharapkan user dapat berkomunikasi dengan komputer menggunakan suara.

4. Robotika dan sistem sensor

   Robotika dan sistem sensor merupakan irisan ilmu elektro dengan AI, dimana sistem robotik dapat mengenal pola, dan bertingkah layaknya seperti apa yang di inginkan pembuatnya.

5. *Komputer vision*

   Komputer Vision mencoba untuk dapat menginterprestasikan gambar atau obyek – obyek tampak melalui komputer.

6. *Intelligent Komputer – aided Instruction*

   Komputer dapat digunakan sebagai tutor yang dapat melatih dan mengajar.

7. *Game playing*

   Komputer dapat digunakan membuat bermacam – macam *Game* dengan banyak metode dan kecerdasan buatan didalamnya.

Dengan semakin pesatnya perkembangan teknologi dan semakin bermacam – macam nya penerapan AI yang bisa digunakan, maka dalam *Game* Pertolongan Pertama ini juga akan memanfaatkan penerapan AI untuk menggerakan NPC (*Non-Playable Character*).

2.3.3 Metode Pencarian AI (*searching)*

Banyak metode AI yang dapat diusulkan untuk dijadikan sebagai cara memecahakan suatu permasalahan dalam sebuah sistem, salah satunya dengan metode teknik pencarian (*searching)* yang digunakan untuk pencarian rute optimum untuk memandu seseorang diperjalanan. Metode ini dipilih agar dapat membantu penerapan AI yang diberikan kepada para NPC yang berguna untuk mencari rute atau jalan mendekati tokoh utama yan ada pada *Game* ini.

Berikut cara pemecahan masalah dengan metode *searching* atau pencarian[10] :

1. Mendefinisikan ruang masalah untuk suatu masalah yang dihadapi, ruang masalah ini dapat digambarkan sebagai himpunan keadaan (*state*) atau bisa juga sebagai himpunan rute dari keadaan awal (*initial state*) menuju keadaan tujuan (*goal state*).
2. Mendefinisikan aturan produksi yang digunakan untuk mengubah suatu *state* ke *state* lainnya.
3. Memilih metode pencarian yang tepat sehingga dapat menemukan solusi terbaik dengan usaha yang minimal.

Pada dasarnya teknik pencarian (*searching*) dapat dibagi menjadi 2 kelompok besar, yaitu pencarian *blind* atau *un-informed search* (pencarian buta atau tidak berbekal informasi) dan *heuristic* atau *informed search* (pencarian dengan panduan atau berbekal informasi).

2.3.3.1 Pencarian Blind / Un-informed Search

Pada teknik pencarian ini tidak terdapat informasi awal yang digunakan dalam proses pencarian. 6 metode yang tergolong *blind search* yaitu [10]:

1. *Breadth-First Search* (BFS)

   Pencarian dilakukan pada semua simpul dalam setiap level secara berurutan dari kiri ke kanan. Jika pada satu level belum ditemukan solusi, maka pencarian dilakukan pada level selanjutnya. Demikian seterusnya sampai ditemukan solusi.
2. *Depth-First Search (DFS)*

   pencarian dilakukan pada suatu simpul dalam setiap level dari yang paling kiri. Jika pada level yang terdalam solusi belum ditemukan, maka pencarian pada simpul sebelah kanan dan simpul yang kiri dapat dihapus dari memori. Jika pada level yang paling dalam tidak ditemukan solusi, maka pencarian dilanjutkan pada level sebelumnya. Demikian selanjutnya sampai ditemukan solusi.
3. *Depth-Limited Search* (DLS)

   Metode ini berusaha mengatasi kelemahan DFS (tidak *complete*) dengan membatasi kedalaman maksimum dari suatu jalur solusi. DLS bisa menjadi

tidak *complete* jika batasan kedalamannya lebih kecil dibandingkan dengan level solusinya.

4. *Uniform Cost Search* (UCS)

   Kosepnya hampir sama dengan BFS, bedanya adalah bahwa BFS menggunakan urutan level dari yang paling rendah sampai yang paling tinggi, sedangkan UCS menggunakan urutan biaya dari yang paling kecil sampai yang terbesar. UCS berusaha menemukan solusi dengan total baiya terendah yang dihitung berdasarkan biaya dari simpul asal menuju simpul tujuan.

5. *Iteractive-Deepening Search* (IDS)

   IDS merupakan metode yang menggabungkan kelebihan BFS (*complete* dan *optimal*) dengan kelebihan DFS (*space complexity* rendah atau membutuhkan sedikit memori). Tapi konsekuensinya adalah *time complexity*-nya menjadi tinggi.

6. *Bi-Directional Search* (BDS)

   Pada setiap iterasi pencarian dilakukan dari dua arah yaitu pencarian maju (dari *start* ke *goal*) dan pencarian mundur (dari *goal* ke *start*). Ketika dua arah pencarian telah membangkitkan simpul yang sama, maka solusi telah ditemukan, yaitu dengan cara menggabungkan kedua jalur yang bertemu.

2.3.3.2 Pencarian *Heuristic* (terbimbing)

Pencarian buta biasanya tidak efisien karena waktu akses dan memori yang di butuhkan cukup besar. Untuk mengatasi hal ini maka perlu ditambahkan suatu informasi pada domain yang bersangkutan sehingga proses pencarian yang baru akan dihasilkan. Pencarian seperti ini disebut sebagai *informed search* atau pencarian *heuristic* (terbimbing) yaitu pencarian berdasarkan panduan.

Teknik pencarian *heuristic* (*heuristic searching*) merupakan suatu strategi untuk melakukan proses pencarian secara selektif dan dapat memandu proses pencarian yang memiliki kemungkinan sukses paling besar, namun dengan kemungkinan mengorbankan kelengkapan (*completeness*).

Untuk menerapkan pencarian *heuristic* diperlukan suatu fungsi *heuristic*. Fungsi *heuristic* adalah aturan-aturan yang digunakan untuk mendapatkan solusi yang diinginkan. Pencarian *Heuristic* terdiri dari : *Generate And Tes* (Pembangkitan dan Pengujian), *Hill Climbing* (Pendakian Bukit) : *Simple Hill Climbing, Steepest-Ascent Hill Climbing, Best First Search,* Algoritma A* (A *Star*), *Simulated Anneliang* [10].

2.3.3.3 Algoritma A * (A *Star*)

AI yang akan digunakan dalam pembangunan *Game* Pertolongan Pertama ini adalah AI yang mengunakan metode pencarian yang memecahkan masalah menggunakan algoritma A* (A *Star*). Algoritma ini merupakan algoritma *Best First Search* yang menggabungkan *Uniform Cost Search* dan *Greedy Best First Search.* Biaya yang diperhitungkan didapat dari biaya sebenarnya ditambah dengan biaya perkiraan. Dalam notasi matematika dituliskan sebagai berikut :

$$f(n) = g(n) + h(n) \quad \ldots(2.1)$$

Dengan keterangan dimana nilai :

f (n) merupakan fungsi evaluasi (jumlah *g(n)* dengan *h(n)*)

g(n) merupakan ongkos yang diperlukan atau biaya sebenarnya yang dikeluarkan dari keadaan awal sampai keadaan n,

h(n) adalah biaya perkiraan atau estimasi biaya untuk sampai pada suatu tujuan mulai dari n.

Algoritma A* menerapkan teknik *heuristic* dalam membantu penyelesaian persoalan. *Heuristic* adalah penilai yang memberi harga pada tiap simpul yang memandu A* mendapatkan solusi yang diinginkan. Dengan *Heuristic* yang benar, maka A* pasti akan mendapatkan solusi (jika memang ada solusinya) yang dicari. *Heuristic* yang digunakan adalah *heuristic manhattan.* Perhitungannya dapat ditulis sebagai berikut :

h (n) = (abs(n.x–goal.x) + (abs(n.y–goal.y)) … (2.3)

Beberapa terminologi dasar yang terdapat pada algoritma A* yaitu [10]:

1. Simpul (*node*) adalah petak-petak kecil sebagai representasi dari area pencarian (pathfinding). Bentuknya dapat berupa persegi, lingkaran, maupun segitiga.

2. Simpul asal / mulai (source *node*) adalah sebuah terminologi untuk posisi awal sebuah benda.

3. Simpul akhir / tujuan (destination *node*) adalah tempat tujuan yang ingin dicapai pada pencarian.

4. Simpul sekarang (current *node*) adalah simpul terbaik sebelumnya yang dipilih dan menjadi titik acuan untuk membangkitkan simpul tetangganya.

5. Open list adalah tempat menyimpan data simpul yang diakses dari simpul asal / mulai (source *node*) atau dari tetangga simpul sekarang (current *node*) yang belum pernah berada di open list maupun closed list.

6. Closed list adalah tempat menyimpan simpul yang pernah menjadi simpul sekarang (current *node*).

7. Halangan/penghalang adalah simpul yang tidak dapat dilalui.

Cara menerapkan terminologi A* diatas adalah dengan cara menyimpan *source node* ke dalam *open list* dan *closed list*, karena simpul ini terpilih sebagai *Bestnode,* kemudian bangkitkan semua suksesor dari simpul tersebut. Jika semua suksesornya tidak ada di *open list* maupun di *closed list,* maka masukan suksesornya ke *open list.* Semua suksesor cek nilai *f(n)* nya, nilai yang digunakan atau yang dipilih adalah nilai *f(n)* yang terkecil. Setelah itu *current node* simpan suksesornya dan *current node* tersebut menjadi *parent* yang baru. Begitu seterusnya hingga mencapai simpul tujuan (*destination node*). Jika suksesor yang dibangkitkan sudah ada di *open list* maupun di *closed list* maka abaikan suksesor tersebut.

2.4 OOP (*Object Oriented Programming*)

Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya, yang merupakan suatu acara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek didasarkan pada penerapan prinsip-prinsip pengolahan kompleks meliputi rangkaian aktivitas analisis berorientasi objek, perencanaan berorientasi objek, pemograman berorientasi objek, dan pengujian berorientasi objek [12].

Pendekatan berorientasi objek merupakan suatu teknik atau cara pendekatan dalam melihat permasalahan dan sistem (sistem perangkat lunak, sistem informasi dan lainnya) yang akan mendukung sistem yang akan dikembangkan sebagai suatu kumpulan objek yang berkorespondensi dengan objek-objek dunia nyata. Setiap komponen dalam sistem tersebut dapat mewarisi atribut dan sifat komponen lainnya, dan dapat berinteraksi satu sama lain.

Enam keuntungan menggunakan OOP : Alami (*Natural*), dapat diandalkan (*Reliable*), dapat digunakan kembali (*Reusable*), mudah untuk dalam perawatan (*Maintainable*), dapat diperluas (*Extendable*), efisiensi waktu.

Berikut ini beberapa bahasa pemrograman yang sudah menggunakan konsep OOP, adalah [12]:

1. *C++*.
2. *Visual C++*.
3. *Visual Basic*.
4. *Java*.

Berikut adalah beberapa konsep dasar yang harus dipahami tentang metodologi berorientasi objek [12] :

1. Kelas (*class*)

   Kelas adalah kumpulan objek – objek dengan karakteristik yang sama. Kelas merupakan definisi statis himpunan objek yang sama yang mungkin lahir atau diciptakan dan kelas tersebut. Sebuah kelas akan mempunyai sifat (atribut), kelakuan (operasi/metode), hubungan (relationship) dan arti. Suatu

kelas dapat diturunkan dan kelas yang lain, di mana atribut dan kelas semula dapat diwariskan kekelas yang baru. Secara teknis, kelas adalah sebuah struktur tertentu dalam pembuatan perangkat lunak.

2. Objek (*Object*)

   Objek adalah abstraksi dan sesuatu yang mewakili dunia nyata seperti benda, manusia, satuan organisasi, tempat, kejadian, struktur, status atau hal-hal lain yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan) yang dapat diterapkan atau dapat berpengaruh pada status objeknya. Objek mempunyai siklus hidup yaitu diciptakan, dimanipulasi, dan dihancurkan

3. Metode (*Method*)

   Operasi atau metode pada suatu kelas hampir sama dengan fungsi prosedur pada metodologi struktural. Sebuah kelas boleh memiliki lebih dari satu metode atau operasi. Metode atau operasi yang berfungsi untuk memanipulasi objek itu sendiri. Operasi atau metode merupakan fungsi atau transformasi yang dapat dilakukan terhadap objek atau dilakukan oleh objek. Metode atau operasi dapat berasal dari *event,* aktivitas atau aksi keadaan, fungsi, atau kelakuan dunia nyata.

4. Atribut (*attribute*)

   Atribut dari sebuah kelas adalah variabel global yang memiliki sebuah kelas. Atribut dapat berupa nilai atau elemen-elemen data yang dimiliki oleh objek dalam kelas objek. Atribut dipunyai secara individual oleh sebuah objek, misalnya berat, jenis, nama dsb.

5. Abstraksi (*abstraction*)

   Prinsip untuk mempresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.

6. Pembungkusan (*Encapsulation*)

   Pembungkusan atribut data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.

7. Pewarisan (*Inheritance*)

   Mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dari dirinya.

8. Antarmuka (*interface*)

   Antarmuka atau *interface* sangat mirip dengan kelas, tapi tanpa atribut kelas dan memiliki metode yang dideklarasikan tanpa isi. Deklarasi metode pada sebuat *interface* dapat diimplementasikan oleh kelas lain.

9. *Reusability*

   Pemanfaatan kembali objek yang sudah didefinisikan untuk suatu permasalahan pada permasalahan lainnya yang melibatkan objek tersebut.

10. Generalisasi dan Spesialisasi

    Menunjukan hubungan antara kelas dan objek yang umum dengan kelas dan objek yang khusus.

11. Komunikasi Antarobjek

    Komunikasi antarobjek dilakuka lewat pesan (*message*) yang dikirim dari satu objek ke objek lainnya.

12. Polimorfisme (*polymorphism*)

    Kemampuan suatu objek untuk digunakan dibanyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program.

13. *Package*

    Package adalah sebuah kontainer atau kemasan yang dapat digunakan untuk mengelompokkan kelas-kelas sehingga memungkinkan beberapa kelas yang bernama sama disimpan dalam *package* yang berbeda.

2.5 Tools yang Digunakan

Ada banyak tools yang dapat dipergunakan untuk membuat *Game* dan perancangannya, diantaranya adalah UML (*Unified Modeling Language),* adobe flash, corel draw, photoshop, dll.

2.5.1 UML (*Unified Modeling Language)*

Unified Modelling Language (UML) adalah salah satu standar bahasa yang banyak digunakan didunia industri untuk mendefinisikan *requirement,* membuat analisis dan desain, serta menggambarkan arsitektur dalam pemograman berorientasi objek [12].

UML diaplikasikan untuk maksud tertentu, biasanya antara lain untuk [12]:

1. Merancang perangkat lunak.
2. Sarana komunikasi antara perangkat lunak dengan proses bisnis.
3. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasi sistem yang ada, proses-proses dan organisasinya.

Blok pembangun utama UML adalah diagram. Beberapa diagram ada yang rinci, dan ada yang bersifat umum. Dengan demikian UML merupakan alat komunikasi yang konsisten dalam mensupport para pengembang sistem saat ini [13].

2.5.1.1 Diagram UML

Umumnya sebuah system mempunyai sejumlah *stakeholder* (orang yang mempunyai ketertarikan pada suatu sitem namun dari sudut pandang yang berbeda). Sebagai contoh ketika sedang merancang sebuah system untuk seorang pelanggan, maka akan sangat berbeda jika sitem tersebut diterapkan pada banyak pelanggan. Dari sini jelas dibutuhkan banyak diagram dari sudut pandang yang berbeda, dengan tujuan untuk memuaskan semua *stakeholder*.

UML mempunyai sejumlah elemen grafis yang bisa dikombinasikan menjadi diagram. Karena ini merupakan sebuah bahasa, UML mempunyai sejumlah aturan untuk menggabungkan atau mngkombinasikan elemen – elemen tersebut. Berikut penjelasan dari setiap jenis atau tipe diagram UML [14] :

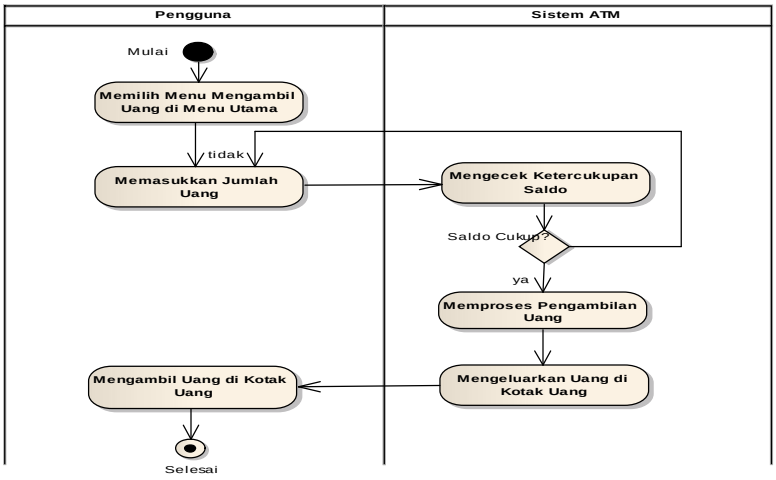
Tabel 2.1 Tipe Diagram UML

Diagram	Tujuan
Activity	Perilaku procedural & parallel
Class	*Class*, fitur dan relasiya
Communication	Interaksi diatara obyek. Lebih menekankan ke link
Component	Struktu dan koneksi dari komponen
Composite structure	Dekomposisi sebuah *class* saat runtime
Deployment	Penyebaran atau instalasi ke klien
Interaction Overview	Gabungan antara *activity & Sequence Diagram*
Object	Contoh konfigurasi instance
Package	Struktur hirarki saat kompilasi
Sequence	Interaksi antar obyek
State machine	Bagaimana *event* mengubah sebuah obyek
Timing	Interaksi antar obyek
Use case`	Bagaimana user berintraksi dengan sebuah system.

Berikut keterangan lebih lanjut dari diagram yang telah disebutkan pada tabel diatas yang akan dipergunakan pada penelitian ini [12]:

1. Diagram Activity

   Diagram aktivitas menggambarkan workflow atau aliran kerja atau aktifitas dari sebuah sistem. Hal yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan actor, jadi aktivitas yang dapat dilakukan oleh sistem. Berikut adalah contoh diagram aktivitas pada studi kasus mengambil uang di ATM.



Gambar 2.1 Studi Kasus Mengambil Uang di ATM

Diagram *activity* memiliki simbol – simbol khusus, pada tabel 2.8 berikut ini merupakan simbol – simbol yang diunakan dalam diagram *activity*.

Tabel 2.2 Simbol pada Diagram *Activity*.

NO	GAMBAR	NAMA	KETERANGAN
1		*Actifity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling beraktifitas / berinteraksi satu sama lain.
3		*Initial Node* / status awal	Bagaimana objek dibentuk atau diawali.
4		*Actifity Final Node* / status akhir	Bagaimana objek dibentuk dan dihancurkan atau dapat di ibaratkan sebagai status akhir yang dilakukan oleh sistem.
5		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran
6		*Join* / penggabungan	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.

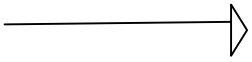
2. Diagram kelas (*class diagram*).

   Diagram ini menggambarkan struktur system dari segi pendefinisian kelas – kelas yang akan dibuat untuk membangun system. Kelas memiliki apa yag disebut atribut dan metode atau operasi. Dimana atribut merupakan variable

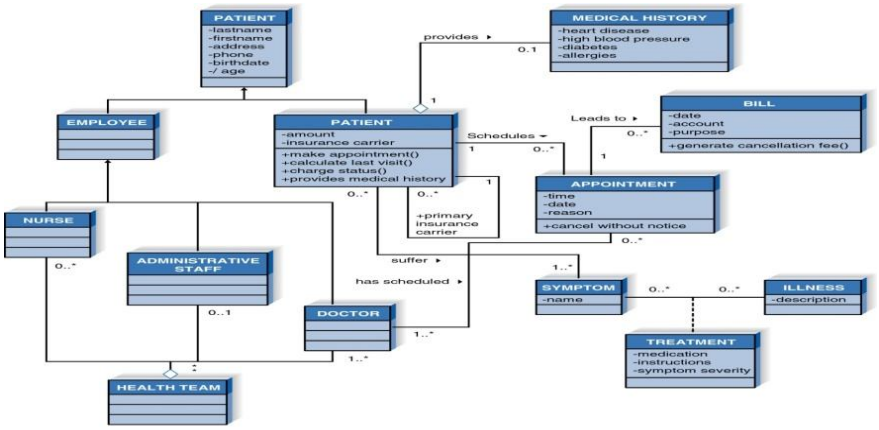
– variable yang dimiliki oleh suatu kelas dan operasi atau metode adalah fungsi – fungsi yang dimiliki oleh suatu kelas.

Diagram kelas mempunyai symbol – symbol tersendiri yang akan disebutkan pada tabel 2.3 berikut :

Tabel 2.3 Simbol – Simbol yang Ada pada Diagram Kelas

Simbol	Deskripsi
Kelas 	Kelas pada struktur sistem
Antarmuka / *interface*  nama _interface	Sama dengan konsep *interface* dalam pemograman berorientasi objek
Asosiasi */association*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan multiplicity
Asosiasi berarah/ *directed association*	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas lainnya, asosiasi biasanya juga disertai dengan *multiplicity*
Generalisasi 	Relasi antar kelas dengan makna generalisasi – spesialisasi (umum – khusus)
Kebergantungan / dependency 	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi / aggregation 	Relasi antar kelas dengan makna semua - bagian (whole – part)

Contoh studi kasus diagram kelas:

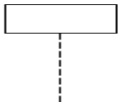


Gambar 2.2 Studi kasus Diagram Kelas

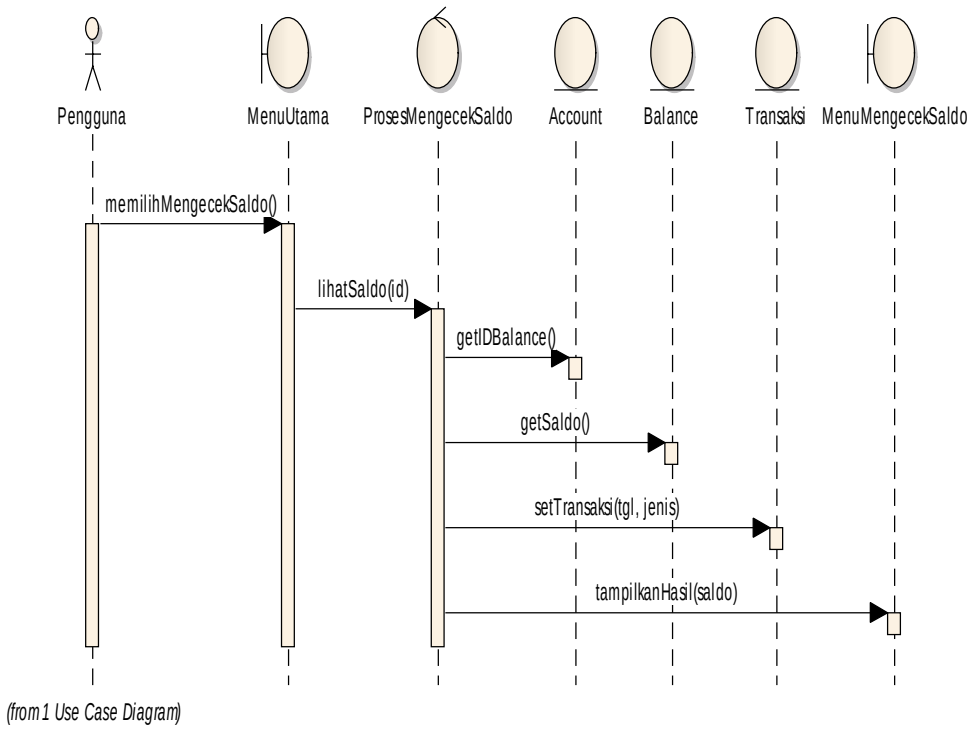
3. Diagram *sequence*

   Diagram *sequence* berfungsi untuk menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan message atau pesan yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram *sequence* harus diketahui objek – objk yang terlibat dalam sebuah *use case* beserta metode – mtode yang dimiliki kelas yang diinisialisasi menjadi objek tersebut. Diagram *sequence* mempunyai simbol untuk mewakili penggambaran fungsi nya seperti yang dijelaskan pada tabel 2.4 berikut ini.

Tabel 2.4 Simbol Diagram S*equence*

NO	GAMBAR	NAMA	KETERANGAN
1		*LifeLine*	Objek *entity*, antarmuka yang saling berinteraksi.
2		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi
3		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi

Berikut adalah gambaran contoh *Sequence Diagram* pada studi kasus pengecekan saldo ATM

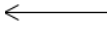
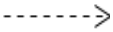
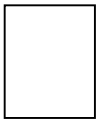


Gambar 2.3 Studi Kasus Pengecekan Saldo ATM

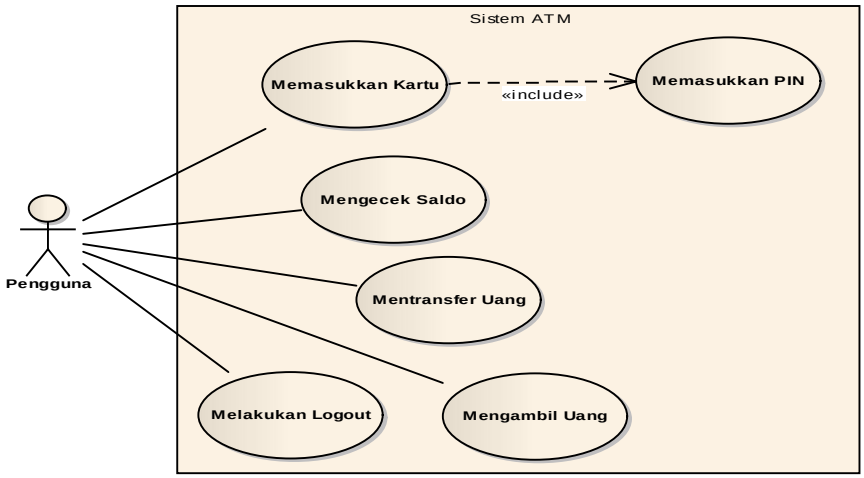
4. Diagram *Use-case*.

   Diagram *Use-case* merupakan pemodelan dari system yang akan dibuat. *Use-case m*endeskripsikan sebuah interaksi antara satu atau lebih actor dengan system yang akan dibuat. Secara kasar, *Use-case* digunakan untuk mengetahui fungsi apa saja ang terdapat dalam system dan siapa saja yang berhak menggunakan fungsi – fungsi tersebut. Pada tabel 2.11 akan dijelaskan terlebih dahulu mengenai simbol – simbol yang terdapat pada diagram *Use-case.*

Tabel 2.5 Simbol – Simbol pada *Use-case*

NO	GAMBAR	NAMA	KETERANGAN
1		***Actor***	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
3		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
4		*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit.*
5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor

Berikut merupakan contoh studi kasus perancangan *Use-case* yang digunakan dalam aplikasi system ATM



Gambar 2.4 Studi Kasus *Usecase* Sistem ATM

2.5.2 Star UML

StarUML adalah software permodelan yang mendukung UML *(Unified Modeling Language).* Berdasarkan pada UML version 1.4 dan dilengkapi 11 macam diagram yang berbeda, mendukung notasi UML 2.0 dan juga mendukung pendekatan MDA *(Model Driven Architecture)* dengan dukungan konsep UML. StarUML dapat memaksimalkan pruduktivitas dan kualitas dari suatu *software project.*

Konsep Dasar pada StarUML :

1. *Model, View and Diagram*

   StarUML membuat perbedaan konseptual yang lebih jelas antara *models, views and diagrams.* Model adalah elemen yang memuat informasi untuk model software. *View* adalah suatu ekspresi visual dari informasi di dalam model dan Diagram adalah suatu koleksi dari elemen yang memberikan pemikiran user di dalam mendesain secara spesifik.

2. *Project and Unit*

   Project adalah unit manajemen dasar di dalam StarUML. Suatu *projec*t dapat mengatur satu atau lebih model software. *Project* merupakan top-level

package yang selalu ada di dalam beberapa model software. Secara umum, satu project disimpan dalam satu file.

File project disimpan ke dalam format XML dengan extension ".UML". Semua model, views dan diagrams yang dibuat dengan StarUML disimpan dalam satu file project. File project berisikan informasi sebagai berikut :

a. UML profile yang digunakan dalam projek.
b. Unit file yang direferensi oleh projek.
c. Informasi untuk semua model yang ada di dalam project.
d. Informasi untuk semua diagrams dan views yang ada di dalam project.

Ada beberapa kasus dimana satu project perlu disimpan di dalam beberapa file – file kecil sehingga para pengembang dapat bekerja di dalam satu project secara bersamaan. Di dalam kasus ini suatu project dapat mengatur bermacam – macam unit. Suatu unit mempunyai struktur hirarki dan berisikan beberapa sub-unit. Unit disimpan sebagai ".UML " file dan beberapa mengacu pada file project (.UML) atau unit file lainnya (.UNT). Komposisi Unit Hanya package, subsystem dan elemen model yang dapat membentuk satu unit. Semua elemen di bawah jenis elemen package ini disimpan di dalam masing – masing file unit (.UNT).

3. *Module*

Modul adalah suatu package yang menyediakan fungsi – fungsi baru dan fitur sebagai perluasan dari StarUML. Modul dapat dibuat sebagai kombinasi dari beberapa elemen – elemen extension dan juga membuat beberapa jenis elemen – elemen di dalam suatu modul[14].

2.5.3 Adobe Flash

Adobe Flash (dahulu bernama Macromedia Flash) adalah salah satu perangkat lunak komputer yang merupakan produk unggulan *Adobe Systems*. Adobe flash digunakan untuk membuat gambar vektor maupun animasi gambar tersebut. Berkas yang dihasilkan dari perangkat lunak ini mempunyai extension .swf dan dapat diputar dipenjelajah web yang telah dipasang Adobe Flash *Player*.

Flash menggunakan bahasa pemograman yang bernama Action Script yang muncul pertama kalinya pada Flash 5.

Adobe flash merupakan sebuah program yang didesain khusus oleh Adobe dan program aplikasi standar *authoring tool profesional* yang digunakan untuk membuat animasi dan bitmap yang sangat menarik untuk keperluan membangun situs web yang interaktif dan dinamis. Flash didesain dengan kemampuan untuk membuat animasi 2 dimensi yang andal dan ringan sehingga Flash banyak digunakan untuk membangun dan memberikan efek animasi pada website, CD interaktif, *Game* dan yang lainnya.

Dalam flash terdapat teknik-teknik membuat animasi, fasilitas action script, filter, custom easing dan dapat memasukan vidio lengkap dengan fasilitas playback FLV, dan dapat disisipkan code pemograman dan database dengan pendekatan XML[15].

2.5.4 Adobe Photoshop CS 5

Adobe Photoshop, atau biasa disebut Photoshop, adalah perangkat lunak editor citra buatan *Adobe Systems* yang dikhususkan untuk pengeditan foto/gambar dan pembuatan efek. Perangkat lunak ini banyak digunakan oleh fotografer digital dan perusahaan iklan sehingga dianggap sebagai pemimpin pasar (*Market Leader*) untuk perangkat lunak pengolah gambar/foto, dan, bersama Adobe Acrobat, dianggap sebagai produk terbaik yang pernah diproduksi oleh *Adobe Systems*. Versi kedelapan aplikasi ini disebut dengan nama Photoshop CS (*Creative Suite*), versi sembilan disebut Adobe Photoshop CS2, versi sepuluh disebut Adobe Photoshop CS3, versi kesebelas adalah Adobe Photoshop CS4 dan versi yang terakhir (kedua belas) adalah Adobe Photoshop CS5. Photoshop tersedia untuk Microsoft Windows, Mac OS X, dan Mac OS versi 9 ke atas juga dapat digunakan oleh sistem operasi lain seperti Linux dengan bantuan perangkat lunak tertentu seperti *CrossOver*.

Tool dalam Adobe Photoshop adalah alat yang dapat membantu pengguna dalam mengedit. Adobe Photoshop CS3 memiliki 59 *tool* yang dapat dipakai oleh

pengguna. *Tool* tersebut terdiri dari berbagai macam *tool* dengan kegunaan yang spesifik [18].

2.5.5 Adobe Audition

Adobe Audition merupakan suatu program yang digunakan untuk merekam, mengedit suara dalam bentuk *digital* yang berbasis Windows. Program ini dilengkapi dengan modul-modul efek suara, seperti *delay*, *echo*, Pereduksi *noise, reverb*, pengatur tempo, *pitch, graphic* dan *parametic equalizer*.

Adobe Audition memberikan fasilitas perekaman suara sampai dengan 128 *track* hanya dengan suara *sound card*, hal ini akan memberikan kemudahan bagi penggunanya untuk mengatur sesuai keinginan. Edit suara bisa dilakukan dalam bentuk .wav dan penyimpanan bisa di *convert* dalam bentuk format seperti .wma, mp3, dll

2.6 Pengujian Aplikasi atau perangkat lunak

Pengujian aplikasi bertujuan untuk mencari kesalahan (*bugs*) dalam sebuah aplikasi. Pengujian yang baik adalah pengujian yang memiliki kemungkinan besar dalam menemukan kesalahan. Oleh karena itu dalam merancang dan mengimplementasikan sistem, baik berbasis komputer atau produk harus ada pengujiannya. Menurut James Bach mengenai karakteristik kemampuan suatu aplikasi dapat diuji (*testability*) sebagai berikut : “kemampuan perangkat lunak untuk dapat diuji adalah seberapa mudahkah sebuah program computer untuk bisa diuji”[17]. Agar lebih mempermudah pemahaman menggenai kemampuan aplikasi dapat diuji, akan dijelaskan dengan mengenalkan tentang kemampuan sebuah perangkat lunak untuk bisa diuji seperti berikut :

1. Kemampuan untuk bisa dioperasikan *(operability).*
   Semakin baik kinerjanya, semakin efisien perangkat lunak untuk bisa diuji.
2. Kemampuan untuk bisa diobservasi (*observability*).
   Apa yang dilihat, maka itulah yang diuji. Masukan (input) tersedia sebagai bagian dari pengujian yang menghasilkan keluaran (output) yang berbeda. Bagian dari variable sistem terlihat atau dapat di pertanyakan selama

eksekusi, sehinggan keluaran yang salah dapat dengan mudah diidentifikasi serta kesalahan internal dapat secara otomatis dideteksi dan dilaporkan.

3. Kemampuan untuk dapat dikontrol (*controllability*).
   Semakin baik sebuah aplikasi dapat dikontrol, maka pengujian akan semakin dapat diotomatisasi dan di optimalkan.
4. Kemampuan untuk dapat disusun (*decomposability*).
   Dengan mengontrol ruang ingkup pengujian, maka akan lebih cepat untuk mengisolasi masalah dan melakukan pengujian ulang dengan lebih baik.
5. Kesederhanaan (*simplicity*).
   Semakin sedikit yang diuji maka akan semakin cepat pengujiannya. Program harus menunjukan kesederhanaan fungsional misalnya fitur –fitur dan kesederhanaan kode program.
6. Stabilitas *(stability*).
   Semakin sedikit perubahan, maka semakin sedikit gangguan untuk pengujian.
7. Kemampuan untuk dapat dipahami (*understandability)*.

Setelah mengetahui tentang kemampuan aplikasi dapat diuji berdasarkan dari pendapat James Bach, selanjutnya yang perlu diketahui adalah karakteristik mengenai pengujian. Karakteristik pengujian menurut Kaner, Falk, dan Nguyen mengambarkan atribut – atribut pengujian yang baik sebagai berikut [17] :

1. Pengujian yang baik memiliki probabilitas tinggi untuk menemukan kesalahan.
   Agar mencapai tujuan ini, penguji harus memahami perangkat lunak dan mencoba untuk mengembangkan sebuah gambaran mental bagaimana suatu perangkat lunak bisa gagal. Idealnya kelas kegagalan dapat diselidiki.
2. Pengujian yang baikt tidak berulang – ulang.
   Waktu dan sumber daya pngujian terbatas. Tidak ada gunanya melakukan pengujian yang memiliki tujuan yang sama dengan pengujian yang lain. Setip pengujian harus memiliki tujuan yang berbeda (bahkan jika itu hanya sedikit berbeda).

3. Pengujian yang baik harus menjadi bibit terbaik.

   Dalam sebuah kelompok pengujian yang memiliki tujuan serupa, keterbatasan waktu dan sumber daya dapat mengurangi pelaksanaan bahkan hanya sebagian kecil dari pengujian ini. Dalam hal kasus seperti itu pengujian yang memiliki kemungkinan tertinggi dalam mengungkap seluruh kelas kesalahan harus digunakan.

4. Pengujian yang baik tidak harus selalu sederhana ataupun rumit.

   Meskipun terkadang keadaan sangat memungkinkan untuk menggabungkan serangkaian pengujian mejadi satu kasus pengujian saja, dengan efek samping yang biasanya terjadi adalah banyaknya kesalahan yang harus ditutupi. Maka secaraa umum , setiap pengujian harus dilaksanakan secara terpisah.

Setiap produk rekayasa dapat diuji dalam satu dari dua cara berikut :

1. Mengetahui fungsi yang telah ditentukan

   Dengan mengetahui fungsi – fungsi suatu produk yang telah dirancang untuk bekerja, maka pengujian pun dapat dilakukan untuk menunjukan bahwa masing – masing fungsi sepenuhnya dapat beroperasi dengan baik, sementara pada saat yang sama juga dapat mencari kesalahan pada setiap fungsi.

2. Dengan mengetahui cara kerja internal sebuah produk.

   Pengujian dapat dilakukan untuk memastikan bahwa semua perseneling telah terhubung, dengan mengetahui operasi – operasi internal telah dilakukan sesuai dengan spesifikasi dan semua komponen internal telah memadai untuk dieksekusi. Pendekatan pengujian prtama kali membutuhkan pandangan eksternal dan disebut dengan pengujian kotak – hitam *(black - box testing,)* dan yang kedua membutuhkan pandangan internal yang disebut dengan kotak – putih *(white – box testing)*.

2.6.1 Pengujian kotak hitam *(Black – box testing)*

Pengujian kotak hitam, juga disebut pengujian perilaku, yang berfokus pada persyaratan fungsional perangkat lunak. Artinya, teknik pengujian kotak hitam memungkinkan untuk membuat beberapa kumpulan kondisi masukan yang sepenuhnya akan melakukan semua kebutuhan fungsional untuk program. Pengujian kotak hitam bukan teknik alternative untuk pengujian kotak putih. Sebaliknya, ini merupakan pendekatan pelengkap yang mungkin dilakukan untuk mengungkap kelas yang kesalahan yang berbeda dari yang diungkapkan oleh metode pngujian kotak putih.

Pengujian kotak hitam berupaya untuk mnemukan kesalahan dalam kateori berikut :

1. Fungsi yang salah atau hilang
2. Kesalahan antarmuka
3. Kesalahan dalam struktur data atau akses basis yang salah atau hilang
4. Kesalahan perilaku dan kinerja
5. Kesaahan inisialisasi dan penghentian [17].

2.6.2 Pengujian kotak putih *(White – box testing)*.

Pengujian kotak putih atau yang terkadang disbut sebagai pengujian kotak kaca, merupakan filosofi perancangan test case yang menggunakan struktur control yang dijelaskan sebagai bagian dari perancangan peringkat komponen untuk menghasilkan test case. Dengan menggunakan metode pengujian kotak putih , dapat diperoleh hasil :

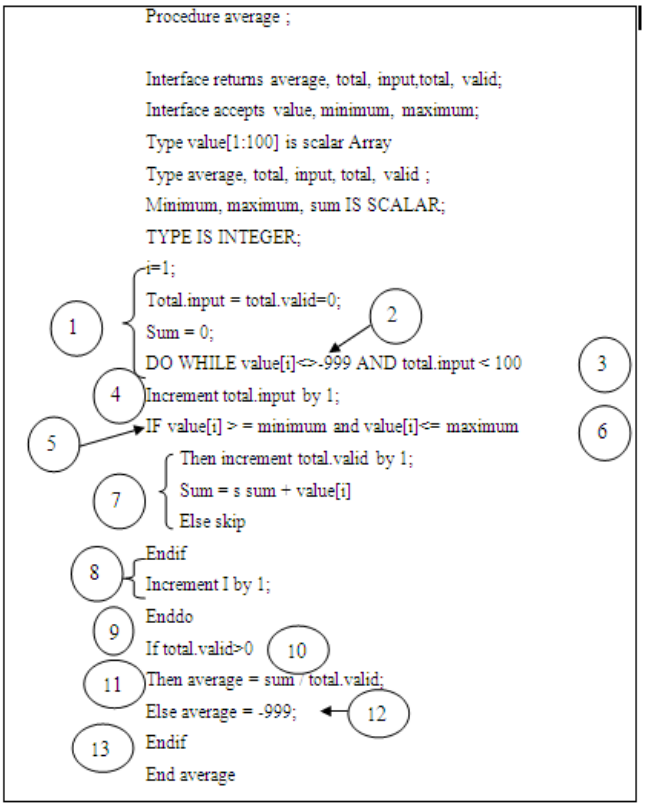
1. Test case yang menjamin bahwa semua jalur independen di dalam modul telah di eksekusi setidaknya satu kali
2. Test case yang melaksanakan semua keputusan logis pada sisi benar dan yang salah
3. Test case yang melaksanakan semua loop pada batas mereka dan dalam batas – batas operasional mereka.
4. Test case yang melakukan struktur data internal untuk memastikan kesahihannya.

2.6.2.1 Menghasilkan Test Case

Metode pengujian jalur – dasar dapat diterapka untuk perancangan procedural atau kode program. Prosedur average, yang digambarkan dalam PDL berikut yang akan digunakan sebagai contoh untuk mengilustrasikan masing – masing langkah dalam metode perancangan *test case*. Hal yang perlu diingat walaupun test case merupakan algoritma yang sangat sederhana, akan tetapi didalamnya terdapat kondisi gabungan dan perulangan. Beberapa langkah berikut ini dapat menurunkan basis set.

1. Membuat grafik alir yang sesuai dengan menggunakan perancangan atau kode sebagai sebuah dasar.

   Grafik alir dibuat dengan menggunakan simbol dan aturan konstruksi yang mengacu pada PDL untuk average. Grafik alir diciptakan dengan member nomor – nomor pada pernyataan PDL yang akan dipetakan ke *node* grafik alir yang sesuai. Seperti contoh gambar PDL berikut :



Gambar 2.5 PDL Procedure Average

2. Menentukan kompleksitas siklomatik dari aliran grafik alir

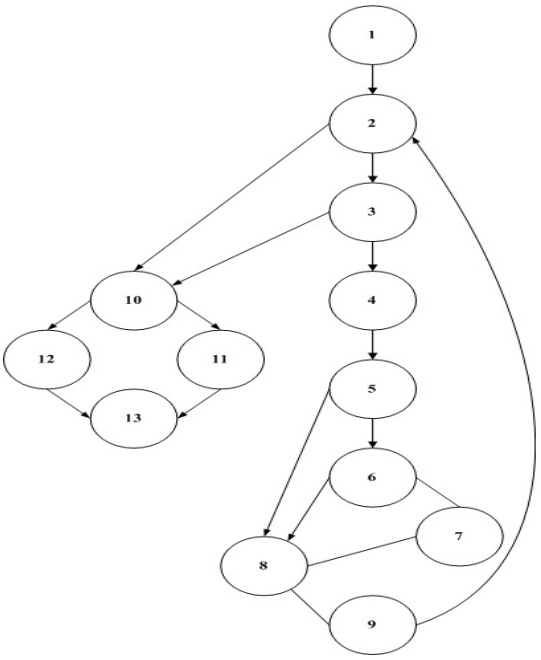
   Kompleksitas siklomatik *V(G)* ditentukan dengan menerapkan algoritma – algoritma yang sudah dijelaskan dalam PDL, kemudian menghitung semua pernyataan – pernyataan kondisional dalam PDL (untuk procedure *average).*
3. Menentukan basis set dari jalur indipenden linier.

   Nilai *V(G)* memberikan batas atas pada jalur independen linear melalui struktur program pengendalian.
4. Menyiapkan test case yang akan memaksa pelaksanaan setiap jalur di basis set.

   Data harus dipilih sehingga kondisi – kondisi di *node* predikat adalah set yang tepat saat masing – masing jalur diuji. Setiap test case dieksekusi dan dibandingkan dengan hasil yang diharapkan.

2.6.2.2 Notasi grafik alir

Dalam pengujian kotak putih, notasi sederhana untuk representasi aliran kotrol yang disebut grafik alir (grafik program / flow graph) menggambarkan arus control logis dengan menggunakan notasi yang diilustrasikan pada gambar 2.10 berikut :



Gambar 2.6 Grafik Alir

Penjelasan dari gambar diatas adalah :

1. Setiap lingkaran, disebut simpul (*node*) grafik alir, yang merupakan sat atau lebih pernyataan – pernyataan procedural.
2. Panah atau yang disebut *edge / link* pada grafik, merupakan aliran kendali dan analog dengan panah diagram alir. Sebuah *edge / link* harus berhenti pada sebuah *node*, bahkan jika *node* tidak mewakili pernyataan – pernyataan proseudral.
3. Area yang dibatasi oleh edge dan *node* disebut *region*. Ketika menghitung *region,* harus dimasukan pula area di luar grafik sebagai *region*.

Ketika kondisi – kondisi gabungan ditemui dalam perancangan procedural pembuatan sebuah grafik alir menjadi sedikit lebih rumit. Sebuah kodisi gabungan terjadi ketika satu atau lebih operator boolan (logika OR, AND, NAND, NOR) hadir dala sebuah pernyataan bersyarat[17].

2.6.2.3 Jalur program independen

Jalur independen adalah jalur yang melalui program yang memprkenalan setidaknya satu kumpulan pernyataan – pernyataan pemrosesan atau kondisi baru. Bila dinyatakan dalam grafik alir jalur independen harus bergerak sepajang setidaknya satu edge yang belum dilintasi sebelum jalur trsebut diefinisikan. Contohnya, satu set jalur independen untuk grafik alir yang diilustrasikan pada gambar 2.10 adalah :

Path 1 : 1-2-10-11-13
Path 2 : 1-2-10-12-13
Path 3 : 1-2-3-10-11-13
Path 4 : 1-2-3-4-5-8-9-2
Path 5 : 1-2-3-4-5-6-8-9-2
Path 6 : 1-2-3-4-5-6-7-8-9-2

Pada ilustrasi diatas dapat diperhatikan bahwa setiap jalur baru, memperkenalkan edge yang baru. Jalur / path 1 hingga 6 merupakan basis set pada grafik alir yang artinya, dapat melakukan rancangan pengujian dengan memaksakan pelaksanaan jalur satu kali dan setiap kondisi telah dilaksanakan dalam posisi true dan false.

Hal yang perlu diingat bahwa basis set tidak lah unik, bahkan sejumlah basis set yang berbeda dapat diturunkan dari perancangan procedural yang diberikan.

Cara mengetahui berapa banyak jalur yang dapat dilalui adalah dengan melakukan perhitungan kompleksitas siklomatik. Kompleksitas siklomatik adalah metric perangkat lunak yang menyediakan ukuran kuantitatif dari kompleksitas logis suatu program. Bila digunakan dalam konteks metode pengujian jalur dasar, nilai yang dihitung untuk kompleksitas siklomatik mendefinisikan jumlah jalur independent dalam sebuah basis set suatu program dan menyediakan batas atas untuk sebuah pengujian yang harus dilakukan guna memastikan bahwa semua pernyataan telah di eksekusi minima sekali. Kompleksitas siklomatik dilandaskan pada teori graph. Kompleksitas dihitung dalam salah satu cara dari tiga cara berikut :

1. Jumlahkan daerah – daerah grafik alir yang berhubungan dengan kompleksitas siklomatik.
2. Kompleksitas siklomatik *V (G)* untuk grafik alir *G* didefinisikan sebagai
   $V(G) = E - N + 2$ ….(2.3)
   Dimana *E* adalah jumlah edge grafik alir dan *N* adalah jumlah *node* grafik alir.
3. Kompleksitas siklomatik *V(G)* untuk grafik aliran *G* juga di definisikan sebagai $V(G) = P + 1$ ….(2.4)
   Dimana *P* adalah jumlah *node* predikat yang terdapat dalam grafik aliran *G*

Contoh kompleksitas siklomatik yang diterapka pada gambar 2.10 adalah :

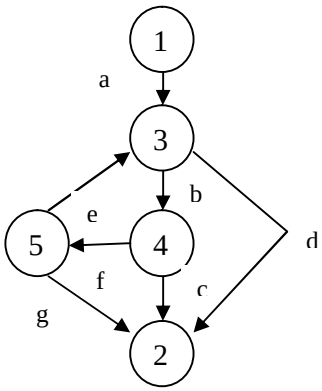
1. Grafik aliran memilki 6 *region.*
2. V(G) = 17 tepi – 13 *node* + 2 =6
3. V(G)= 5 *node* predikat + 1 = 6

Berdasarkan contoh perhitungan diatas nilai kompleksitas siklomatik dari diagram alir pada gambar 2.10 adalah 6.

2.6.2.4 Matrik – matrik grafik

Matrik grafik adalah matriks persegi yang ukurannya (jumlah, baris, dan kolom) sama dengan jumlah *node* pada grafik alir. Setiap baris dan kolom terkait dengan *node* yang diidentifikasi, dan masukan matriks terkait dengan koneksi – koneksi (*edge*) yang berada diantara *node – node*.

Contoh sederhana grafik alir dan grafik matrix adalah sebagai berikut [17].



Grafik alir

Node \ Terhubung ke *node*	1	2	3	4	5
1			a		
2					
3		d		b	
4		c			f
5		g	e		

Grafik matrix

Gambar 2.7 Grafik Alir dan Matrix

2.6.3 Pengujian Alpha

Pengujian alpha dilakukan disisi pengembang oleh sekelompok perwakilan dari pengguna akhir. Perangkat lunak ini diguanakan dalam posisi natural, dimana pengembang “melihat dengan kacamata” pengguna dan mencatat kesalahan – kesalahan dan masalah - masalah yang timbul.

2.6.4 Pengujian Beta

Pengujian *beta* merupakan pengujian yang dilakukan secara objektif, dimana dilakukan pengujian secara langsung terhadap pengguna dengan menggunakan kuesioner mengenai kepuasan pengguna atas aplikasi yang telah dibangun. Adapun metode penilaian pengujian yang digunakan adalah metode kuantitatif berdasarkan data dari pengguna.

Dalam penelitian kuantitatif peneliti menggunakan istrumen untuk mengumpulkan data. Creswell (2012) menyatakan bahwa " peneliti kuantitatif dalam mengumpulkan data menggunakan instrument merupakan alat untuk mengukur, mengobservasi yang dapat mnghasilkan data kuantitatif.

Instrument penelitian digunakan untuk mengukur nilai variable yang diteliti, dengan demikian jumlah instrument yang akan digunakan untuk penelitian akan tergantung pada jumlah variable yang diteliti. Bila variabelnya lima, maka jumlah instrument nya juga lima. Instrument – instrument penelitian sudah ada yang dibakukan, tetapi masih ada yang harus dibuat oleh peneliti sendiri. Instrument penelitian digunakan untuk melakukan pengukuran dengan tujuan menghasilkan data kuantitatif yang akurat, maka setiap instumen harus mempunyai skala. Skala pengukuran terdiri dari banyak macam yaitu :

1. *Skala Likert*
2. *Skala Guttman*
3. *Rating scale*
4. *Semantic Deferential*

Dalam pembagunan aplikasi *Game* pertolongan pertama ini skala pengukuran yang digunakan adalah skala *Likert.*

2.6.4.1 Skala *Likert*

skala *Likert* digunakan untuk megukur sikap, pendapat, dan persepsi seseorang atau sekelompok orang tentang sebuah fenomena. Dalam penelitian, fenomena ini telah ditetapkan secara spesifik oleh peneliti, yang selanjutnya disebut sebagai variable penelitian.

Jawaban setiap instrument yang menggunakan skala likert mempunyai gradasi dari sangat positif sampai sangat negative, contohnya :

a. Sangat setuju (SS)	a. selalu
b. Setuju (ST)	b. sering
c. Ragu – ragu (RG)	c. kadang – kadang
d. Tidak setuju (TS)	d. tidak pernah
e. Sangat tidak setuju (STS)	

Untuk keperluan analisis kuantitatif, maka setiap jawaban tersebut dapat di beri skor, misalnya :

1. Sangat Setuju diberi skor 5
2. Setuju/ selalu diberi skor 4
3. Ragu –ragu / sering diberi skor 3
4. Tidak setuju / kadang – kadang diberi skor 2
5. Sangat tidak setuju/ tidak pernah 1

Selanjutnya dengan teknik pengumpulan data angket, maka instrument tersebut misalnya diberikan pada 100 karyawan yang diambil secara random. Dari hasil analisis terhadap 100 orang karyawan missal terdapat hasil analisis seperti berikut:

25 orang menjawab SS
40 orang menjawab ST
5 orang menjawab RG
20 orang menjawab TS
10 orang menjawab STS

Berdasarkan data tersebut 65 orang (40 + 25) atau 65% karyawan menjawab setuju dan sangat setuju.

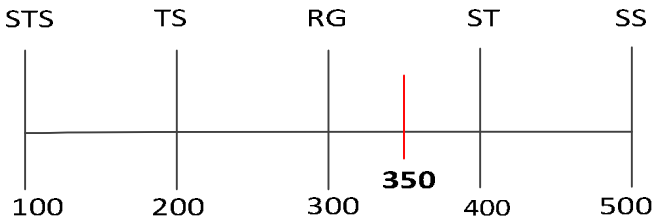
Data interval tersebut juga dapat dianalisis dengan menghitung rata – rata jawaban berdasarkan scoring setiap jawaban dari responden. Berdasarkan skor yang telah ditetapkan dapat dihitung sebagai berikut :

Jumlah skor untuk 25 orang yang menjawab SS = 25 x 5 = 125
Jumlah skor untuk 40 orang yang menjawab ST = 40 x 4 = 160
Jumlah skor untuk 5 orang yang menjawab RG = 5 x 3 = 15
Jumlah skor untuk 20 orang yang menjawab TS = 20x 2 = 40
Jumlah skor untuk 10 orang yang menjawab STS = 10 x 1 = 10

jumlah total = 150

Jumlah skor ideal (kriterium) untuk seluruh item = 5 x 100 = 500 (seandainya semua menjawab SS). Jumlah skor yang diperoleh dari penelitian = 350. Jadi berdasrkan data itu maka tingkat persetujuan karyawan = (350 : 500) x 100 % =

70% dari yang diharapkan (100%). Secara kontinum dapat digambarkan seperti berikut :



Jadi berdasarkan data yang diperoleh dari 100 responden maka rata – rata 350 terletak pada daerah setuju [18].