BAB II
TINJAUAN PUSTAKA

2.1 Diare

Penyakit Diare masih merupakan masalah kesehatan masyarakat di Indonesia, karena angka kesakitan dan kematian yang masih tinggi. Untuk mengatasi hal tersebut Departemen Kesehatan telah menerbitkan buku Pedoman Pengendalian Penyakit Diare berdasarkan Keputusan Menteri Kesehatan Republik Indonesia dengan nomor 1216/MENKES/SK/XI/2001.Diare adalah suatu kondisi dimana seseorang buang air besar dangan konsistensi lembek, cair, bahkan dapat berupa air saja dan frekuensinya lebih sering (biasanya tiga kali atau lebih) dalam suatu hari [1].

2.1.1 Penyebab Diare

Secara klinis penyebab diare dapat dikelompokan dalam 6 golongan yaitu infeksi (disebabkan oleh bakteri, virus atau infestasi parasit), malabsorpsi, alergi, keracunan, imunodefiesiensi dan sebab-sebab lainnya, Penyebab yang sering ditemukan dilapangan ataupun secara klinis adalah diare yang disebabkan infeksi dan keracunan [1].

2.1.2 Jenis-Jenis Diare

Jenis diare ada dua, yaitu Diare akut, Diare persisten atau Diare kronik. Diare akut adalah diare yang berlangsung kurang dari 14 hari, sementara Diare persisten atau diare kronik adalah diare yang berlangung lebih dari 14 hari [1].

2.1.3 Pencegahan Diare

Cara melakukan pencegahan diare yang benar dan efektif dengan cara meminum air yang sudah direbus dan menggunakan air bersih yang cukup dalam aktivitas sehari-hari, mencuci tangan dengan air dan sabun sebelum makan dan

sesudah buang air besar, membuang sampah pada tempatnya agar tidak menjadi sarang berkumpulnya lalat sebagai faktor penularan penyakit diare [1].

2.2 *Game*

Menurut Elliot Avedon dan Brian Sutton-Smith, *game* adalah bentuk latihan dari sistem yang dilakukan secara sukarela, dimana ada sebuah kompetisi diantara kekuatan (kekuasaan), dibatasi oleh aturan untuk memproduksi hasil yang tidak seimbang (ada menang dan kalah).

Pertama, "bentuk latihan dari sistem yang dilakukan secara sukarela" artinya, game masuk dengan sengaja dan sukarela tanpa ada paksaan.

Kedua, "kompetisi kekuasaan", ini tampaknya menjadi bagian dari kebanyakan game. Dua atau lebih pemain berjuang untuk mendominasi. Namun frasa ini akan melintasi dua hal: game memiliki tujuan, dan game memiliki konflik.

Ketiga, "dibatasi oleh aturan": Sebuah poin yang sangat penting. Permainan memiliki aturan. Aturan yang pasti salah satu aspek yang menentukan permainan.

Keempat, "hasil yang tidak seimbang", artinya sebagai gantinya berarti bahwa pada satu waktu ada keseimbangan, tapi itu kemudian hilang. Hal ini memang benar dari permainan yang ada untuk menentukan menang atau kalah [3].

2.2.1 Klasifikasi *Game*

Berikut ini adalah klasifikasi game berdasarkan platform atau alat yang digunakan dan berdasarkan genre permainannya.

2.2.2 Berdasarkan Jenis *Platform* atau Alat yang Digunakan

Berdasarkan jenis platform atau alat yang digunakan, game terdiri dari beberapa jenis, yaitu [4]:

1. *Game* PC / Desktop

   Game yang dimainkan pada PC (*Personal Computer*) yang memiliki kelebihan yaitu memiliki tampilan antarmuka yang baik untuk input maupun output, output visual kualitas tinggi karena layar computer biasanya memiliki resolusi yang jauh lebih tinggi dibandingkan dengan layar televise biasa. Kekurangannya adalah spesifikasi computer yang sangat bervariasai antara satu computer dengan computer lainnya menyebabkan beberapa *game* dapat ditampilkan dengan baik pada satu computer, tetapi tidak berjalan dengan baik pada computer yang lainnya.

2. *Game Console*

   Game yang dijalankan pada suatu mesin spesifik yang biasanya tersedia di rumah pribadi, seperti Microsoft Xbox 360, Nintendo Wii dan sebagainya.

3. *Game Arcade*

   Game yang dijalankan pada mesin dengan input dan output audio visual yang telah terintegrasi dan tersedia ditempat-tempat umum, seperti mal, bandara dan sebagainya.

4. *Game Online*

   Game yang hanya dapat dimainkan secara *online* melalui LAN atau internet.

5. *Mobile Games*

   Game yang dapat dimainkan atau khusus untuk *mobile phone* atau PDA.

2.2.3 Berdasarkan Representasi Visualnya

Berdasarkan pada representasi visualnya, game dapat dibedakan menjadi 2 jenis, yaitu [4]:

1. Game 2 dimensi (2D)

   Game 2D adalah game, yang secara matematis, hanya melibatkan 2 elemen koordinat kartesius, yaitu x dan y, sehingga konsep kamera pada game 2D hanya menentukan gambar mana pada game yang dapat dilihat oleh pemain.

2. Game 3 dimensi (3D)

   Game 3D adalah game yang selain melibatkan elemen x dan y, juga melibatkan elemen z pada perhitungannya, sehingga konsep kamera pada game 3D benar-benar menyerupai konsep kamera pada kehidupan nyata, yaitu selain digeser (seperti padagame 2D), juga dapat diputar dengan sumbu tertentu.

2.2.4 Berdasarkan Genre Permainannya

Berdasarkan genre permainannya, game terdiri dari beberapa genre, yaitu [4]:

1. *Action*

   Tembak - tembakan , atau hajar - hajaran bisa juga tusuk - tusukan, tergantung cerita dan tokoh di dalamnya), *video game* jenis ini sangat memerlukan kecepatan refleks, koordinasi mata-tangan, juga timing, inti dari *game* jenis ini adalah tembak, tembak dan tembak. Termasuk di dalam-nya :

   a. *First person shooting (FPS)* seperti *Counter Strike* dan *Call of Duty*.

   b. *Drive n shoot*, menggunakan unsur simulasi kendaraan tetapi tetap dengan tujuan utama menembak dan menghancurkan lawan, contoh : *Spy Hunter, Rock and Roll Racing, Road Rash*.

   c. *Shoot em up*, seperti *Raiden*, *1942*, dan *gradius*.

d. *Beat em up* (tonjok hajar) seperti *Double Dragon* dan *Final Fight*, lalu *hack and slash* (tusuk tebas) seperti *Shinobi* dan *Legend of Kage*.

e. *Light gun shooting,* yang menggunakan alat yang umumnya berbentuk seperti senjata, seperti *Virtua Cop* dan *Time Crisis*.

2. *Fighting*

inti dari *game* ini adalah penguasaan jurus (hafal caranya dan lancar mengeksekusinya), pengenalan karakter dan timing sangatlah penting, *combo*-pun menjadi esensial untuk mengalahkan lawan secepat mungkin. Dan berbeda seperti *game* Aksi pada umumnya hanya melawan *Artificial Intellegence* atau istilah umumnya *melawan komputer* saja, pemain jenis *fighting game* ini baru teruji kemampuan sesungguhnya dengan melawan pemain lainnya. Seri *Street Fighter*, *Tekken*, *Mortal Kombat*, *Soul Calibur* dan *King of Fighter* adalah contohnya.

3. *Education*

Video Game jenis ini dibuat dengan tujuan spesifik sebagai alat pendidikan, entah untuk belajar mengenal warna untuk balita, mengenal huruf dan angka, matematika, sampai belajar bahasa asing. *Developer* yang membuatnya harus memperhitungkan berbagai hal agar *game* ini benar-benar dapat mendidik, menambah pengetahuan dan meningkatkan ketrampilan yang memainkannya. Target segmentasi pemain harus pula disesuaikan dengan tingkat kesulitan dan design visual ataupun animasinya. Contoh edu*game*s : *Bobi Bola*, *Dora the explorer*, Petualangan Billy dan Tracy.

4. *Adventure*

Bedanya dengan jenis *video game* aksi-petualangan, refleks dan kelihaian pemain dalam bergerak, berlari, melompat hingga memecut atau menembak tidak diperlukan di sini. *Video Game* murni petualangan lebih menekankan pada jalan cerita dan kemampuan berpikir pemain dalam

menganalisa tempat secara visual, memecahkan teka-teki maupun menyimpulkan rangkaian peristiwa dan percakapan karakter hingga penggunaan benda-benda tepat pada tempat yang tepat. Termasuk di dalamnya:

a. *Petualangan dengan teks* atau sistem tunjuk dan klik, contoh: *Kings Quest*, *Space Quest*, *Heroes Quest*, *Monkey Island*, *Sam and Max*,

b. *Novel atau film interaktif*, seperti *game* "*dating*" yang banyak beredar di jepang, *Dragons Lair* dan *Night Trap*.

5. Racing

Dari namanya sudah jelas, siapa sampai duluan di garis finish dialah pemenangnya! Terkadang malah pemain dapat memilih kendaraan, mendandani, upgrade mesin bahkan mengecatnya. Contoh: *Top Gear*, *Test Drive*, *Sega Rally Championship*, *Daytona*, *Grand Turismo*, *Need For Speed*, *Mario Cart*, *ManXTT*.

6. *Strategy*

Kebalikan dari *video game* jenis *action* yang berjalan cepat dan perlu refleks secepat kilat, *video game* jenis strategi, layaknya bermain catur, justru lebih memerlukan keahlian berpikir dan memutuskan setiap gerakan secara hati-hati dan terencana. *Video game* strategi biasanya memberikan pemain atas kendali tidak hanya satu orang tapi minimal sekelompok orang dengan berbagai jenis tipe kemampuan, sampai kendaraan, bahkan hingga pembangunan berbagai bangunan, pabrik dan pusal pelatihan tempur, tergantung dari tema ceritanya. Pemain *game* strategi melihat dari sudut pandang lebih meluas dan lebih kedepan dengan waktu permainan yang biasanya lebih lama dan santai dibandingkan *game action*. Unsur-unsur permainannya biasanya berkisar sekitar, prioritas pembangunan, peletakan pasukan, mencari dan memanfaatkan sumberdaya (uang, besi, kayu,minyak,dll), hingga ke

pembelian dan peng-*upgrade*-an pasukan atau teknologi. *Game* jenis ini terbagi atas:

a. *Real time Strategy, game* berjalan dalam waktu sebenarnya dan serentak antara semua pihak dan pemain harus memutuskan setiap langkah yang diambil saat itu juga berbarengan mungkin saat itu pihak lawan juga sedang mengeksekusi strateginya. Contoh: *Starcraft, Warcraft ,* dan *Command and Conquer*.

b. *Turn based Strategy , game* yang berjalan secara bergiliran, saat kita mengambil keputusan dan menggerakan pasukan, saat itu pihak lawan menunggu, begitu pula sebaliknya, layaknya catur. contoh: *Front Mission, Super robot wars, Final Fantasy tactics, Heroes of might and magic, Master of Orion*.

7. Simulation, Construction and Management

Video Game jenis ini seringkali menggambarkan dunia di dalamnya sedekat mungkin dengan dunia nyata dan memperhatikan dengan detil berbagai faktor. Dari mencari jodoh dan pekerjaan, membangun rumah, gedung hingga kota, mengatur pajak dan dana kota hingga keputusan memecat atau menambah karyawan. Dunia kehidupan rumah tangga sampai bisnis membangun konglomerasi, dari jualan limun pinggir jalan hingga membangun laboratorium cloning. *Video Game* jenis ini membuat pemain harus berpikir untuk mendirikan, membangun dan mengatasi masalah dengan menggunakan dana yang terbatas. Contoh: *Sim City*, *The Sims, Tamagotchi.*

8. Puzzle

Video game jenis ini sesuai namanya berintikan mengenai pemecahan teka-teki, baik itu menyusun balok, menyamakan warna bola, memecahkan perhitungan matematika, melewati labirin, sampai mendorong-dorong kota masuk ke tempat yang seharusnya, itu semua termasuk dalam jenis ini. Sering pula permainan jenis ini adalah juga

unsur permainan dalam *video game* petualangan maupun *game* edukasi. *Tetris, Minesweeper, Bejeweled, Sokoban dan Bomberman.*

9. *Sport*

Singkat padat jelas, bermain *sport* di PC atau konsol. Biasanya permainannya diusahakan serealistik mungkin walau kadang ada yang menambah unsur fiksi seperti *NBA JAM*. Contohnya pun jelas, *Seri Winning Eleven, seri NBA, seri FIFA, John Madden NFL, Lakers vs Celtics, Tony hawk pro skater,* dll.

10. *RPG (Roler Playing Game)*

Video game jenis ini sesuai dengan terjemahannya, bermain peran, memiliki penekanan pada tokoh/peranperwakilan pemain di dalam permainan, yang biasanya adalah tokoh utamanya, dimana seiring kita memainkannya, karakter tersebut dapat berubah dan berkembang ke arah yang diinginkan pemain (biasanya menjadi semakin hebat, semakin kuat, semakin berpengaruh, dll) dalam berbagai parameter yang biasanya ditentukan dengan naiknya **level**, baik dari status kepintaran, kecepatan dan kekuatan karakter, senjata yang semakin sakti, ataupun jumlah teman maupun mahluk peliharaan.Secara kebudayaan, pengembang *game* Jepang biasanya membuat *Role Playing Game (**RPG**)* ke arah cerita linear yang diarahkan seolah karakter kita adalah tokoh dalam cerita itu, seperti *Final Fantasy, Dragon Quest dan Xenogears*. Sedangkan pengembang *game **RPG*** Eropa, cenderung membuat karakter kita bebas memilih jalan cerita sendiri secara non-linear, seperti *Ultima, Never Winter Nights, baldurs gate, Elder Scroll, dan Fallout*.

2.2.5 *Game* Edukasi

Game edukasi merupakan permainan digital yang dapat memberikan kesempatan untuk bermain melalui lingkungan simulasi dan dapat menjadi bagian integral dari pembelajaran dan pengembangan intelektual. Sampai akhir abad 19 game diasosiasikan dengan hiburan, tetapi setelah mendapatkan pengaruh dari

John Dewey pada tahun 1944, game mulai memegang peranan dalam teknologi pengajaran.

1. Tujuan *Game* Edukasi

Game edukasi mampu membantu masyarakat dalam pengembangan akhlak, intelektual, motivasi, keahlian, kecakapan. Agar tujuan pembelajaran menggunakan *game* edukasi ini tepat sasaran, pada tahun 1997 Calvo mengemukakan teori bahwa *game* harus mampu meningkatkan fungsi-fungsi berikut [3]:

a. Pengembangan Motorik

Game harus banyak melibatkan gerakan untuk meransang keakuratan gerakan, koordinasi gerakan dan kecepatan gerakan.

b. Pengembangan Intelektual

Game harus melibatkan pemahaman bagaimana sesuatu bekerja, pemecahan masalah, pengambilan strategi, pengambilan keputusan dan lain-lain.

c. Pengembangan Afektif

Game harus mampu merangsang siswa untuk memahami pengalam kehidupan mereka dan membantu mereka menjadi dewasa.

d. Pengembangan Sosial

Game harus mampu mengakomodasi interaksi antar pemain dalam bentuk-bentuk simbol dan aturan-aturan sehingga siswa yang bermain.

2.3 *Artificial Intelligence*

AI atau *Artifical Inteligence* pertama kali dikemukakan pada tahun 1956 di konferensi Darthmouth. Sejak saat itu, AI terus dikembangkan sebab berbagai penelitian mengenai teori-teori dan prinsip-prisipnya juga terus berkembang.

Terdapat beberapa definisi AI, menurut Stuart Russel dan Peter Norvig mengelompokan definisi AI, kedalam empat kategori. Pertama *Thinking humanly* dan *Acting humanly* adalah dua definisi dalam arti yang sangat luas. Saat ini, pemikiran manusia yang di luar rasio, yakni refleks dan inuitif (berhubungan dengan perasaan), belum dapat ditiruka sepenuhnya oleh komputer. Kedua *thinking rationally* dan *acting rationally* merupakan dasar pemikiran bahwa komputer juga bisa melakukan aksi secara rasional berdasarkan hasil penalaran tersebut [5].

2.3.1 Teknik-Teknik Dasar Pencarian

Pencarian merupakan salah satu teknik dari AI untuk menyelesaikan beberapa permasalahan. Keberhasilan suatu sistem salah satunya ditentukan oleh kesuksesan dalam pencarian dan pencocokan. Berikut adalah contoh aplikasi yang menggunakan teknik pencarian ini, yaitu :

a. *Game* papan dan puzzle, contoh: tic-tac-toe, catur, dan menara Hanoi,
b. Penjadwalan dan masalah *routing* pada *travelling salesman problem,*
c. Parsing bahasa dan inteprestasinya pada masalah pencarian struktur dan arti kata,
d. Logika pemrograman dalam hal pencarian fakta dan implikasinya,
e. *Computer vision* dan pengenalan pola,
f. Sistem pakar bebasis kaidah (*rule based expert system*).

Pencarian adalah proses mencari solusi dari suatu permasalahan melalui sekumpulan kemungkinan ruang keadaan (*state space*). Ruang keadaan merupakan suatu ruang yang terdapat semua keadaan yang mungkin. Kondisi suatu pencarian meliputi :

a. Keadaan sekarang atau awal,
b. Keadaan tujuan atau solusi yang dijangkau dan perlu diperiksa apakah telah mencapai sasaran,

c. Biaya (*cost*) yang diperoleh dari solusi.

Solusi merupakan suatu lintasan dari keadaan awal sampai keadaan tujuan. Secara umum, proses pencarian dapat dilakukan seperti berikut :

a. Memeriksa keadaan sekarang atau awal
b. Mengeksekusi aksi yang dibolehkan untuk memindahkan ke keadaan berikutnya
c. Memeriksa jika keadaan baru merupakan solusinya. Jika tidak, keadaan baru tersebut menjadi keadaan sekarang dan proses ini diulangi sampai solusi ditemukan atau ruang keadaan habis terpakai.

2.3.2 Algoritma Pencarian (*Search Algorithm*)

Berbagai algoritma untuk pencarian (*search algorithm*) yang ada berbeda satu dengan yang lain dalam hal pengembangan kumpulan node untuk mencapai *goal state*. Perbedaan ini terutama dalam hal cara dan urutan pengembangan node, dan sangat berpengaruh pada kinerja masing-masing algoritma. Empat kriteria yang menjadi ukuran algoritma pencarian adalah :

a. *Completeness*, apakah algoritma pasti dapat menemukan solusi?
b. *Time Comlexity*, berapa lama waktu yang dibutuhkan untuk menemukan sebuah solusi?
c. *Space Complexity*, berapa memori atau *resource* yang diperlukan untuk melakukan pencarian?
d· *Optimality*, apakah algoritma tersebut dapat menemukan solusi yang terbaik jika terdapat beberapa solusi yang berbeda?

Permasalahan pencarian dapat diselesaikan dengan 2 golongan, yaitu:

a. *Uninformed Search/Blind Search,* merupakan pencarian solusi tanpa adanya informasi yang dapat mengarahkan pencarian untuk mencapai *goal state* dari *current state* disebut juga pencarian buta. Beberapa contoh algoritma tersebut antara lain adalah *Breadth First Search, Uniform Cost Search, Depth First Search, Depth*

Limited Search, Iterative Deepening Search dan Bidirectional Search.

b. *Informed Search/Heuristic Search* merupakan pencarian solusi dengan adanya informasi tentang biaya (*cost*) yang dapat mengarahkan pencarian untuk mencapai *goal state* dari *current state*. Dengan informasi tersebut, dapat melakukan pertimbangan untuk mengembangkan atau memeriksa kumpulan node yang mengarah ke *goal state*. Beberapa contohnya adalah *Best First Search, Greedy Search, Djikstra*, A* (A *Star*) *Search*, dan *Hill Climbing Search*.

2.3.3 Algoritma A *Star*

Algoritma A* diperkenalkan pertama kali oleh Peter Hart, Nils Nilsson dan Bertram Raphael pada tahun 1968. Algoritma A* merupakan format pencarian heuristik untuk menghitung efisiensi solusi optimal. A* merupakan algortima *Best First Search* yang menggabungkan *Uniform Cost Search* dan *Greedy Best First Search* [14].

Terdapat terminologi dasar yaitu *strating point*, simpul (*node*), A, *openlist, closedlist*, harga (*cost*), halangan (*unwalkable*).

Starting point adalah sebuah terminologi untuk posisi awal sebuah benda. Simpul adalah petak-petak kecil sebagai representasi dari area *pathfinding*. *Openlist* merupakan tempat menyimpan data simpul yang mungkin diakses dari *starting point* maupun simpul yang sedang dijalankan. *Closedlist* adalah tempat menyimpan data simpul sebelum A yang juga merupakan bagian jalur terpendek yang telah berhasil didapatkan. Sedangkan A adalah simpul yang sedang dijalankan dalam algoritma pencarian jarak terpendek.

Pada algoritma A* terdapat fungsi evaluasi terhadap *node* n adalah

$$\mathbf{f(n) = g(n) + h(n)} \qquad \mathbf{(2.1)}$$

f adalah fungsi evaluasi yang diperoleh dari penjumlahan nilai g (*actual cost*) dan h (*heuristic cost*). g adalah jumlah nilai setiap simpul dalam jalur terpendek dari *starting point* ke A. h adalah jumlah nilai perkiraan dari simpul ke simpul tujuan.

Pencarian jarak terpendek menggunakan algoritma A* memiliki prinsip yang sama dengan algoritma BFS, hanya saja dengan dua faktor tambahan.

a. Setiap sisi mempunyai *cost* yang berbeda-beda, seberapa besar *cost* untuk pergi dari satu simpul ke simpul yang lain.
b. *Cost* dari setiap simpul ke simpul tujuan bisa diperkirakan. Ini membantu pencarian, sehingga lebih kecil kemungkinan mencari ke arah yang salah.

Cost untuk setiap simpul tidak harus berupa jarak. *Cost* bisa saja berupa waktu bila ingin mencari jalan dengan waktu tercepat untuk dilalui.

Pada Algoritma A* memiliki langkah-langkah sebagai berikut :

a. Masukkan simpul awal ke *Open List*.
b. Ulangi langkah berikut sampai pencarian berakhir.
c. Cari *node* n dengan nilai f(n) paling rendah, dalam *Open List*. *Node* ini akan menjadi *current node*.
d. Keluarkan *current node* dari *Open List* dan masukkan ke *Closed List*.
e. Untuk setiap suksesor dari *current node* lakukan langkah berikut :

   Jika sudah terdapat dalam *Closed List*, abaikan, jika tidak lanjutkan.

   Jika belum ada pada *Open List*, masukkan ke Open List. Simpan *current node* sebagai parent dari suksesor-suksesor ini. Simpan *cost* masing-masing simpul.

   Jika belum ada dalam *Open List*, periksa jika simpul suksesor ini mempunyai nilai lebih kecil dibanding suksesor sebelumnya. Jika lebih kecil, jadikan sebagai *current node* dan ganti *parent node* ini.
f. Walaupun telah mencapai simpul tujuan, jika masih ada suksesor yang memiliki nilai yang lebih kecil, maka simpul tersebut akan

terus dipilih sampai bobotnya jauh lebih besar atau mencapai simpul akhir dengan bobot yang lebih kecil dibanding dengan simpul sebelumnya yang telah mencapai simpul tujuan.

g. Pada setiap pemilihan simpul berikutnya, nilai f(n) akan dievakuasi, dan jika terdapat nilai f(n) yang sama maka akan dipilih berdasarkan nilai g(n) terbesar.

2.4 Bahasa Pemrograman

ActionScript adalah bahasa pemrograman/skrip yang dimiliki oleh Flash. Bahasa ini menunjuk ke salah satu style programming yang merujuk ke ECMAScript 4, merupakan basisnya JavaScript. ActionScript menunjukan koleksi set dari *action, function, event dan event handler* yang memungkinkan dikembangkan oleh para developer untuk membuat flash movie yang lebih komplek dan interaktif. ActonScript mengalami evolusi kearah standar bahasa pemrograman, yaitu versi 1, versi 2 dan versi.

Sebuah flash movie dapat terdiri dari beberapa scenes. Masing-masing scene mempunyai sebuah timeline. Masing-masing timeline dimulai dengan frame 1 dan seterusnya. Secara normal state sebuah flash movie bersifat dapat berpindah dari scene 1 frame 1dan berakhir pada scene 1 dan scene 2 dan seterusnya. Kita dapat mengeset movie berjalan dari awal sampai akhir frame dan akhir disebuah scene, serta menghentikannya. Kita dapat mengeset movie tersebut untuk berulang.

Tujuan utama ActionScript adalah mengubah kebiasaan linier tersebut. Sebuah ActionScript dapat menghentikan sebuah movie di frame tertentu, dan berulang ke frame sebelumnya, atau frame mana yang akan ditampilkan tergantung masukan yang diberikan user. ActionScript dapat digunakan untuk membuat sebuah movie komplek, bukan berbentuk linier (standar).

Akan tetapi, tidak semua flash movie memerlukan ActionScript. Berikut fungsi dasar yang dapat dilakukan oleh ActionScript [8]:

1. Animation

   Tidak memeerlukan ActionScript jika hanya membuat animasi sederhana. Tetapi script dapat membantu membuat animasi yang komplek. Sebagai contoh, sebuah bola dapat memantul pada layer disekeliling layer yang mengabaikan prinsip fisika. Tetapi jika bola itu memantul ke tanah maka akan memerlukan prinsip gravitasi. Tanpa ActionScript, kita akan membuat animasi tersebut sebanyak ratusan frame. Namun, dengan ActionScript dapat dilakukan hanya dengan satu frame.

2. Navigation

   Secara default bergerak ke depan satu demi satu frame sampai selesai. Dengan ActionScript, kita dapat membuat menu untuk berhenti disembarang frame dan meneruskan ke frame sesuai pilihan dari user.

3. User Input

   Kita dapat memberikan konfirmasi (pertanyaan) ke user untuk meminta masukan dan mengirimkan informasi tersebut ke server. Sebuah flash movie dengan beberapa ActionScript dapat digunakan untuk membangun aplikasi web.

4. Memperoleh Data

   Sebuah ActionScript dapat berinteraksi dengan server. Kita dapat meng-update informasi dan menampilkannya ke user.

5. Calculation

   ActionScript dapat melakukan kalkulasi, , misalnya dapat diterapkan pada aplikasi shopping chart.

6. Graphic

   ActionScript dapat mengubah ukuran graphic, sudut rotasi, warna movie clip dalam movie, serta dapat menduplikasi dan menghapus item dari screen.

7. Dapat mengenali Environment

   ActionScript dapat mengambil nilai waktu dari sistem yang digunakan oleh user

8. Memutar Music

   Memutar music dengan ActionScript adalah sebuah alternatif yang bagus. ActionScipt dapat mengontrol balance dan volume.

2.5 OOP (*Object Oriented Programming*)

OOP (*Object Oriented Programming*) atau yang dikenal dengan Pemrograman Berorientasi Objek merupakan teknik dalam menyelesaikan masalah yang kerap muncul dalam pengembangan perangkat lunak. Orientasi objek merupakan teknik pemodelan sistem riil yang berbasis objek. Inti dari sistem ini adalah objek yang merupakan model dari sistem nyata. Semua data dan fungsi di dalam paradigma ini dibungkus ke dalam kelas-kelas atau objek-objek. Model data berorientasi objek dikatakan dapat memberi fleksibilitas yang lebih, kemudahan mengubah program, dan digunakan luas dalam teknik piranti lunak skala besar.

Pemrograman orientasi-objek menekankan konsep berikut [9]:

1. Objek (object)

   Objek mendefinisikan fakta. Objek merepresentasikan sistem real seperti siswa, sistem kontrol permukaan sayap pesawat, sensor atau mesin. Objek juga merepresentasikan sesuatu dalam bentuk konsep seperti nasabah bank, merek dagang, pernikahan atau sekedar listing. Bahkan bisa juga menyatakan visualisasi seperti, bentuk huruf (font), histogram, poligon, garis atau lingkaran. Semuanya memiliki fitur atribut (untuk data), behaviour (operation atau method), keadaan (memori), identitas dan tanggung jawab. Proses menjabarkan sistem nyata menjadi objek dinamakan abstraksi (abstraction). Abstraksi mengeliminir aspek yang tidak perlu dalam suatu objek.

2. Kelas (class)

   Kelas adalah penggambaran satu set objek yang memiliki atribut dan behaviour yang sama atau sejenis. Kelas mendefinisikan aturan- aturan. Kelas mirip tipe data pada pemrograman non objek, tetapi lebih komrehensif karena terdapat struktur sekaligus karakteristiknya. Kita

dapat membentuk kelas baru yang lebih spesifik dari kelas generalnya. Kelas dan objek merupakan jantung dari pemrograman berorientasi objek. Untuk menghasilkan program jenis ini sangat penting untuk selalu berfikirdalam konsep objek.

3. Pembungkusan (encapsulation)

   Pembungkusan sebagai penggabungan potongan-potongan informasi dan perilaku – perilaku spesifik yang bekerja pada informasi tersebut, kemudian mengemasnya menjadi apa yang disebut sebagai objek.

4. Pewarisan (Inheritance) dan Generalisasi / Spesialisasi

   Konsep dimana metode dan atau atribut yang ditentukan di dalam sebuah objek kelas dapat diwariskan atau digunakan lagi oleh objek kelas lainnya. Sedangkan generalisasi/spesialisasi merupakan teknik dimana atribut dan perilaku yang umum pada beberapa tipe kelas objek, dikelompokan (diabstraksi) kedalam kelasnya sendiri (dinamakan supertype). Atribut dan metode kelas objek supertype kemudian diwariskan oleh kelas objek tersebut (dinamakan subtype).

5. Polimorfisme

   Polimorfisme berarti suatu fungsionalitas yang diimplementasikan dengan berbagai cara yang berbeda. Pada terminologi berorientasi objek, ini berarti kita dapat memiliki berbagai implementasi untuk sebagian funsionalitas tertentu. Manfaat dari polimorfisme adalah kemudahan pemeliharaanya. Jika perlu menambahkan gambar baru maka cukup menambahkan fungsi baru sedangkan fungsi umumnya tidak mengalami perubahan.

2.6 UML (*Unified Modeling Language*)

UML singkatan dari *Unified Modeling Language* yang berarti bahasa pemodelan standar. Unified Modeling Language (UML) adalah himpunan struktur dan teknik untuk pemodelan desain program berorientasi objek (OOP) serta aplikasinya. UML merupakan dasar bagi perangkat (Tool) desain berorientasi objek dari IBM. UML dikembangkan sebagai suatu alat untuk analisis dan desain

berorientasi objek oleh Grady Booch, Jim Rumbaugh, dan Ivar Jacobson. Ketika membuat model menggunakan konsep UML ada aturan – aturan yang harus diikuti. Bagaimana elemen pada model – model yang kita buat berhubungan satu dengan lainnya harus mengikuti standar yang ada. UML bukan hanya sekedar diagram, tetapi juga menceritakan konteksnya.

UML diaplikasikan untuk maksud tertentu, biasanya antara lain untuk [9]:

a. Merancang perangkat lunak.
b. Sarana komunikasi antara perangkat lunak dengan proses bisnis.
c. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
d. Mendokumentasi sistem yang ada, proses – proses dan organisasinya.

Blok pembangun utama UML adalah diagram. Beberapa diagram ada yang rinci (jenis timing diagram) dan lainnya ada yang bersifat umum (misalnya diagram kelas). Para pengembang sistem berorientasi objek menggunakan bahasa model untuk menggambarkan, membangun dan mendokumentasikan sistem yang dirancang.

Jenis diagram yang digunakan antara lain [9]:

a. Diagram Use-Case.

   Bersifat statis. Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi. Menggambarkan sejumlah eksternal actor dan hubungannya ke use case yang diberikan oleh system.

b. Diagram interaksi dan Sequence (urutan)

   Bersifat dinamis. Diagram urutan adalah diagram interaksi yang menekankan pada pengiriman pesan dalam suatu waktu tertentu. Menggambarkan kolaborasi dinamis antara sejumlah objek dan untuk menunjukan rangkaian pesan yang dikirim antara objek juga

interaksi, sesuatu yang terjadi pada titik tertentu dalam eksekusi sistem.

c. Diagram Aktivitas (Activity Diagram)

Bersifat dinamis. Diagram aktivitas adalah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dari suatu sistem. Diagram ini terutama penting dalam pemodelan fungsi-fungsi suatu sistem dan memberi tekanan pada aliran kendali antar objek sehingga dapat juga digunakan untuk aktivitas lainya seperti Use Case atau interaksi.

d. Diagram Kelas (Class Diagram)

Bersifat Statis. Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi. Diagram ini umum dijumpai pada pemodelan sistem berorientasi objek. Meskipun bersifat statis, sering pula diagram kelas menurut kelas – kelas aktif. Menggambarkan struktur dan deskripsi class, package (paket) dan objek beserta hubungan satu sama lain seperti containment (penahanan), pewarisan, asosiasi dan lain-lain.

2.7 Adobe Flash Pro CS5.5

Adobe Flash Profesional CS 5.5 merupakan salah satu program animasi 2D vektor yang sangat digemari oleh para animator. Dalam perkembangannya, program adobe flash pro melakukan banyak penyempurnaan pada setiap versinya. Program ini dapat menampilkan beberapa fitur baru yang membuat adobe flash semakin canggih untuk urusan animasi 2D berbasis vektor.

Berbagai aplikasi animasi 2D mulai dari animasi kartun, animasi interaktif, game, company profile, presentasi, video clip, animasi web dan aplikasi animasi lainnya sesuai kebutuhan dapat diolah. Dan kini flash didukung dengan kemampuannya dalam embuat animasi transformasi 3D, serta panel yang khusus untuk membuat animasi dengan ActionScript yang lebih mudah pemakaiannya [11].

2.8 Adobe Illustrator CS5.1

Adobe Illustrator merupakan aplikasi pengolah vektor yang handal dalam menciptakan bentuk-bentuk geometri dasar, seperti titik, garis, kurva dan juga polygon dapat di kreasikan secara menarik. Fungsi utama dari aplikasi ini adalah sebagai pengolah *vector* namun tidak hanya itu, aplikasi ini juga dapat menggambar kartun dan juga membuat efek-efek untuk sebuah gambar [12].

2.9 Adobe Photoshop

Photoshop adalah sebuah software photo editing dan desain grafis yang dibuat oleh *Adobe* yang sudah diakui kehandalannya, karena fasilitas yang diberikan di dalamnya tergolong lengkap dari mulai fitur mengedit foto, memanipulasi foto, serta menciptakan kreasi – kreasi grafis yang menarik [13].

2.10 Pengujian Perangkat Lunak

Pengujian perangkat lunak adalah jaminan kualitas dari elemen kritis perangkat lunak dan merepresentasikan review akhir dari spesifikasi, desain, dan pembuatan kode [19]. Pengujian perangkat lunak juga memberikan pandangan mengenai perangkat lunak secara obyektif dan independen, yang bermanfaat pada operasional bisnis untuk memahami tingkat risiko pada implementasinya. Teknik-teknik pengujian mencakup, tetapi tidak terbatas pada proses mengeksekusi suatu bagian program atau keseluruhan aplikasi dengan tujuan untuk menemukan kesalahan perangkat lunak. Pada umumnya terdapat 2 pengujian yaitu :

1. **Pengujian Alpha**

   Pengujian alpha dilakukan pada sisi pengembang yang dianggap pemakai dan merekam semua kesalahan dan masalah pemakaian. Pengujian alpha dilakukan pada sebuah lingkungan yang terkendali. Tahap pertama memulai pengujian produk. Pada fase ini produk masih dalam tahap pengujian dan belum sampai pada publik.

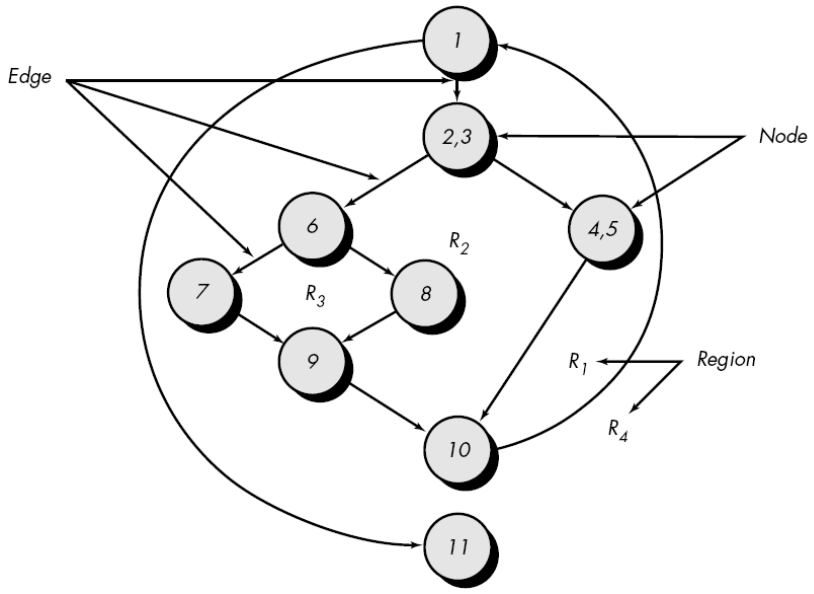
   Terdapat 2 macam pendekatan pengujian, yaitu sebagai berikut:

a. White Box Testing

Pengujian white box, kadang disebut juga pengujian glass box, merupakan metode desain uji kasus yang menggunakan struktur kontrol dari desain prosedural untuk memperoleh uji kasus. Dengan menggunakan metode pengujian white box, perekayasa sistem dapat melakukan test case untuk memberikan jaminan bahwa :

1. Semua jalur independen pada suatu modul ditelusuri minimal 1 kali
2. Semua jalur keputusan logis True/False dilalui
3. Semua loop dieksekusi pada batas yang tercantum dan batas operasionalnya
4. Struktur data internal digunakan agar validitas terjamin

Pengujian white box dapat dilakukan dengan pengujian *basis path*, metode ini merupakan salah satu teknik pengujian struktur kontrol untuk menjamin semua statemen dalam setiap jalur independen program dieksekusi minimal 1 kali. Perhitungan jalur independen dapat dilakukan melalui metrik *Cyclomatic Complexity*. Sebelum menghitung nilai *Cyclomatic Complexity*, terlebih dahulu diterjemahkan desain prosuderal ke bagan alir (*flow chart*), kemudian dibuat *flow graph*.



Gambar 2. 1 Contoh Flow Graph

Cyclomatic complexity digunakan untuk mencari jumlah path dalam satu flow graph. Dapat dipergunakan rumusan sebagai berikut :

1. Rumus Cyclomatix complexity V(G) untuk grafik alir dihitung dengan cara:

$$\mathbf{V(G) = E - N + 2} \qquad \mathbf{(2.2)}$$

Dimana:

E = jumlah edge pada grafik alir

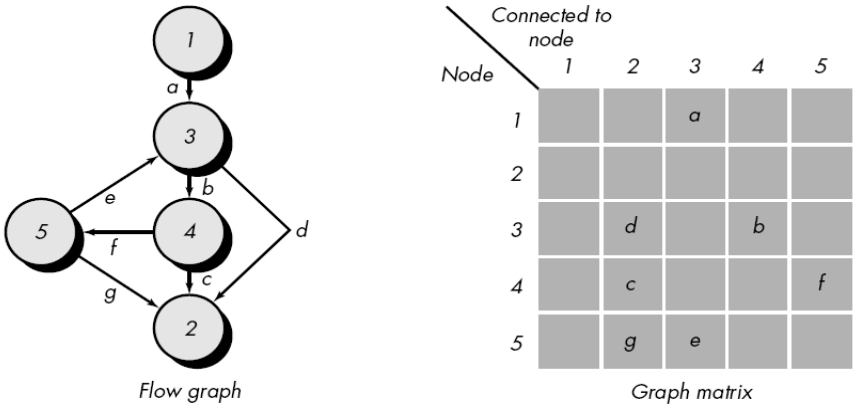
N = jumlah node pada grafik alir

2. Rumus Cyclomatix complexity V(G) juga dapat dihitung dengan cara:

$$\mathbf{V(G) = P + 1} \qquad \mathbf{(2.3)}$$

Dimana P = jumlah predicate node pada grafik alir

Untuk mengembangkan perangkat lunak yang membantu dalam pengujian basis path, *graph matrix* dapat sangat berguna. Graph matrix adalah matriks persegi yang terdiri dari jumlah baris dan kolom yang sama dengan jumlah node pada flow graph. Setiap baris dan kolom sesuai dengan identifikasi simpul, dan entri matriks sesuai dengan koneksi tepi antara node.



Gambar 2. 2 Contoh Graph Matrix

b. Black Box Testing

Pengujian black box, juga disebut pengujian perilaku, berfokus pada persyaratan fungsional perangkat lunak. Pengujian black-box memungkinkan perekayasa perangkat lunak untuk mendapatkan set kondisi input yang akan sepenuhnya melaksanakan semua persyaratan fungsional untuk suatu program. Pengujian black-box bukan merupakan alternatif untuk teknik white-box. Sebaliknya, merupakan pendekatan komplementer yang kemungkinan akan mengungkap kelas yang berbeda dari kesalahan daripada metode white box. Pengujian pada Black Box berusaha menemukan kesalahan seperti:

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

2. Pengujian Beta

Pengujian beta digunakan untuk preview dalam sebuah produk perangkat lunak. pengujian beta hampir sama dengan pengujian alpha, namun pada tahap ini produk sudah didistribusikan ke publik. Pengujian beta juga bermanfaat Untuk mengetahui tanggapan dan penilaian pengguna atau publik terhadap produk perangkat lunak. Untuk mendapatkan tanggapan dari publik dapat menggunakan kuisoner, angket, poling. Setelah sampel terkumpul maka dilakukan pengukuran menggunakan skala pengukuran [20].

1. Skala *Linkert*

Skala likert digunakan untuk mengukur sikap, pendapat dan persepsi seseorang atau sekelompok orang tentang kejadian atau gejala sosial.

Dengan menggunakan skala likert, maka variabel dijabarkan menurut urutan variabel, sub variabel, indikator, deskriptor. Dan

deskriptor ini dapat dijadikan titik tolak untuk membuat butir instrumen berupa pernyataan atau pertanyaan yang perlu dijawab oleh responden.

Item item likert menyediakan repon dengan kategori yang berjenjang. Biasanya banyaknya jenjang adalah lima, yaitu sangat setuju, setuju, ragu ragu, tidak setuju dan sangat tidak setuju. Setiap kategori respon selanjutnya diberi skor. Untuk item positif skor terbesarnya adalah lima, skor terendah adalah satu dan sebaliknya jika item negatif.

Untuk menghitung total skor tiap responden adalah dengan cara menjumlahkan skor skor item yang diperoleh reponden. Oleh karena itu, prosedur penskalaan likert sering disebut sebagai *Likert's Summeted Rating.*

Langkah langkah menyusunnya adalah sebagai berikut:

a. Tentukan secara tegas sikap terhadap topik apa yang akan diukur.
b. Tentukan secara tegas sub varibel / dimensi yang menyusun sikap tersebut, kognitif, afektif dan konatif.
c. Susun pernyataan / pertanyaan atau item yang merupakan alat pengukur dimensi yang menyusun sikap yang akan diukur sesuai dengan indikator.
d. Setiap item diberi respon yang sifatnya tertutup.
e. Untuk setiap respon, jawaban diberi skor berdasarkan kriteria sebagai berikut: apabila item positif maka angka terbesar diletakkan pada respon sangat setuju, sedangkan bila item negatif maka angka terbesar diletakkan pada respon sangat tidak setuju.
f. Untuk mengetahui posisi setiap responden tentang suatu variabel, tentukan skor maksimal dan skor minimal yang mungkin dicapai oleh responden.

Langkah langkah untuk mengetahui bagaimana sikap tiap responden terhadap kualitas produk adalah :

a. Menentukan skor maksimal, yaitu skor jawaban terbesar dikali banyak item.
b. Menentukan skor minimal, yaitu skor jawaban terkecil dikali banyak item.
c. Menentukan nilai median, yaitu hasil penjumlahan skor maksimal dengan skor minimal dibagi dua.
d. Menentukan nilai kuartil 1, yaitu hasil penjumlahan skor minimal dengan median dibagi dua.
e. Menentukan nilai kuartil 3, yaitu hasil penjumlahan skor maksimal dengan median dibagi dua.
f. Membuat skala yang menggambarkan skor minimal, nilai kuartil 1, median, nilai kuartil 3 dan skor maksimal.
g. Mencari batas batas skor untuk masing masing kategori sikap.
h. Membuat tabel distribusi frekuensi sikap tiap responden terhadap kualitas produk.

Langkah langkah untuk mengetahui bagaimana sikap responden secara keseluruhan terhadap kualitas produk adalah:

a. menentukan skor maksimal, yaitu skor maksimal yang diperoleh tiap responden dikali banyaknya responden.
b. Menentukan skor minimal, yaitu skor minimal yang diperoleh tiap responden dikali banyaknya responden.
c. Menentukan nilai median, yaitu hasil penjumlahan total skor maksimal dengan total skor minimal dibagi dua.
d. Menentukan nilai kuartil 1, yaitu hasil penjumlahan total skor minimal dengan median dibagi dua.
e. Menentukan nilai kiartil 3, yaitu hasil penjumlahan total skor maksimal dengan median dibagi dua.
f. Membuat skala yang menggambarkan total skor minimal, nilai kuartil 1, median, kuartil 3 dan total skor maksimal.
g. Mencari batas batas skor untuk masing masing kategori sikap.

h. Menentukan skor total yang diperoleh seluruh responden.

i. Mengintrepretasikan skor total responden dengan skala pada point.