# BAB 3
# ANALISIS DAN PERANCANGAN SISTEM

## 3.1. Analisis Sistem

Pada bagian analisis sistem merupakan penelitian atas sistem yang telah ada dengan tujuan untuk merancang sistem yang baru atau diperbarui. Analisis sistem ini mencakup analisis-analisis terhadap masalah yang akan dilakukan terhadap *game* mulai dari analisis masalah, analisis *game* sejenis, analisis kebutuhan non-fungsional, analisis pengguna, analisis algoritma, analisis fungsional hingga perancangan antarmuka dari *game* yang akan dibuat.

### 3.1.1. Analisis Masalah

Berdasarkan tinjauan analisis sistem yang telah dilakukan dengan melakukan evaluasi terhadap aplikasi *game* yang akan dibangun. Beberapa permasalahan yang merupakan hasil dari analisis yang dilakukan dijadikan sebagai latar belakang dalam pembangunan aplikasi *game Save Our Planet from Extinction.*

### 3.1.2. Analisis *Game* Sejenis

Analisis *game* sejenis dilakukan untuk memberikan gambaran sistem agar lebih mudah dalam pembangunan dan sebagai pembanding untuk kesempurnaan *game* yang akan dicapai.

#### 3.1.2.1. Tomb Raider Chronicle

Tomb Raider Chronicle adalah sebuah *video game* ber-genre *third person shooter* yang dikembangkan oleh Core Desain dan diterbitkan oleh Eidos Interactive. Ini merupakan seri kelima dalam seri Tomb Raider. Permainan ini awalnya dirilis pada tahun 2000 untuk PlayStation, Dreamcast, Microsoft Windows dan Mac OS. Ini juga dirilis pada PAL Play Station Store pada 12 Januari 2011 terjual 1,5 juta kopi di seluruh dunia, membuat permainan Tomb Raider kedua paling sukses dalam seri, setelah Tomb Raider : Anniversary. Di dalam *game* ini pemain disuguhkan dengan petualangan yang panjang dan ada

empat *level* yang harus dilalui, misi utama dalam permainan ini adalah menemukan beberapa artefak mistis yang tersembunyi di pulau misterius.



**Gambar 3.1 Screenshot Tomb Raider Chronicle**

*Game* memiliki grafis dan alur cerita yang cukup baik, namun di dalam *game* masih terdapat kekurangannya yaitu instruksi yang diberikan masih sangat kurang jelas sehingga dapat membingungkan dalam mencari jalan keluar selain dari itu karakter laracroft memakai pakaian yang minim hal ini dapat memberikan efek *negatif* jika dimainkan oleh anak-anak yang belum cukup umur.

**Tabel 3.1 Analisis *Game* Sejenis**

| Nama *Game* | *Genre Game* | Desain Grafis | *Storyline* dan *gameplay* | *multiplayer* | Basis |
|---|---|---|---|---|---|
| Tomb Raider Chronicle | *Third Person Shooter* | Memiliki gambar 3D yang cukup baik serta | Alur cerita bersifat fiktif bukan diambil dari | tidak | *Desktop* |

| | | | | | |
|---|---|---|---|---|---|
| | | tidak memiliki spesifikasi *hardware* yang tinggi | hal-hal yang pernah terjadi, terdapat banyak teka-teki yang harus dipecahkan di dalam *game* tersebut | | |

**a. *Gameplay***

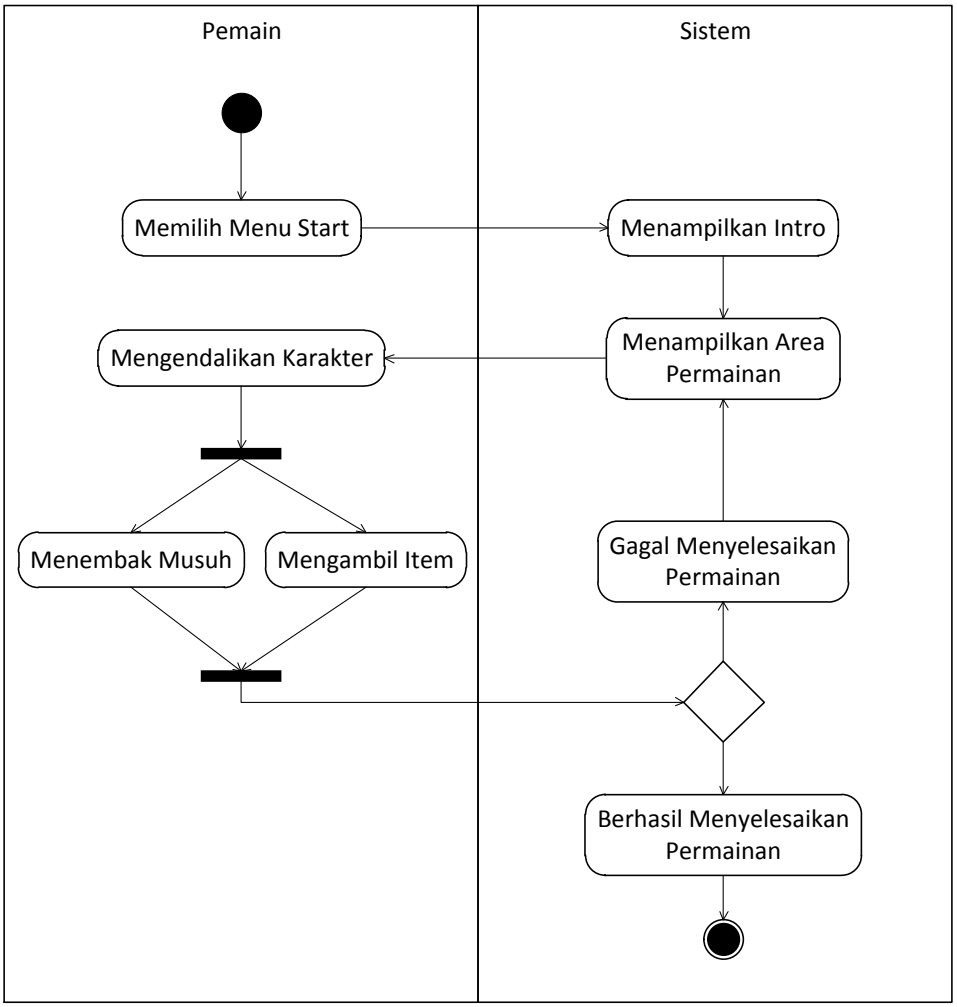
Pada *game Tomb Raider Chronicle,* pemain harus mengalahkan musuh-musuh yang ada di dalam *game* dan harus sampai ke titik yang telah ditentukan untuk menuju *level* selanjutnya.

**b. Skenario *Game***

Skenario dalam permainan *Tomb Raider Chronicle* dapat dilihat alur permainannya sebagai berikut :

a. Pemain berada di sebuah kota di roma.
b. Pemain harus mencari sebuah titik yang sudah ditentukan untuk menuju *level* selanjutnya
c. Pemain harus menghindari dan membasmi musuh ketika berada di dalam arena permainan
d. Jika pemain terkena serangan dari musuh, maka *healthpoint* akan berkurang dan apabila habis maka permainan diulangi ke awal permainan.

   Alur sistem dari *game Tomb Raider* ini dapat digambarkan melalui *activity diagram* yang terlihat pada gambar 3.2.




**Gambar 3.2 Alur permainan Pada *Game* Tomb Raider**

*Game Tomb Raider* memiliki beberapa keterbatasan yaitu :

1. *Game* ini memiliki banyak teka-teki yang harus dipecahkan di setiap *level*-nya.
2. *Game* ini hanya bisa dimainkan oleh satu pemain saja.
3. Instruksi yang diberikan ketika menerima misi kurang detail.

Disamping ada beberapa keterbatasan yang dimiliki *game Tomb Raider* ada juga beberapa kelebihan yang dimiliki yaitu :

1. Konten yang disajikan cukup lengkap di dalam *game* ini
2. Memiliki grafis yang cukup baik
3. Memiliki banyak senjata
4. Berbagai rintangan ada di *game* ini

### 3.1.3. Analisis *game* yang dibangun

*Game* yang akan dibangun berjudul *Save Our Planet From Extinction*. *Game* ini bergenre *third person shooter*, *Game* ini memiliki grafis 3D (3 Dimensi). Memiliki alur cerita tentang seorang polisi hutan yang berusaha membantu para warga desa untuk melumpuhkan penebang liar yang telah merusak hutan yang ada disekitar perkampungan.

*Game* dibangun dengan konsep sudut pandang orang ketiga agar permainan dapat dimainkan secara leluasa dalam menentukan tujuan.

Berikut ini adalah fitur-fitur yang terdapat di dalam *game* :

1. Grafik *game* 3D
2. *Game* ini memiliki 5 buah *level*
3. *Game* ini dibuat dengan *genre action third person shooter*
4. *Game* ini mempunyai menu *pause* sehingga dapat dihentikan ketika permainan sedang berlangsung
5. Memiliki nilai-nilai kebudayaan seperti rumah tradisional yang ada di Kalimantan yaitu rumah panggung dimana info dari objek rumah tersebut bisa ditampilkan pada saat permainan berlangsung selain itu warga yang ada di desa merupakan warga dayak yang ada di Kalimantan sehingga dapat mengenal lebih jauh tentang kebudayaan tersebut.
6. *Game* dilengkapi dengan *menu option* untuk pengaturan suara dan resolusi gambar.

### 3.1.4. *Storyline*

Jalan cerita dari *game save our planet from extinction* ini berupa fiktif belaka dengan melatarbelakangi masalah tentang kerusakan hutan yang ada di daerah Kalimantan Indonesia. Cerita berawal ketika seorang polisi yang bernama Arman ditugaskan untuk pergi ke suatu perkampungan untuk menelusuri keadaan hutan yang ada disekitar desa tersebut dikarenakan banyaknya laporan-laporan warga setempat akan adanya penebangan liar yang ada di desa. Keadaan lingkungan yang ada di desa tersebut sangatlah berantakan banyak pohon-pohon tumbang dikarenakan aliran sungai yang membawanya. Lama kelamaan hutan

yang ada disekitar kampung tersebut pun mulai gundul tidak menutup kemungkinan bencana alam akan sering terjadi dari mulai banjir ataupun bencana lainnya. Banyaknya laporan dari warga desa tentang kerusakan hutan membuat Arman perlu menyelidiki ke daerah tersebut, hal ini membuat Arman merasa kasihan terhadap warga di desa tersebut dan berniat untuk membantu warga desa dalam memerangi penebangan liar.

Ketika Arman tiba di desa, Arman disambut oleh para warga desa dan dituntut untuk menemui kepala desa untuk mengetahui jalan menuju hutan. Kepala desa menuturkan kepada arman secara rinci keluh kesahnya yang mewakili para warga tentang masalah-masalah lingkungan yang ada di desa. Arman berinisiatif akan membantu menyelesaikan masalah-masalah lingkungan yang ada untuk segera diselesaikan. Berikut *level-level* yang harus dilalui :

1. *Level* Pertama, Pemain harus menemui kepala desa untuk mengetahui masalah yang dihadapi desa tersebut, selanjutnya misi yang diberikan pemain harus mencari portal untuk menuju ke *level* berikutnya.
2. *Level* kedua, setelah keluar dari desa pemain tiba di goa, misi utama yang harus dilakukan di *level* ini yaitu menemukan portal keluar dari goa, dengan melumpuhkan semua anjing-anjing liar yang ada agar dapat melaju ke *level* berikutnya.
3. *Level* ketiga, misi utama yang harus dilakukan di *level* ini yaitu menemukan portal keluar dari goa, disini pemain harus melewati jalan yang bercabang dan rintangan yang harus dilalui dengan mengikuti petunjuk arah dalam goa tersebut.
4. *Level* keempat, Setelah keluar dari goa, pemain tiba dihutan. Misi utama yang harus dilakukan yaitu melumpuhkan semua musuh-musuh yang ada di hutan tersebut dan tetap waspada, karena musuh dalam *level* ini membawa senjata api.
5. *Level* kelima, ini merupakan *level* terakhir dari *game* ini, misi utama yang harus dilakukan yaitu mengambil semua kayu-kayu hasil dari penebangan liar yang ada di gudang. Dalam hal ini pemain harus mencari dimana letak

kayu tersebut dengan melihat radar, jika pemain berhasil dalam melakukan misi tersebut maka pemain telah menyelesaikan semua misi yang ada.

### 3.1.5. *Gameplay*

Misi utama dari *game* ini adalah mengusir semua penebang liar yang ada di hutan. Menu utama memiliki beberapa menu yaitu Main, Pengaturan, Petunjuk dan Keluar. Di dalam *game* ini terdapat lima buah *level* dimana setiap *level* mempunyai latar tempat yang berbeda. Setiap *level* mempunyai tingkatan kesulitan tersendiri. Pada setiap *level, player* akan mendapatkan beberapa *item* yaitu berupa *box* darah dan *box* amunisi agar mempermudah dalam menyelesaikan misi disetiap *level-nya*. Dalam *game* ini disisipkan beberapa unsur edukasi tentang kebudayaan suku dayak yang ada di Kalimantan, hal ini muncul di awal *level,* pemain akan menjumpai bangunan adat dan senjata tradisional, informasi tersebut akan tampil jika pemain mendekati objek tertentu. Senjata yang digunakan di dalam *game* ini berupa senapan untuk melumpuhkan musuh-musuh yang ada disetiap *level*-nya.

### 3.1.6. Analisis Pelevelan

Pada *game Save Our Planet from Extinction* memiliki 5 tingkatan *level* yaitu *level* mudah, sedang dan sulit, berikut ini penjelasan tingkatan *level* dapat dilihat pada tabel 3.2

**Tabel 3.2 Penjelasan Tingkatan *Level* Pada *Game***

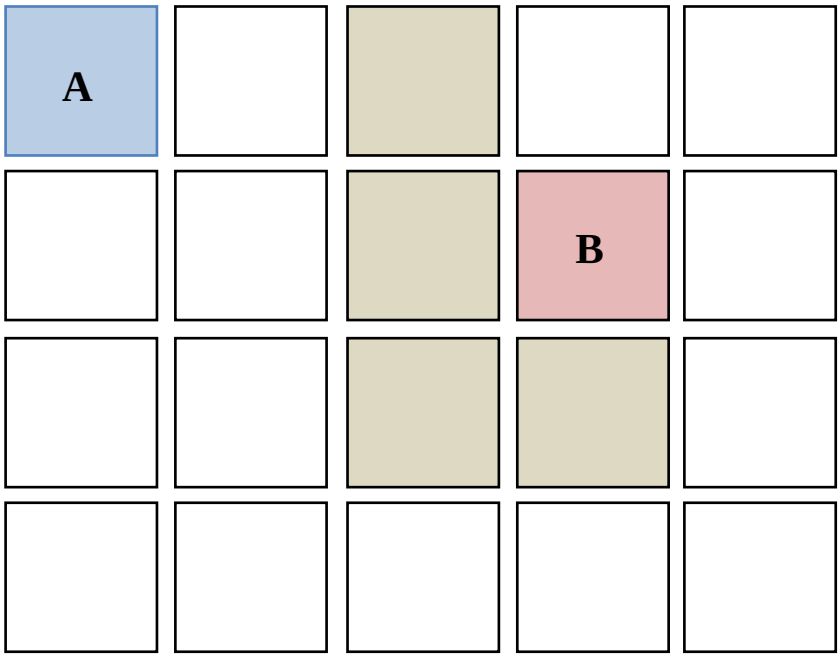
| Tingkatan *level* | Deksripsi |
|---|---|
| Mudah | Pada permainan *save our planet from extinction* dengan *level* mudah terdapat pada *level* 1 dan 5, Map yang tidak rumit dan misi yang harus dilakukan hanya mengumpulkan *item* sehingga tidak memerlukan kegesitan dalam bermain. |
| Sedang | Pada permainan *save our planet from extinction* dengan *level* sedang terdapat pada *level* 2 dan 3, *map* cukup rumit dan menantang dan musuh |

| | |
|---|---|
| | yang ditempatkan di *level* 2 tidak terlalu banyak. Disini pemain memerlukan konsentrasi untuk melewati rintangan yang ada. |
| Sulit | Pada permainan *save our planet from extinction* dengan *level* sulit terdapat pada *level* 4. Disini pemain dituntut untuk lebih berkonsentrasi dan memerlukan kegesitan bermain karena *map* sangat rumit dan musuh yang ditempatkan ada banyak, dengan *damage* yang diberikan musuh lebih besar daripada di *level* 2 |

Tujuan pelevelan pada permainan *save our planet from extinction* yaitu untuk mengembangkan permainan agar tidak membosankan, pada dasarnya dalam sebuah *game* ada tingkat kesulitan yang dapat berubah-ubah dimana tingkat kesulitan permainan didasarkan pada aksi-aksi pemain dengan memberikan banyak pilihan aktifitas dengan berbagai tingkat kesulitan di dalam permainan. Pemain dapat menjalankan aktifitas-aktifitas permainan sesuai alur yang memuaskan selera masing-masing, sehingga dapat mencakup banyak variasi permainan dengan berbagai tingkat kesulitan yang memungkinkan berbagai macam pemain dapat menikmati permainan sesuai dengan kecepatan dan kemampuan masing-masing dan membiarkan pemain membuat pilihan seiring dalam memainkannya.

### 3.1.7. Analisis Algoritma A* (A *Star*)

Algoritma A* di dalam *game* akan diterapkan pada NPC (*Non Playable Character*) musuh sebagai penentuan jalur terpendek atau *pathfinding* menuju *target* yaitu *player*, dalam hal ini node A merupakan NPC dan node B adalah *player*. Cara kerja algoritma A* di dalam *game* lebih jelasnya bisa dilihat pada gambar 3.3




**Gambar 3.3 Tahap Awal Pencarian Algoritma A***

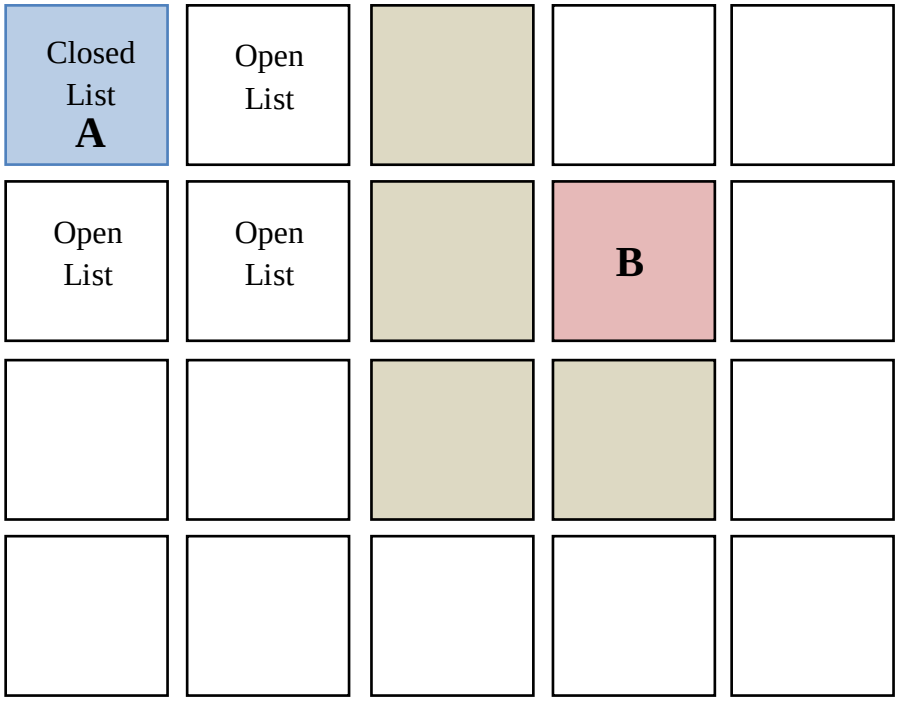
Pada gambar 3.3 pencarian wilayah dibagi ke dalam *node-node* yang terdiri dari node berwarna biru merupakan *node* awal (*node* A), *node* berwarna merah merupakan *node* akhir (*node* B) dan yang berwarna abu-abu merupakan penghalang (*unwalkable node)*.

Selanjutnya adalah, melakukan pencarian untuk menemukan jalan terpendek dengan cara memeriksa *node-node* yang berdekatan dengan *node* awal untuk bisa sampai *node* B. Berikut adalah langkah pencarian yang dilakukan oleh *node* awal:

1. Langkah pertama, dimulai dari *node* awal yang ditambahkan ke dalam *Open list* yang merupakan *list* untuk *node-node* yang harus diperiksa.
2. Periksa semua *node* yang berdekatan dengan *node* awal apakah bisa dilalui atau tidak, *node* penghalang yang berwarna abu-abu diabaikan lalu tambahkan semua *node* tersebut ke dalam *open list*, untuk setiap *node* yang baru ditambahkan, simpan *node* awal sebagai "*parent*" yang nantinya akan digunakan untuk menelusuri jalan.
3. Hapus *node* awal dari *open list*, dan tambahkan ke dalam *closed list. Closed list* merupakan *list* untuk *node* yang tidak perlu diperiksa atau

dalam artian merupakan bagian dari jalur terpendek yang sudah didapatkan.

Untuk lebih jelasnya *open list* dan *closed list* diilustrasikan pada gambar 3.4

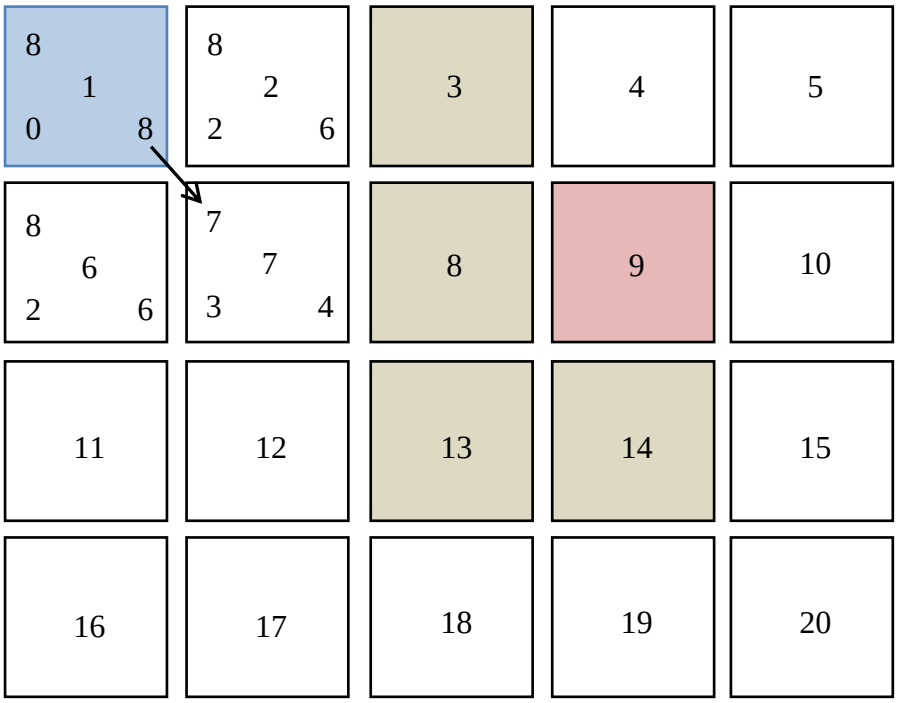



**Gambar 3.4 Ilustrasi O*pen List* dan *Closed List***

Pada gambar 3.4 diatas *node* awal yang berwarna biru sudah ditambahkan ke dalam *closed list.* Sedangkan kotak yang berada di sekelilingnya adalah *open list* yang harus diperiksa satu persatu nilai dari masing-masing *node open list*.

Langkah berikutnya adalah memilih *node* yang memiliki nilai F terendah di dalam *open list* dengan menggunakan rumus (F = G + H). G adalah nilai yang dikeluarkan untuk dari *node* A ke *node* yang ada disekitarnya. Sedangkan H adalah fungsi *heuristic* yang digunakan untuk menghitung perkiraan biaya yang dikeluarkan dari *node* awal menuju *node* tujuan.

Proses ini dilakukan secara berulang-ulang, dimulai dari *open list* dan menemukan *node* dengan nilai F paling rendah. Perhitungan persamaan untuk mendapatkan nilai F dijelaskan pada ilustrasi gambar 3.5.




**Gambar 3.5 Perhitungan menentukan nilai F**

Gambar 3.5 menunjukkan pada *open list* dari *node* awal yaitu NPC nilai F yang dihasilkan adalah 7 yang didapatkan dari menambahkan nilai G = 3 dan nilai H = 2. Di dalam contoh ini G akan diberi nilai 2 apabila *node* bergerak secara *vertical* atau *horizontal* dan diberi nilai 3 apabila *node* bergerak secara *diagonal*.

Perhitungan nilai H yaitu nilai estimasi jalur terpendek dari *node* awal ke *node* tujuan yaitu *player* dengan menggunakan (persamaan 2.1). Sebagai contoh *node* yang terletak disebelah kanan dari *node* awal diberikan nilai H = 6 yang didapatkan dari perhitungan 3 *node* menuju *node* tujuan dan dikalikan 2, 2*3 = 6.

Setelah didapatkan nilai F dari setiap *node* yang ada di *open list*, maka akan dipilih *node* yang memiliki nilai F paling rendah yang kemudian disebut dengan *current node*, langkah selanjutnya adalah sebagai berikut.

1. Hapus *current node* dari *open list* dan masukkan ke dalam *closed list.*
2. Periksa semua *node* yang berdekatan dengan *current node*, abaikan *node* penghalang. Jika tidak ada pada *open list,* tambahkan ke dalam *open list*, lalu tambahkan "*parent*" untuk *node* tersebut.

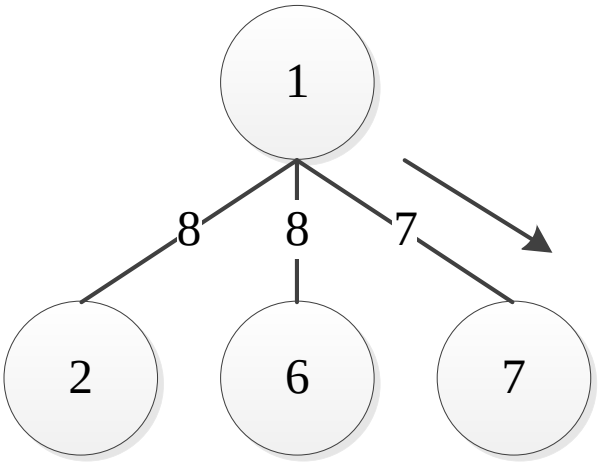
3. Jika *node* sudah ada pada *open list,* lalu bandingkan dengan *node* awal apakah jalur yang di ambil lebih baik dari *node* awal, dengan dengan memeriksa nilai G-nya apakah lebih rendah jika dalam menggunakan *node* ini, jika lebih rendah maka hitung ulang nilai F dan G serta merubah arah *pointer,* jika tidak maka jangan lakukan apa-apa.

Langkah pertama menghasilkan *node-node* di dalam *open list* dan *closed list* dijelaskan pada tabel 3.3

**Tabel 3.3 Perhitungan Langkah Pertama**

| | |
|---|---|
| *Open List* | 2, 6, 7 |
| *Closed List* | 1 |
| *Parent* | 1 |
| *Current Node* | 7 |

Gambar 3.6 merupakan penjelasan langkah pertama pencarian jalur dari algoritma *astar.* Dibawah ini adalah penjelasan langkah pertama algoritma A *star* dengan menggunakan *tree.*

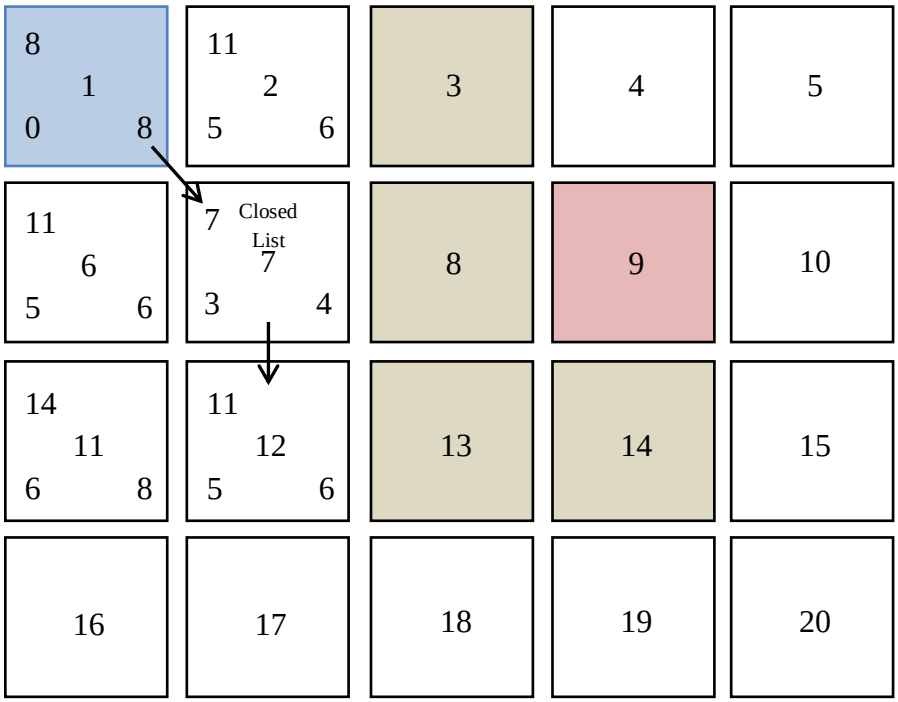



**Gambar 3.6 Penjelasan A* Menggunakan T*ree* Pada Langkah Pertama**

Proses yang dijelaskan di atas, dilakukan berulang kali atau *looping,* dimulai dari *open list,* dimana *open list* juga berperan sebagai *parent* dan menemukan *node* dengan nilai F yang paling rendah yang selanjutnya akan dijadikan *current.* Proses yang dijelaskan di atas, dilakukan berulang kali atau *looping*, dimulai dari *open list,* dimana *open list* juga berperan sebagai *parent* dan menemukan *node* dengan nilai F yang paling rendah yang selanjutnya akan dijadikan *current node*, langkah-langkahnya sebagai berikut.

1. Hapus *current node* dari *open list* atau *parent* lalu masukkan *current node* tersebut ke dalam *closed list.*
2. Periksa semua *node* yang berdekatan dengan *current node*, abaikan penghalang atau *unwalkable node*. Jika tidak ada pada *open list,* tambahkan ke dalam *open list* dan tambahkan *parent* untuk node tersebut
3. Jika *node* sudah ada pada *open list,* periksa *open list* yang mana yang memiliki nilai F terendah dengan rumus F = G + H, node dengan nilai F terendah akan dijadikan *current node* selanjutnya.

   Perhitungan pada langkah kedua yang berada pada *node* dengan koordinat (7) dapat dilihat pada gambar 3.7



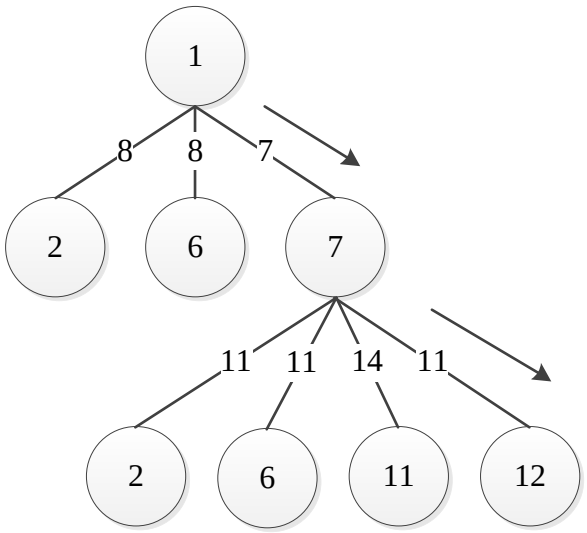

**Gambar 3.7 Penjelasan A* Pada Langkah Kedua**

Tabel 3.7 menunjukan penjelasan tentang pencarian jalur terpendek untuk langkah kedua dapat dilihat pada tabel 3.4

**Tabel 3.4 Perhitungan Langkah Kedua**

| | |
|---|---|
| *Open List* | 2, 6, 11, 12 |
| *Closed List* | 1, 3, 7, 8, 13 |
| *Parent* | 7 |
| *Current Node* | 12 |

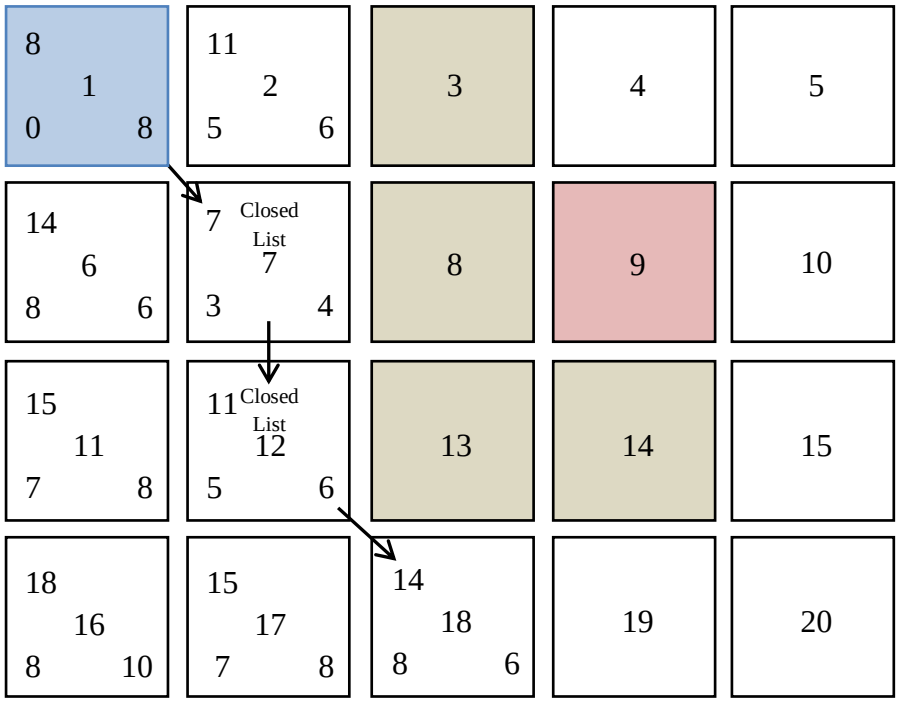
Gambar 3.8 menunjukkan penjelasan melalui *tree* untuk pencarian jalan terpendek pada langkah kedua.




**Gambar 3.8 Penjelasan A* Dengan *Tree* Pada Langkah Kedua**

Pada langkah kedua terdapat node yang memiliki nilai F yang sama pada kasus seperti ini dapat memilih jalur yang mana saja tinggal menyesuaikan dengan node mana yang mendekati node tujuan, pada langkah kedua ini jalur yang dipilih ialah indeks nomor 7.

Perhitungan langkah ketiga yang berada pada *node* dengan indeks nomor 12 dapat dilihat pada gambar 3.9.



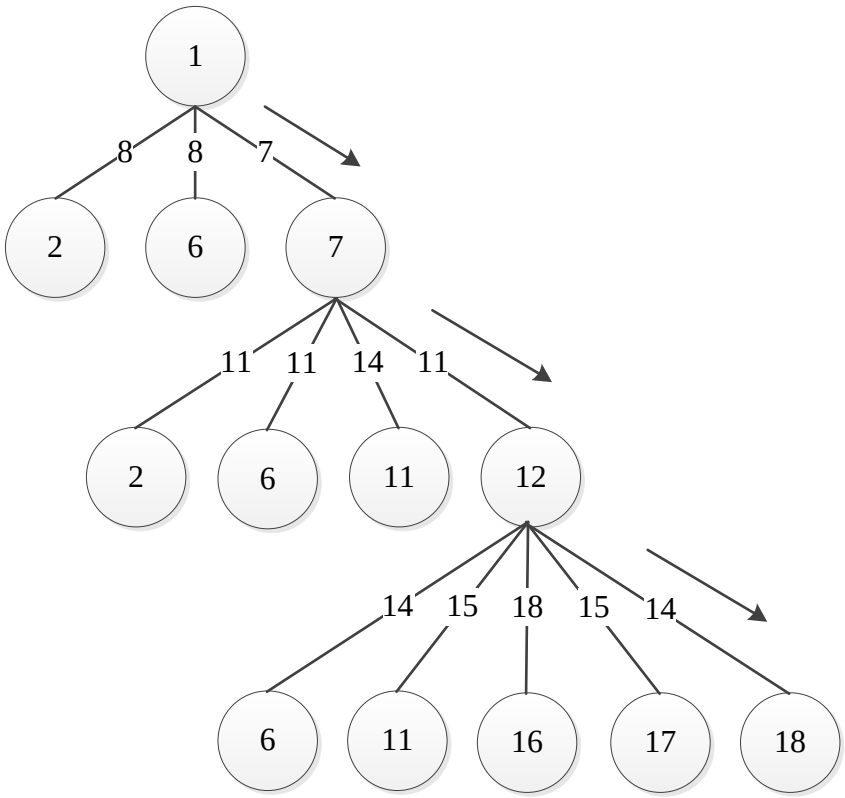

**Gambar 3.9 Penjelasan A* Pada Langkah Ketiga**

Tabel 3.5 menunjukan penjelasan tentang pencarian jalur terpendek untuk langkah ketiga dapat dilihat pada tabel 3.5.

**Tabel 3.5 Perhitungan Langkah Ketiga**

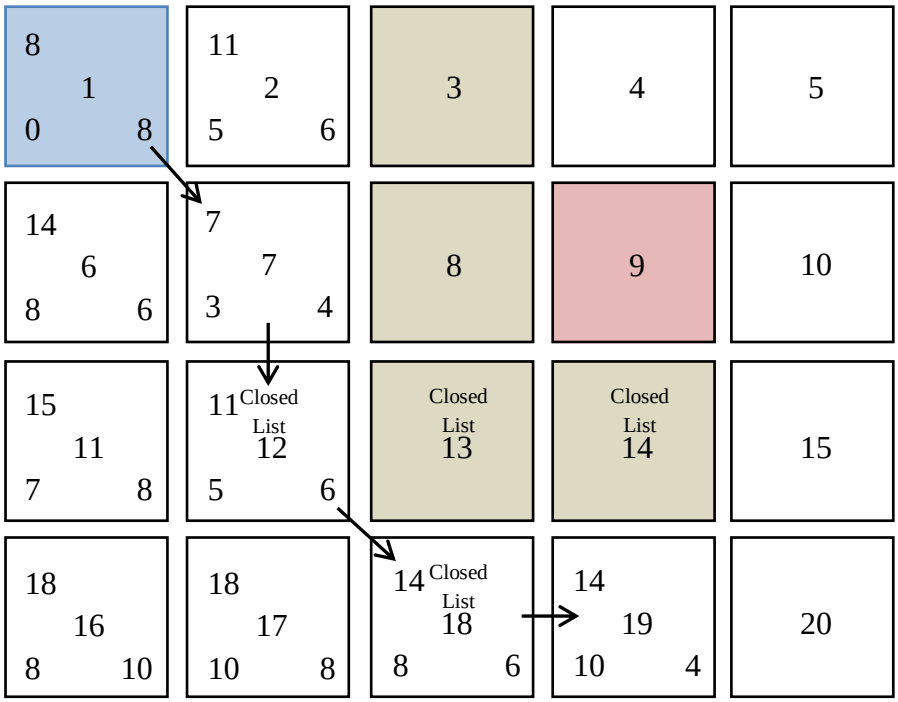
| | |
|---|---|
| *Open List* | 6, 11, 16, 17, 18 |
| *Closed List* | 12, 13, 8, 7 |
| *Parent* | 12 |
| *Current Node* | 18 |

Gambar 3.10 menunjukkan penjelasan melalui *tree* untuk pencarian jalan terpendek pada langkah ketiga.




**Gambar 3.10 Penjelasan A* Dengan *Tree* Pada Langkah Ketiga**

Perhitungan langkah keempat yang berada pada *node* dengan indeks 18 dapat dilihat pada gambar 3.11.



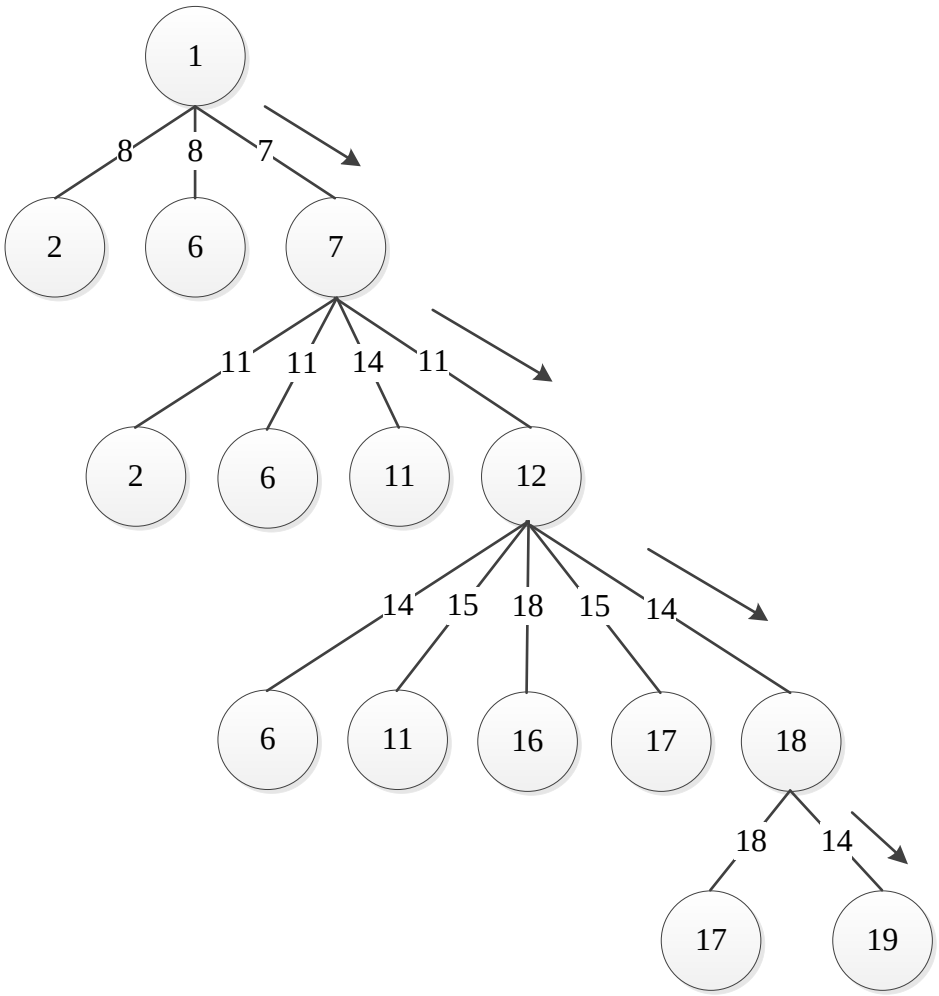

**Gambar 3.11 Penjelasan A* Pada Langkah Keempat**

Tabel 3.6 menunjukan penjelasan tentang pencarian jalur terpendek untuk langkah keempat dapat dilihat pada tabel 3.6.

**Tabel 3.6 Perhitungan Langkah Keempat**

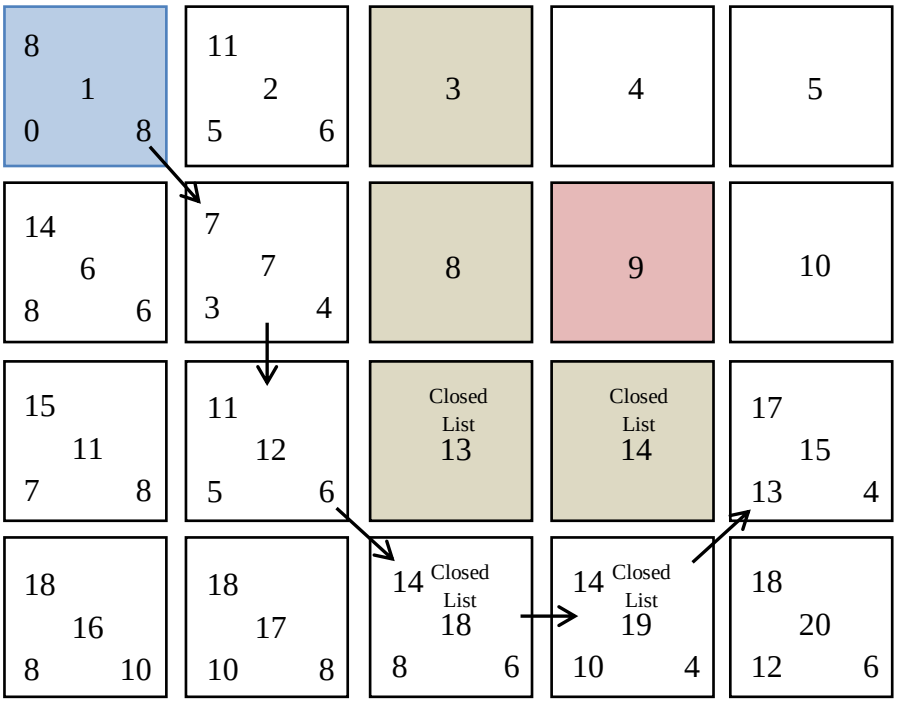
| | |
|---|---|
| *Open List* | 17, 19 |
| *Closed List* | 12, 13, 14, 18 |
| *Parent* | 18 |
| *Current Node* | 19 |

Gambar 3.12 menunjukkan penjelasan melalui *tree* untuk pencarian jalan terpendek pada langkah keempat.




**Gambar 3.12 Penjelasan A* Dengan *Tree* Pada Langkah Keempat**

Perhitungan pada langkah kelima yang berada pada *node* dengan indeks 15 yang memiliki nilai F = 17 bisa dilihat pada gambar 3.13.



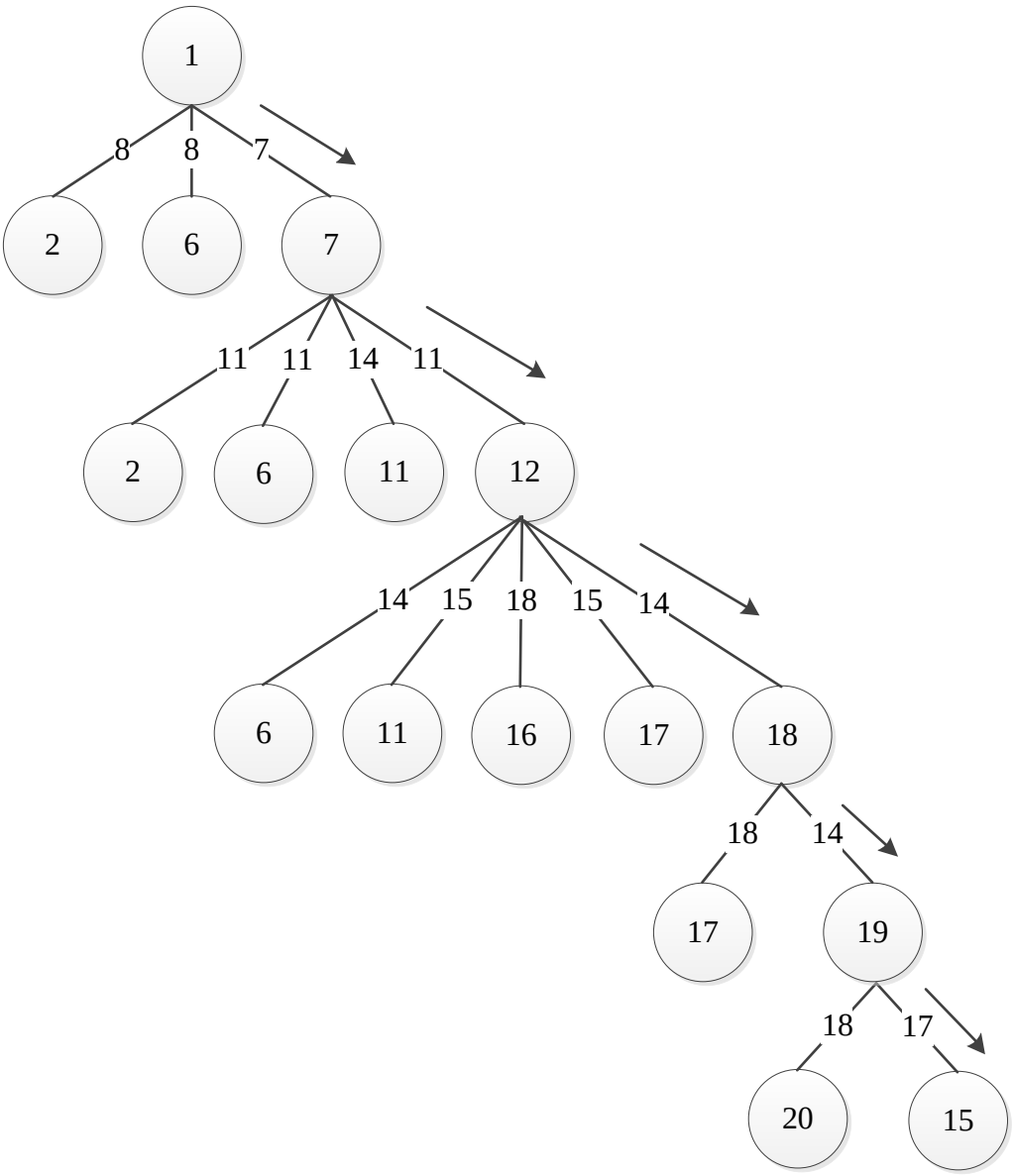

**Gambar 3.13 Penjelasan A* Pada Langkah Kelima**

Pada tabel 3.7 penjelasan melalui tabel untuk pencarian jalur terpendek pada langkah kelima.

**Tabel 3.7 Perhitungan Langkah Kelima**

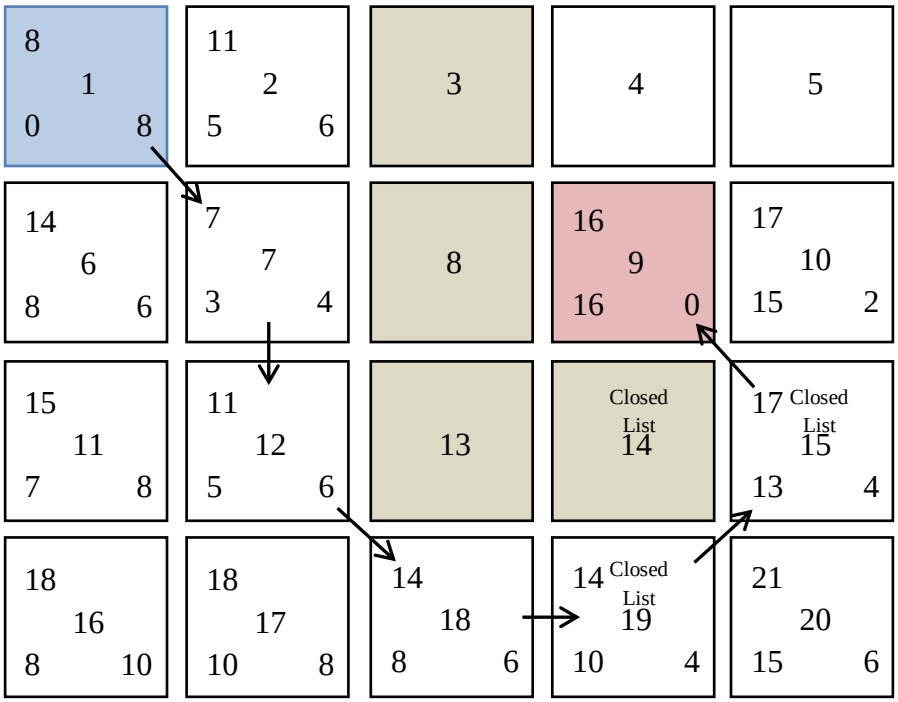
| | |
|---|---|
| *Open List* | 15, 20 |
| *Closed List* | 13, 14, 18, 19 |
| *Parent* | 19 |
| *Current Node* | 15 |

Gambar 3.14 menunjukkan penjelasan melalui *tree* untuk pencarian jalan terpendek pada langkah kelima.




**Gambar 3.14 Penjelasan A* Dengan *Tree* Pada Langkah Kelima**

Perhitungan pada langkah keenam yang berada pada *node* dengan indeks ke 15 yang memiliki nilai F = 17 bisa dilihat pada gambar 3.15.



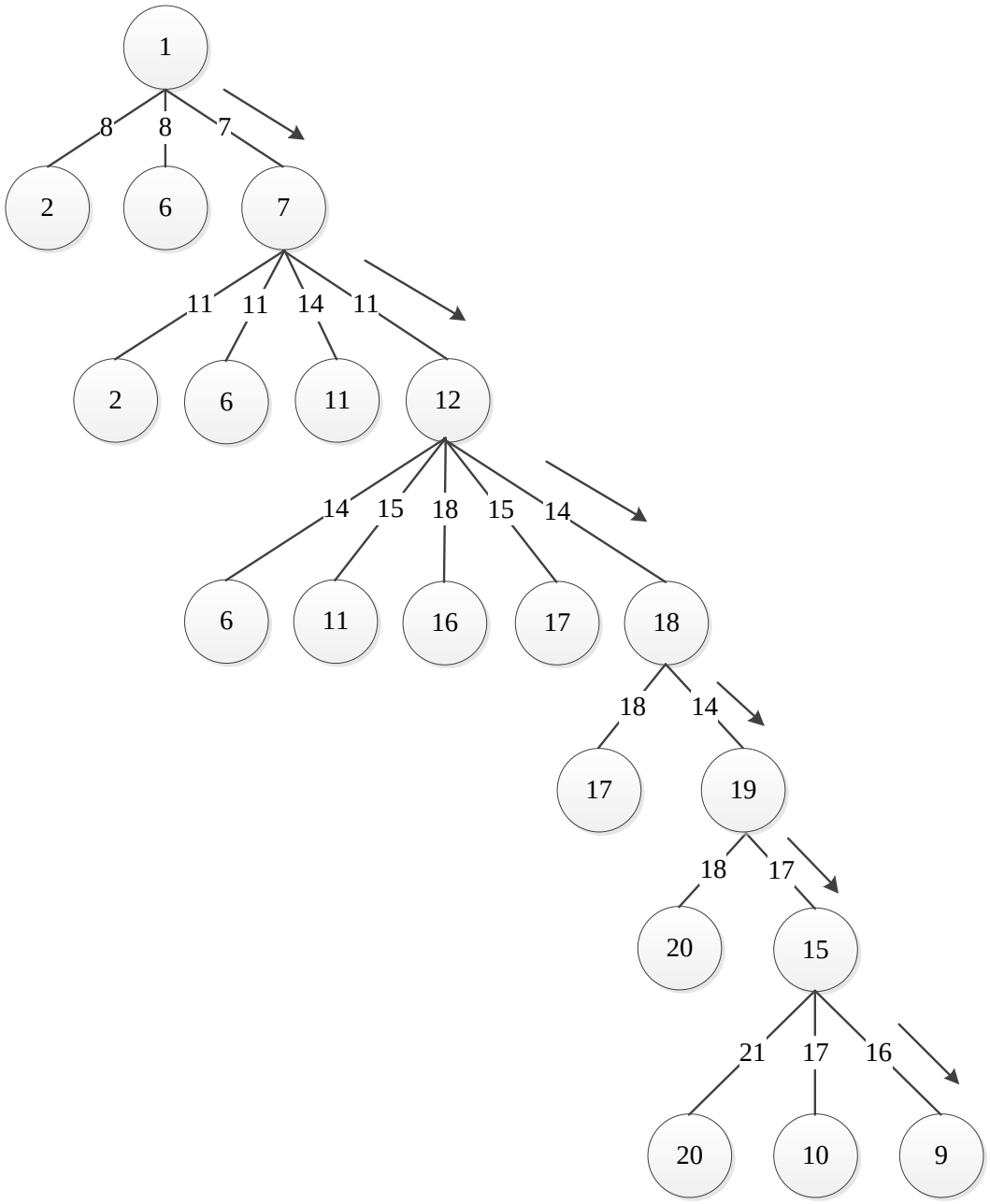

**Gambar 3.15 Penjelasan A* Pada Langkah Keenam**

Pada tabel 3.8 penjelasan melalui tabel untuk pencarian jalur terpendek pada langkah keenam.

**Tabel 3.8 Perhitungan Langkah Keenam**

| | |
|---|---|
| *Open List* | 9, 10, 20 |
| *Closed List* | 14, 15, 19 |
| *Parent* | 15 |
| *Current Node* | 9 |

Gambar 3.16 menunjukkan penjelasan melalui *tree* untuk pencarian jalan terpendek pada langkah keenam.



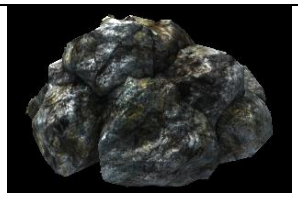
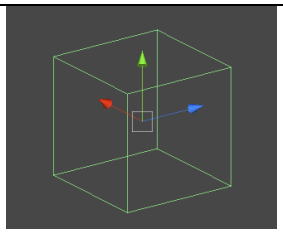
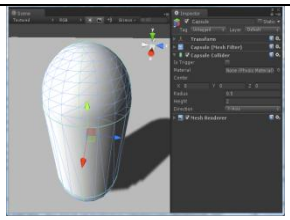
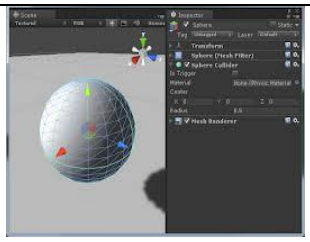

**Gambar 3.16 Penjelasan A* Dengan *Tree* Pada Langkah Keenam**

### 3.1.8. Analisis *Collision Detection*

*Collision detection* adalah pendeteksian objek-objek yang ada di dalam *game* agar apabila terjadi tabrakan dengan objek lain maka tidak akan tembus melewati objek tersebut begitu saja. Ada beberapa *collider* yang digunakan di dalam *game Save Our Planet From Extinction* ini yaitu *terrain collider, mesh collider, box collider* dan *capsule collider*. *Terrain collider* diterapkan pada

*terrain* atau peta yang ada di dalam *game*, *mesh collider* diterapkan pada bentuk bangunan yang tidak beraturan atau simetris seperti rumah-rumah dan pohon-pohon yang terdapat di dalam *game* ini, sementara *box collider* dibuat transparan sebagai batas ruang gerak dari *user* sendiri.

**Tabel 3.9 Macam-macam *Collider* yang Digunakan di Dalam *Game***

| Gambar Collider | Nama Collider |
|---|---|
|  | *Mesh collider*: digunakan untuk bentuk objek tidak beraturan misalnya batu |
|  | *Box Collider*: digunakan pada pembatas transparan yang berguna sebagai batas dari area peta |
|  | *Capsule Collider*: digunakan sebagai collider yang di implementasikan pada *player* |
|  | *Sphere collider* : *collider* ini digunakan pada benda-benda yang berbentuk lingkaran |

### 3.1.9. Sasaran *User* dan Kegunaan

Aplikasi *game* yang akan dibangun ditujukan untuk *user* berusia 15 tahun keatas dan kegunaan dari aplikasi *game* ini yaitu sebagai sarana hiburan serta melatih pola pikir *user* dalam menentukan strategi untuk menyelesaikan permainan serta dengan disisipkannya unsur kebudayaan didalam *game* diharapkan dapat menambah ilmu pengetahuan terhadap kebudayaan Indonesia salah satunya berupa pengetahuan tentang rumah adat.

### 3.1.10. Analisis Kebutuhan *Non*-Fungsional

Analisis *non*-fungsional merupakan analisis yang dibutuhkan untuk menentukan spesifikasi kebutuhan sistem. Spesifikasi ini juga meliputi elemen atau komponen-komponen apa saja yang dibutuhkan untuk sistem yang akan dibangun sampai dengan sistem tersebut diimplementasikan. Analisis kebutuhan ini juga menentukan spesifikasi masukan yang diperlukan sistem, keluaran yang akan dihasilkan sistem dan proses yang dibutuhkan untuk mengolah masukan sehingga menghasilkan suatu keluaran yang diinginkan. Pada analisis kebutuhan sistem non-fungsional ini dijelaskan analisis kebutuhan perangkat lunak, analisis kebutuhan perangkat keras dan analisis pengguna.

#### 3.1.10.1. Analisis Kebutuhan Perangkat Lunak

Perangkat lunak digunakan dalam sebuah sistem merupakan perangkat yang dibutuhkan untuk mengaktifkan dan mengoptimalkan kinerja perangat keras. Perangkat lunak yang dibutuhkan untuk membangun aplikasi *game save our planet from extinction* ini adalah sebagai berikut :

1. Sistem Operasi Windows 7.
2. Unity 4.0.1
3. Autodesk 3ds Max.
4. Adobe Photoshop CS5
5. Blender 2.64

Perangkat lunak yang dibutuhkan pengguna untuk dapat menjalankan aplikasi ini adalah sebagai berikut :

1. Sistem Operasi Windows XP SP2
2. DirectX 9

#### 3.1.10.2. Analisis Kebutuhan Perangkat Keras

Komputer terdiri dari perangkat keras dan perangkat lunak yang saling berinteraksi. Perangkat lunak memberikan instruksi-instruksi kepada perangkat keras untuk melakukan suatu tugas tertentu, sehingga dapat menjalankan suatu sistem di dalamnya. Adapun perangkat keras yang digunakan untuk membangun aplikasi *game* ini adalah sebagai berikut :

Perangkat keras yang digunakan oleh pengembang untuk membangun aplikasi ini adalah sebagai berikut :

1. Processor 3,00 Ghz
2. RAM : 4 GB.
3. VGA : 2 GB.
4. Harddisk 60 GB.
5. Monitor.
6. *Mouse* dan *Keyboard*.
7. Speaker.

Perangkat keras yang direkomendasikan bagi pengguna untuk dapat menjalankan aplikasi ini adalah sebagai berikut :

1. Processor 2,00 Ghz.
2. RAM : 2 GB.
3. VGA : 1 GB
4. Harddisk 10 GB.
5. Monitor.
6. *Mouse* dan *Keyboard*.
7. Speaker

#### 3.1.10.3. Analisis Pengguna Sistem (*User*)

Pada analisis *user* (pemakai) ini akan mencakup analisis beberapa parameter terhadap calon *user* dari aplikasi.

##### 3.1.10.3.1. *User Knowledge and Experience*

*Game Save Our Planet from Extinction* ini bisa digunakan oleh kalangan apapun, tetapi pengetahuan dan pengalaman akan memudahkan *user* dalam penggunaanya. Terutama pengetahuan dan pengalaman dalam memainkan *game third person shooter*. Tabel 3.10 berikut menjelaskan analisis klasifikasi *knowledge and experience* dari pengguna aplikasi.

**Tabel 3.10 Analisis Klasifikasi *Knowledge and Experience***

| **Educational *level*** | **Reading *Level*** | **Typing Skills** |
|---|---|---|
| Bisa digunakan oleh berbagai kalangan, seperti pelajar, mahasiswa hingga masyarakat awam | Bisa digunakan oleh berbagai *level* pendidikan dengan *reading level* yang sedang | *Average* (40 wpm)<br>Tidak memerlukan *typing skills* yang tinggi |
| **Computer Literacy** | **Task Experience** | **System Experience** |
| Moderate (menengah) | Bisa digunakan oleh dengan pengalaman penggunaan komputer dan *game* yang sedang | Bisa digunakan oleh dengan pengalaman penggunaan komputer dan *game* yang sedang |
| **Application Experience** | **Native Language** | **Use Of Other System** |
| Bisa digunakan dalam semua sistem operasi Windows xp/vista/7 | Menggunakan satu bahasa, yakni Indonesia | Bisa di jalankan tanpa perlu menginstal aplikasi lain |

### 3.1.10.3.2. *Users Physical Characteristic*

Keadaan fisik seseorang mungkin akan berpengaruh pada penggunaan aplikasi *game* ini. Ada hal-hal yang harus diperhatikan juga terhadap *user* dari karakteristik fisiknya untuk dapat menggunakan aplikasi ini yaitu, *Color Blind*, *Handednes* dan *Gender* seperti terlihat pada tabel 3.11.

**Tabel 3.11 Analisis *Users Physical Characteristic***

| | |
|---|---|
| Usia | 15 tahun keatas |
| Jenis Kelamin | Pria dan Wanita |
| Tangan | Kanan dan Kiri |
| Buta Warna | *User* yang tidak bisa membedakan warna yang satu dengan yang lainnya (buta warna) masih mampu menggunakan aplikasi ini, karena tidak ada indikator warna-warna khusus yang membedakan antara fungsional yang satu dengan fungsional yang lainnya. Akan tetapi penggunaannya tidak akan optimal karena dalam *game* ini terdapat banyak sekali perbedaan warna yang menunjang interaksi dan ketertarikan dalam permainan. |

### 3.1.11. Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional menggambarkan proses kegiatan yang akan diterapkan dalam sebuah sistem dan menjelaskan kebutuhan yang diperlukan sistem agar sistem dapat berjalan dengan baik sesuai kebutuhan.

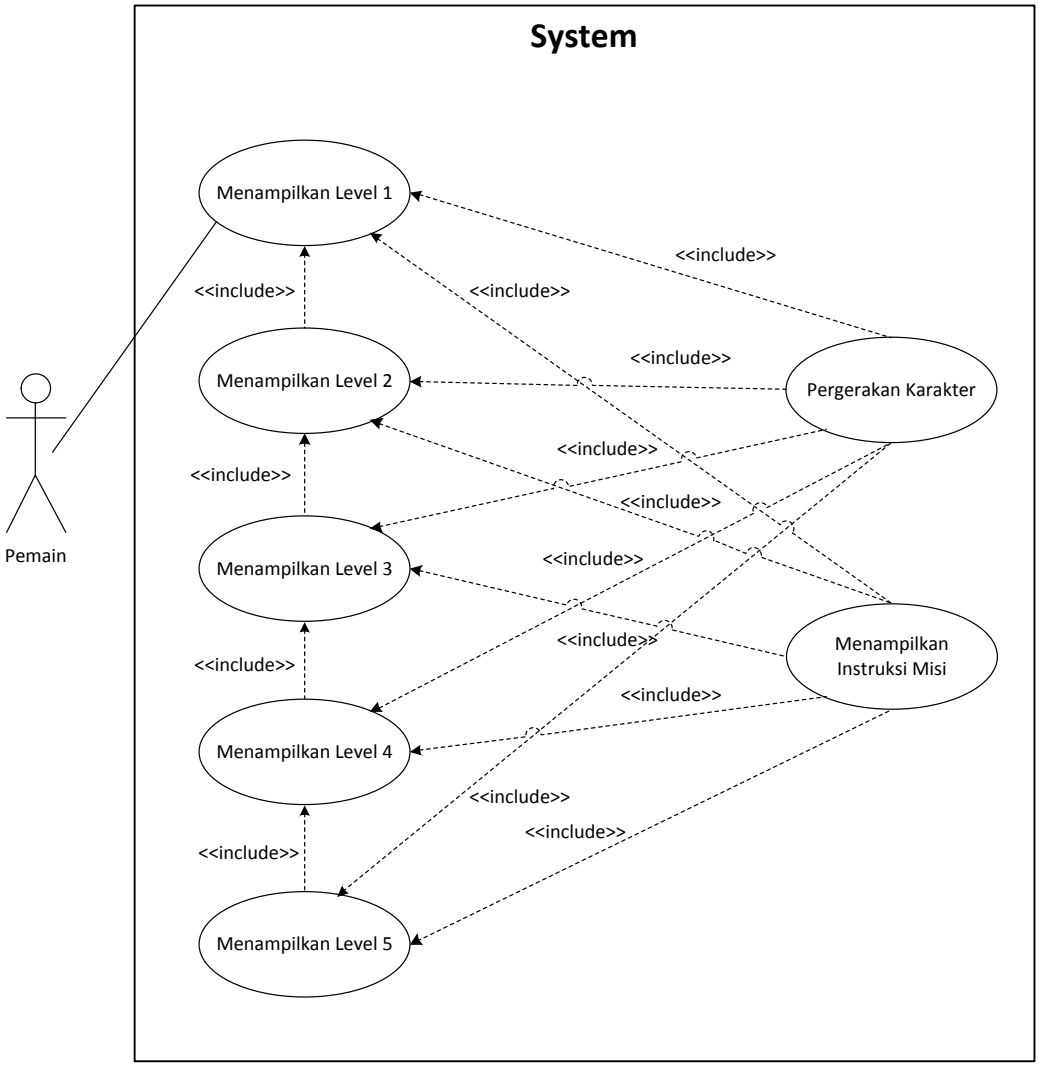
#### 3.1.11.1. Identifikasi Aktor

Tahap pertama yang dilakukan dalam melakukan analisis berorientasi objek menggunakan UML adalah menentukan aktor atau pengguna sistem. Kata aktor dalam konteks UML, menampilkan peran (*roles*) yang pengguna (atau sesuatu di luar sistem yang dikembangkan yang dapat berupa perangkat keras, end *user*, sistem yang lain, dan sebagainya).

#### 3.1.11.2. *Use Case Diagram*

*Use case Diagram* merupakan konstruksi untuk mendeskripsikan hubungan-hubungan yang terjadi antar aktor dengan aktivitas yang terdapat pada sistem. Sasaran pemodelan *use case* diantaranya adalah mendefinisikan kebutuhan

fungsional dan operasional sistem dengan mendefinisikan skenario penggunaan sistem yang akan dibangun. Dari hasil analisis aplikasi yang ada maka *usecase diagram* untuk *game 3D Save Our Planet from Extiction* dapat dilihat pada gambar 3.17.




**Gambar 3.17** ***Use Case Diagram game Save Our Planet From Extinction***

### 3.1.11.3. Definisi *Actor*

Definisi *Actor* berfungsi untuk menjelaskan *Actor* yang terdapat pada *Use Case Diagram*. Definisi *Actor* diterangkan pada Tabel 3.12.

**Tabel 3.12 Definisi *Actor***

| ***Actor*** | **Deskripsi** |
|---|---|
| Pemain | Orang yang akan memainkan *Game*. |

### 3.1.11.4. Definisi *Use Case*

Definisi *Use Case* berfungsi untuk menjelaskan proses yang terdapat pada setiap *Use Case*.

**Tabel 3.13 Definisi *Use Case***

| **No** | ***Use Case*** | **Deskripsi** |
|---|---|---|
| **1** | Menampilkan *Level* 1 | Untuk menampilkan permainan *level* 1 |
| **2** | Menampilkan *Level* 2 | Untuk menampilkan permainan *Level* 2 |
| **3** | Menampilkan *Level* 3 | Untuk menampilkan permainan *Level* 3 |
| **4** | Menampilkan *Level* 4 | Untuk menampilkan permainan *Level* 4 |
| **5** | Menampilkan *Level 5* | Untuk menampilkan permainan *Level 5* |
| **6** | Pergerakan Karakter | Untuk menggerakkan karakter |
| **7** | Menampilkan Instruksi Misi | Untuk Menampilkan instruksi misi tiap *level*-nya |

### 3.1.11.5. *Use Case Skenario*

Skenario *Use Case* menggambarkan alur penggunaan sistem dimana setiap skenario digambarkan dari sudut pandang aktor, seseorang atau piranti yang berinteraksi dengan perangkat lunak dalam berbagai cara.

#### 3.1.11.5.1. *Skenario Use Case Level 1*

**Tabel 3.14 Skenario *Use Case Level* 1**

| **Identifikasi** | |
|---|---|
| **Nomor** | 1 |
| **Nama** | *Level* 1 |

| | |
|---|---|
| **Tujuan** | Menampilkan *Level* 2 permainan |
| **Deskripsi** | Proses untuk menampilkan *Level* 2 permainan |
| **Aktor** | Pemain/Pengguna |
| **Skenario Utama** | |
| **Kondisi Awal** | Pemain berada menu utama permainan |

| **Aksi Aktor** | **Reaksi Sistem** |
|---|---|
| 1. Memilih menu mulai | |
| | 2. Menampilkan *loading* permainan *level* 1<br>3. menampilkan *Level* 1 permainan<br>4. Menempatkan *player* pada area permainan |
| 5. Input tombol "*Keyboard*" W,A,S,D | |
| | 6. Menggerakan karakter utama<br>7. Menyentuh *trigger* instruksi<br>8. Menampilkan instruksi |
| 9. Mencapai tujuan | |
| | 10. Menyentuh *trigger* tujuan<br>11. Menghentikan pencarian jalur<br>12. Menuju *level* berikutnya |

| | |
|---|---|
| **Kondisi Akhir** | Menampilkan *Level* 2 permainan |

### 3.1.11.5.2. *Skenario Use Case Level 2*

**Tabel 3.15 Skenario *Use Case Level* 2**

| **Identifikasi** | |
|---|---|
| **Nomor** | 2 |
| **Nama** | *Level* 2 |
| **Tujuan** | Menampilkan *Level* 3 permainan |

| **Deskripsi** | Proses untuk menampilkan *Level* 3 permainan |
|---|---|
| **Aktor** | Pemain/Pengguna |
| **Skenario Utama** | |
| **Kondisi Awal** | Pemain berada menu utama permainan |

| **Aksi Aktor** | **Reaksi Sistem** |
|---|---|
| 1. Memilih menu mulai | |
| | 2. Menampilkan *loading* permainan *level* 2<br>3. Menampilkan *Level* 2 permainan<br>4. Menempatkan *player* pada area permainan<br>5. Menentukan lokasi pemunculan musuh<br>6. Memunculkan musuh<br>7. Melakukan pencarian<br>8. Menyerang karakter<br>9. Mengenai karakter<br>10. Mengurangi darah karakter |
| 11. Input tombol "*Keyboard*" W,A,S,D | |
| | 12. Menggerakan karakter utama<br>13. Menyentuh *trigger* instruksi<br>14. Menampilkan instruksi |
| 15. Menekan "*Mouse*" klik kiri | |
| | 16. Mengurangi jumlah peluru<br>17. Tembakan mengenai musuh<br>18. Mengurangi jumlah darah musuh |
| 19. Mencapai tujuan | |

| | |
|---|---|
| | 20. Menyentuh *trigger* tujuan<br>21. Menghentikan pencarian jalur |
| | 22. Menuju *level* berikutnya |
| **Kondisi Akhir** | Menampilkan *Level* 3 permainan |

### 3.1.11.5.3. *Skenario Use Case Level 3*

**Tabel 3.16 Skenario *Use Case Level* 3**

| **Identifikasi** | |
|---|---|
| **Nomor** | 3 |
| **Nama** | *Level* 3 |
| **Tujuan** | Menampilkan *Level* 4 permainan |
| **Deskripsi** | Proses untuk menampilkan *Level* 4 permainan |
| **Aktor** | Pemain/Pengguna |
| **Skenario Utama** | |
| **Kondisi Awal** | Pemain berada menu utama permainan |
| **Aksi Aktor** | **Reaksi Sistem** |
| 1. Memilih menu mulai | |
| | 2. Menampilkan *loading* permainan *level* 3<br>3. Menampilkan *Level* 3 permainan<br>4. Menempatkan *player* pada area permainan |
| 5. Input tombol "*Keyboard*" W,A,S,D | |
| | 6. Menggerakan karakter utama<br>7. Menyentuh *trigger* instruksi<br>8. Menampilkan instruksi |
| 9. Mencapai tujuan | |

| | |
|---|---|
| | 10. Menyentuh *trigger* tujuan<br>11. Menghentikan pencarian jalur<br>12. Menuju *level* berikutnya |
| **Kondisi Akhir** | Menampilkan *Level* 4 permainan |

### 3.1.11.5.4. *Skenario Use Case Level 4*

**Tabel 3.17 Skenario *Use Case Level* 4**

| **Identifikasi** | |
|---|---|
| **Nomor** | 4 |
| **Nama** | *Level* 4 |
| **Tujuan** | Menampilkan *Level* 5 permainan |
| **Deskripsi** | Proses untuk menampilkan *Level* 5 permainan |
| **Aktor** | Pemain/Pengguna |
| **Skenario Utama** | |
| **Kondisi Awal** | Pemain berada menu utama permainan |
| **Aksi Aktor** | **Reaksi Sistem** |
| 1. Memilih menu mulai | |
| | 2. Menampilkan *Level* 4 permainan<br>3. Menempatkan *player* pada area permainan<br>4. Menentukan lokasi pemunculan musuh<br>5. Memunculkan musuh<br>6. Melakukan Pencarian<br>7. Menembak karakter<br>8. Peluru musuh berkurang<br>9. Mengenai karakter<br>10. Mengurangi darah karakter |
| 11. Input tombol "*Keyboard*" | |

| | |
|---|---|
| W,A,S,D | |
| | 12. Menggerakan karakter utama<br>13. Menyentuh *trigger* instruksi<br>14. Menampilkan instruksi |
| 15. Meng-klik "*Mouse*" kiri | |
| | 16. Mengurangi jumlah peluru<br>17. Tembakan mengenai musuh<br>18. Mengurangi jumlah darah musuh |
| 19. Mencapai tujuan | |
| | 20. Menyentuh *trigger* tujuan<br>21. Menghentikan pencarian jalur<br>22. Menuju *level* berikutnya |
| **Kondisi Akhir** | Menampilkan *Level* 5 permainan |

### 3.1.11.5.5. *Skenario Use Case Level 5*

**Tabel 3.18 Skenario *Use Case Level* 5**

| **Identifikasi** | |
|---|---|
| **Nomor** | 5 |
| **Nama** | *Level* 5 |
| **Tujuan** | Menampilkan *ending* permainan |
| **Deskripsi** | Proses untuk menampilkan *ending* permainan |
| **Aktor** | Pemain/Pengguna |
| **Skenario Utama** | |
| **Kondisi Awal** | Pemain berada menu utama permainan |
| **Aksi Aktor** | **Reaksi Sistem** |
| 1. Memilih menu mulai | |
| | 2. Menampilkan *Level* 5 permainan |

| | |
|---|---|
| | 3. Menempatkan *player* pada area permainan |
| 4. Input tombol "*Keyboard*" W,A,S,D | |
| | 5. Menggerakan karakter utama<br>6. Mengenai item<br>7. Mengecek item |
| 8. Mencapai tujuan | |
| | 9. Menyentuh *trigger* tujuan<br>10. Menghentikan pencarian jalur<br>11. Menuju *level* berikutnya |
| **Kondisi Akhir** | Menampilkan *ending* permainan |

### 3.1.11.5.6. *Skenario Use Case* Pergerakan Karakter

Skenario*Use Case* pergerakkan karakter berfungsi untuk menampilkan interaksi antara *actor* dengan *use case* pergerakkan karakter, tabel pergerakkan karakter dapat dilihat pada tabel 3.19.

**Tabel 3.19 Skenario *Use Case* Pergerakan Karakter**

| **Identifikasi** | |
|---|---|
| **Nomor** | 6 |
| **Nama** | Pergerakan karakter |
| **Tujuan** | Untuk menggerakan karakter yang dimainkan |
| **Deskripsi** | Proses untuk menggerakan karakter |
| **Aktor** | Pemain/Pengguna |
| **Skenario Utama** | |
| **Kondisi Awal** | Karakter diam |
| **Aksi actor** | **Reaksi Sistem** |
| 1. Input tombol (W, A, S, D) pada *keyboard* | |

| | |
|---|---|
| | 2. Menggerakkan karakter sesuai dengan tombol yang ditekan<br>3. Karakter bergerak maju<br>4. Karakter bergerak ke kiri<br>5. Karakter bergerak mundur<br>6. Karakter bergerak ke kanan |
| **Kondisi Akhir** | Karakter bergerak |

#### 3.1.11.5.7. Skenario *Use Case* Menampilkan Instruksi Misi

Skenario *Use Case* menampilkan petunjuk misi berfungsi untuk menampilkan misi yang ada di dalam *game*, tabel menampilkan petunjuk misi dapat dilihat pada tabel 3.20.

**Tabel 3.20 Skenario *Use Case* Menampilkan Petunjuk Misi**

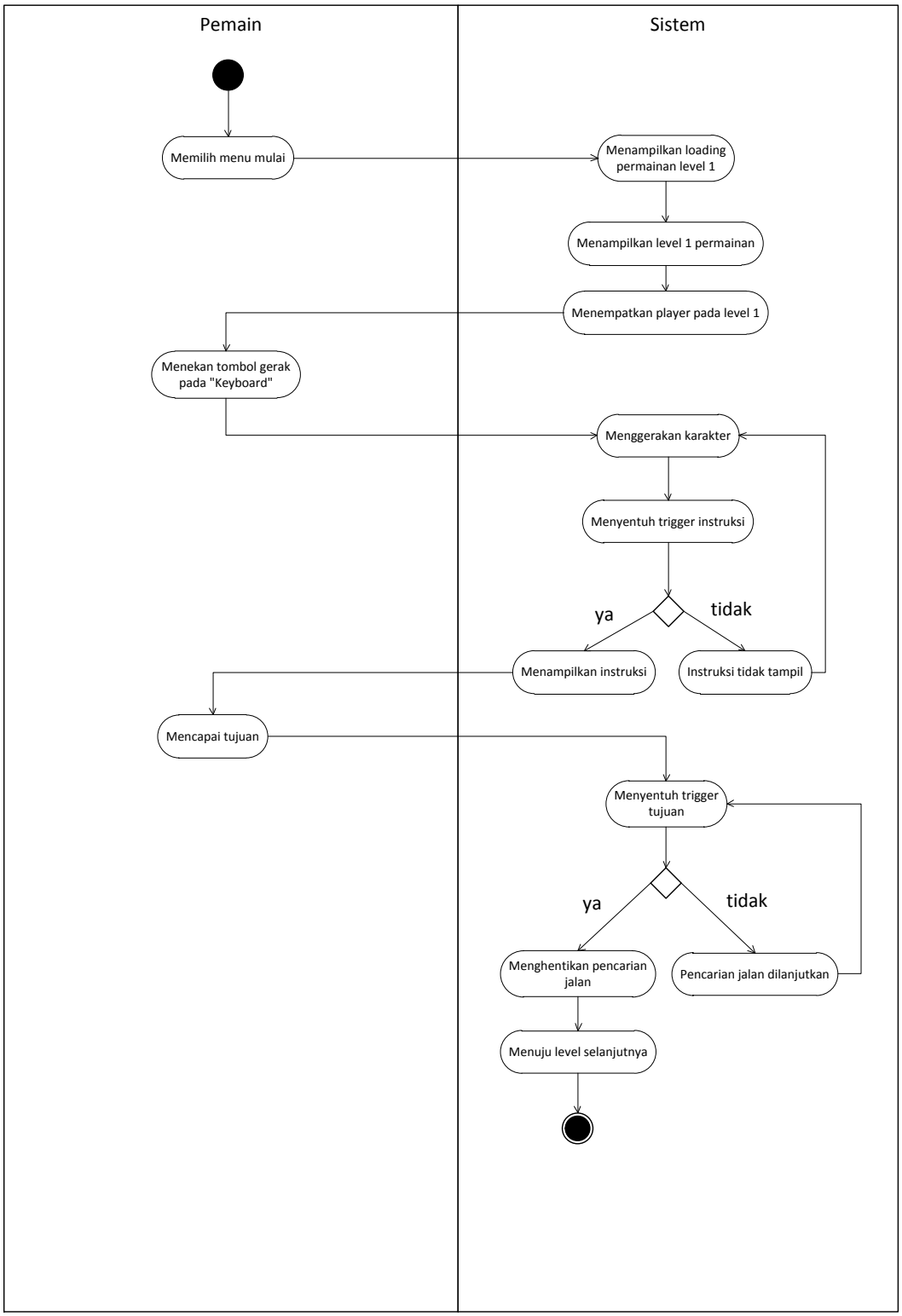
| **Identifikasi** | |
|---|---|
| **Nomor** | 7 |
| **Nama** | Menampilkan Petunjuk Misi |
| **Tujuan** | Untuk menampilkan misi yang ada pada *game* |
| **Deskripsi** | Proses untuk menampilkan Instruksi misi |
| **Aktor** | Pemain/Pengguna |
| **Skenario Utama** | |
| **Kondisi Awal** | Menyajikan permainan |
| **Aksi actor** | **Reaksi Sistem** |
| 1. Menggerakkan karakter menuju trigger instruksi | |
| | 2. Menghilangkan *trigger*<br>3. Memanggil instruksi misi |
| **Kondisi Akhir** | Menampilkan instruksi misi |

#### 3.1.11.6. *Activity Diagram*

*Activity diagram* menggambarkan berbagai aliran aktifitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang

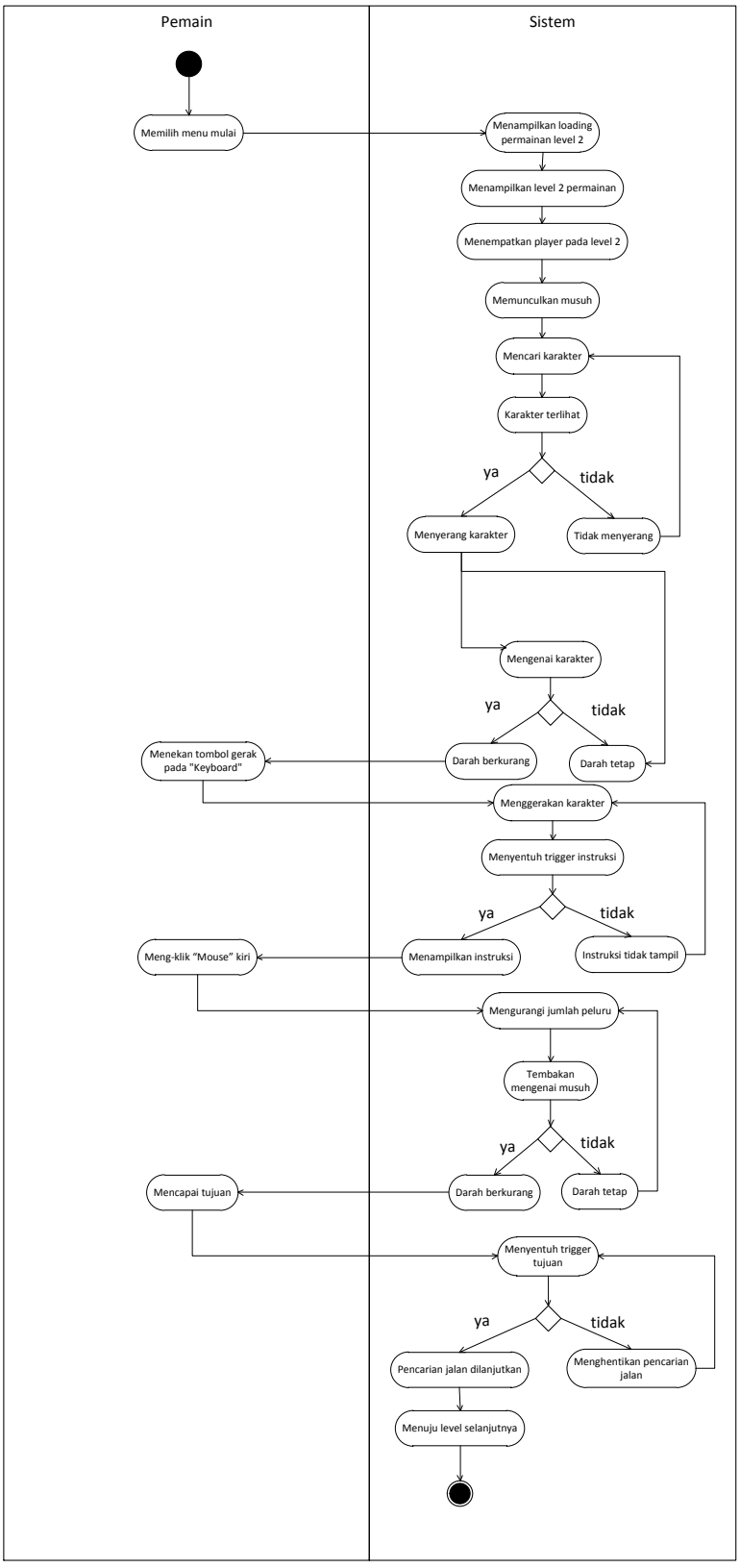
mungkin terjadi, dan bagaimana berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *Activity diagram* merupakan *state diagram* khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tidak menggambarkan *behaviour* internal sebuah sistem dan interaksi antar subsistem secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktifitas dari *level* atas secara umum. Sebuah aktivitas dapat direalisasikan oleh satu *Use Case* atau lebih. Aktifitas menggambarkan proses yang berjalan, sementara *Use Case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktifitas.

### 3.1.11.6.1. *Activity Diagram Level* 1



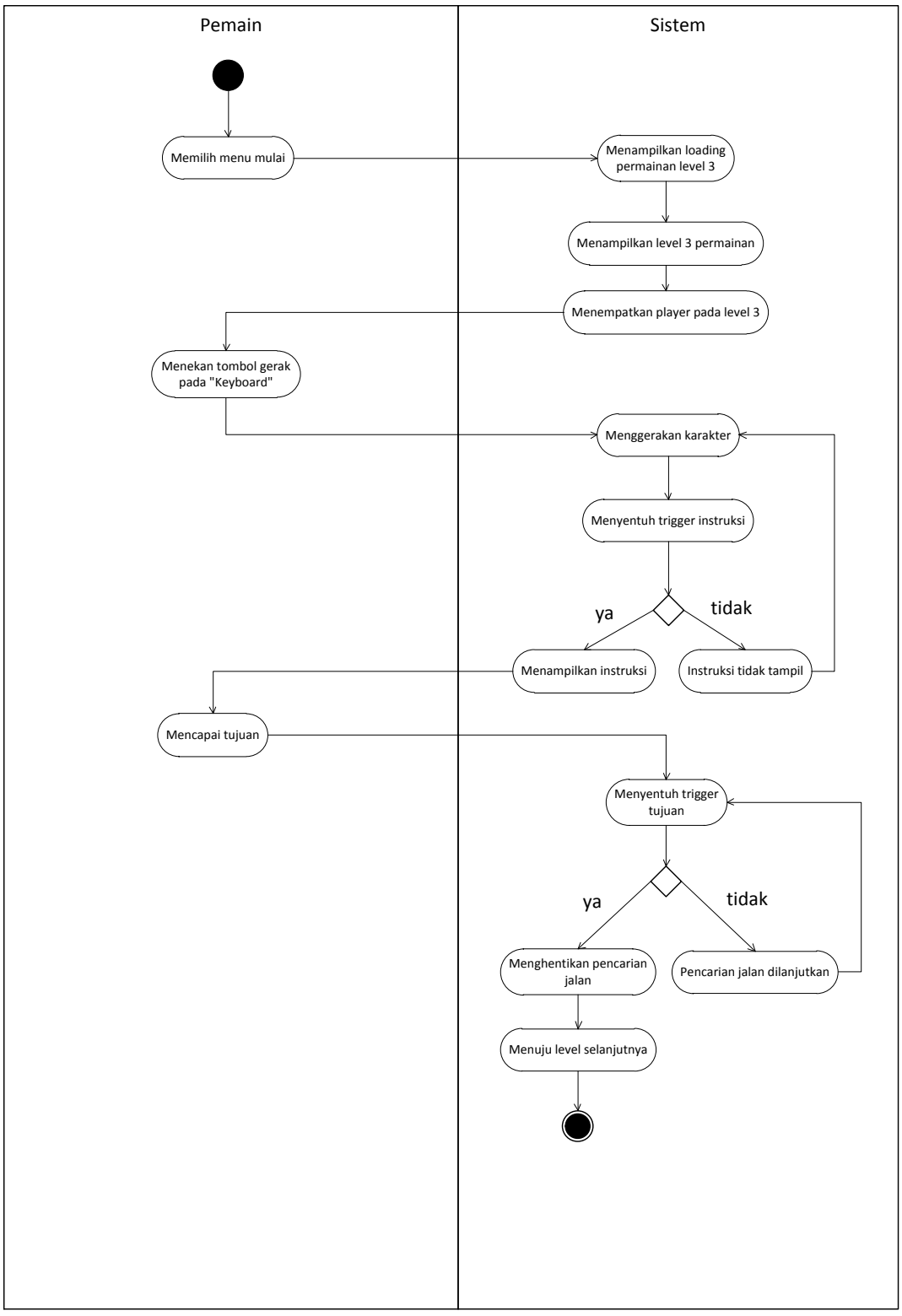

**Gambar 3.18 *Acitivity Diagram Level* 1**

### 3.1.11.6.2. *Activity Diagram Level* 2



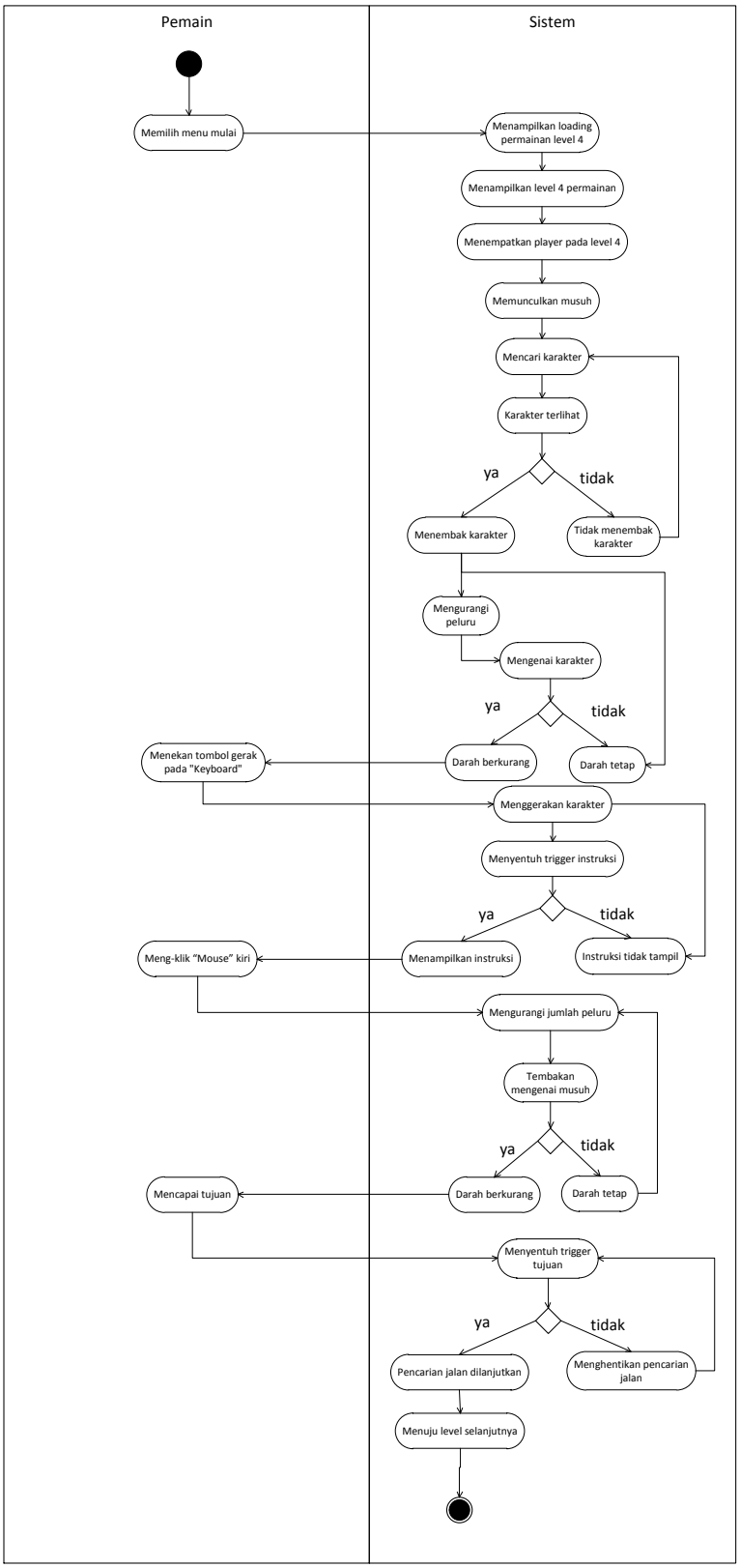

**Gambar 3.19** ***Acitivity Diagram Level*** **2**

### 3.1.11.6.3. *Activity Diagram Level 3*



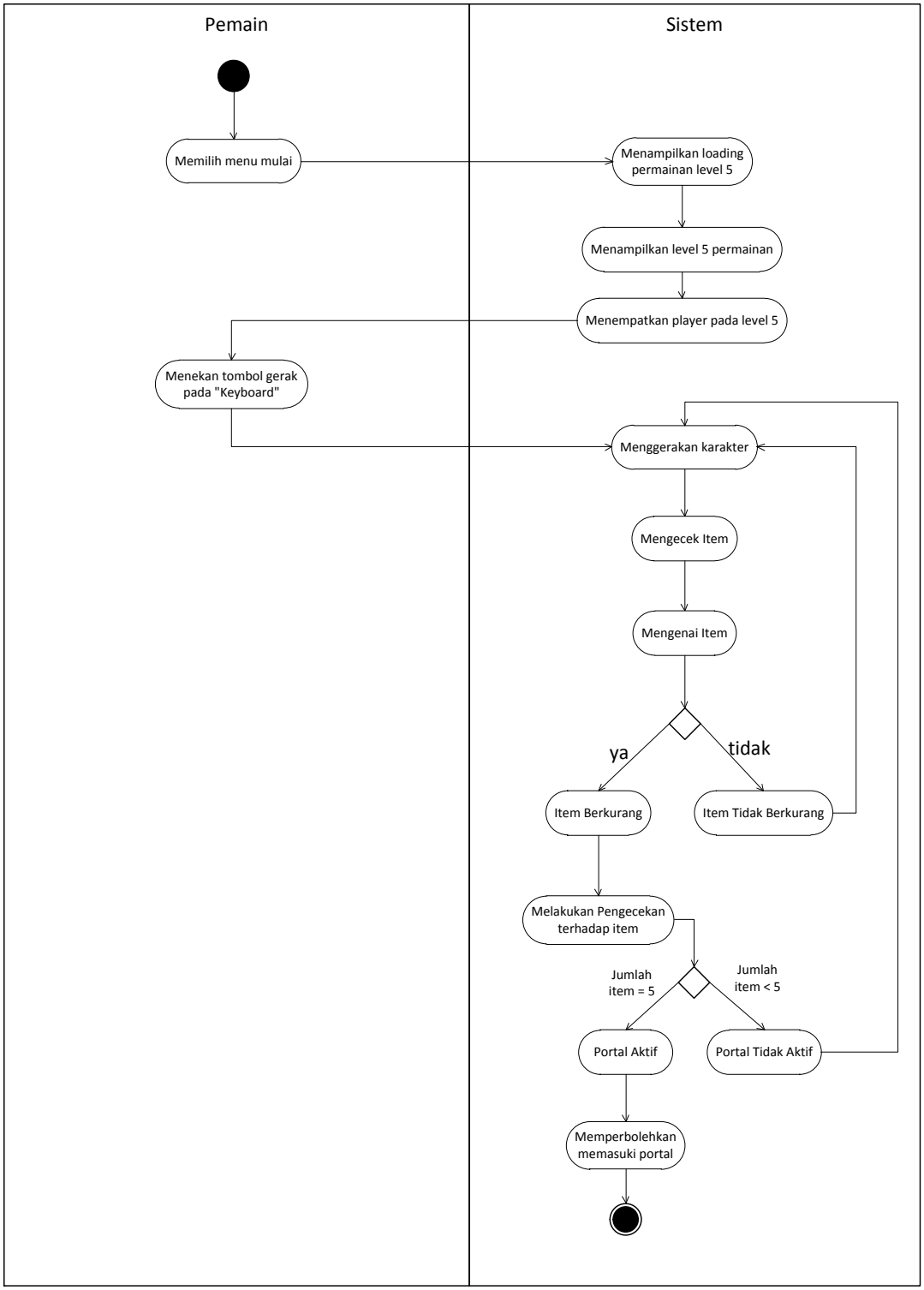

**Gambar 3.20 *Acitivity Diagram Level* 3**

### 3.1.11.6.4. *Activity Diagram Level* 4



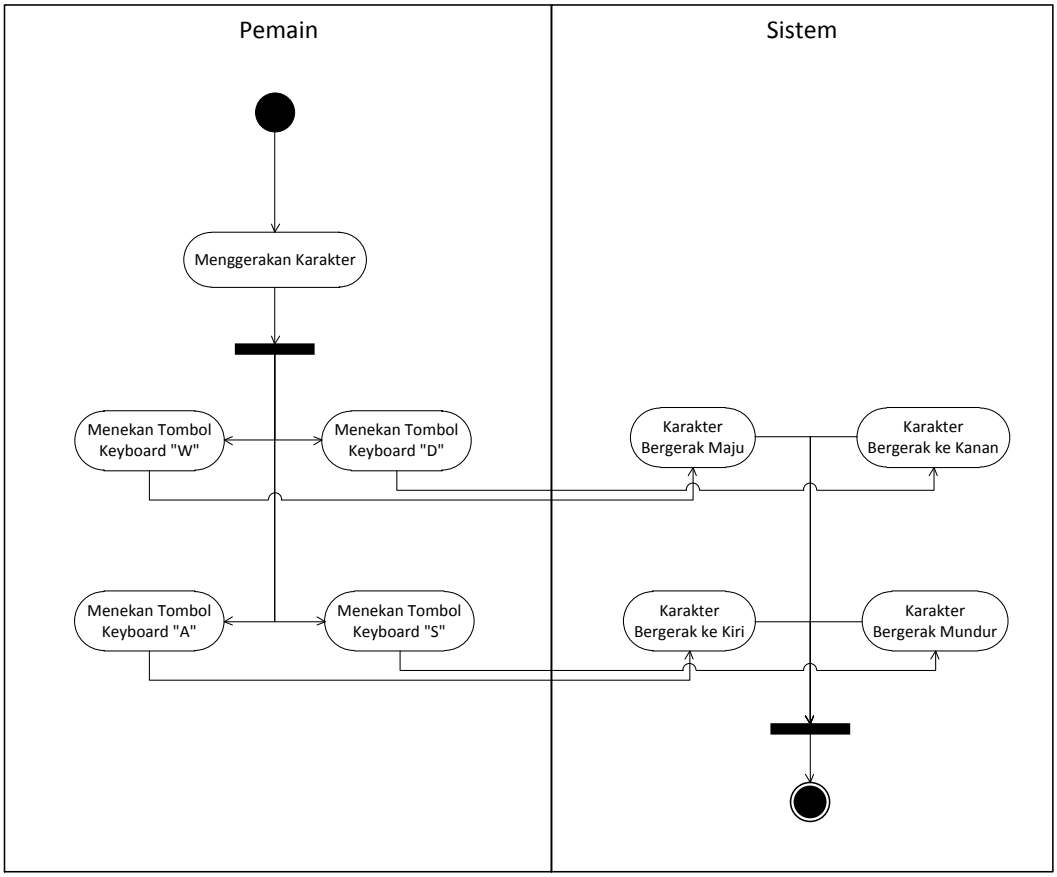

**Gambar 3.21 *Acitivity Diagram Level* 4**

### 3.1.11.6.5. *Activity Diagram Level* 5



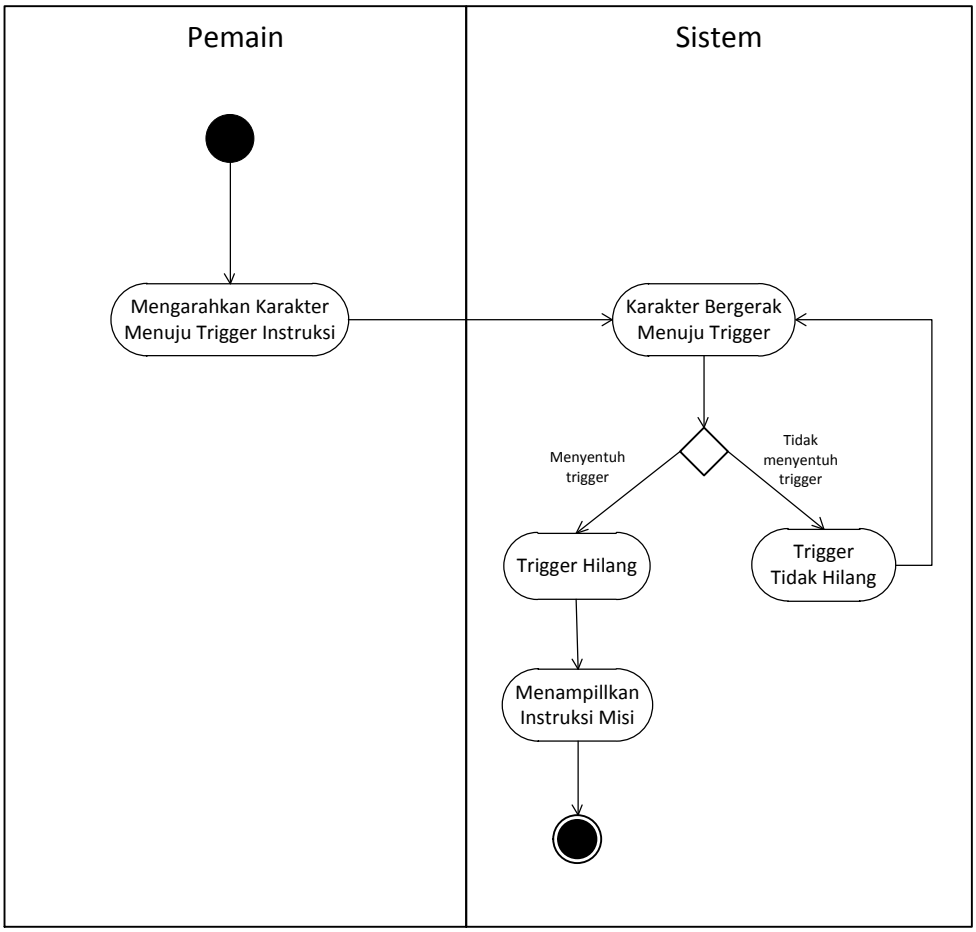

**Gambar 3.22 *Acitivity Diagram Level* 5**

### 3.1.11.6.6. *Activity Diagram* Pergerakan Karakter




**Gambar 3.23 *Acitivity Diagram* Pergerakan Karakter**

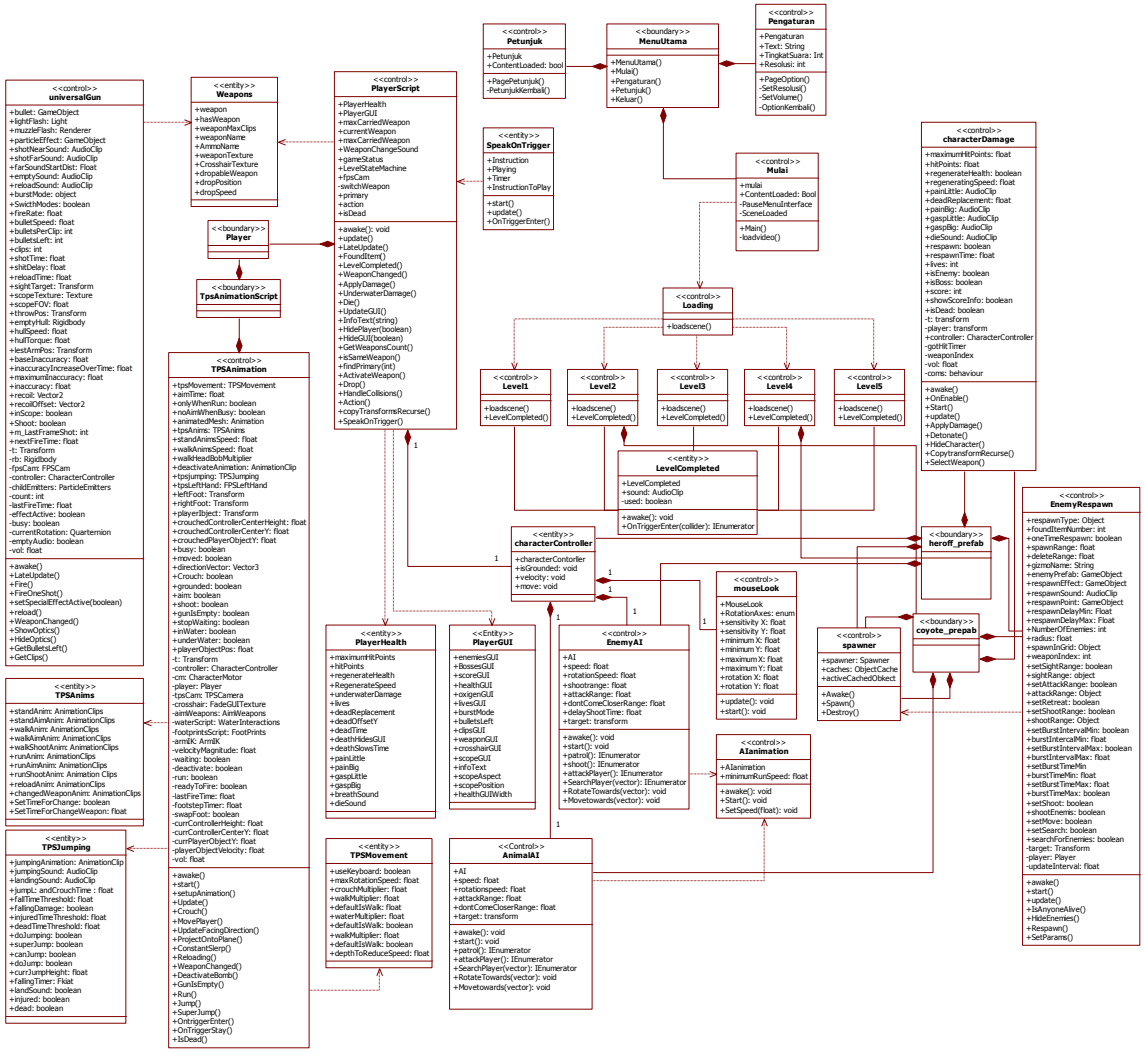
### 3.1.11.6.7. *Activity Diagram* Menampilkan Instruksi Misi




**Gambar 3.24 *Acitivity Diagram* Menampilkan Instruksi Misi**

### 3.1.11.7. *Class Diagram*

*Class* adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi dan lain-lain. Gambar 3.25 berikut adalah *class diagram* dari aplikasi *game*.



**Gambar 3.25 *Class Diagram Game Save Our Planet From Extinction***

### 3.1.11.8. *Sequence Diagram*

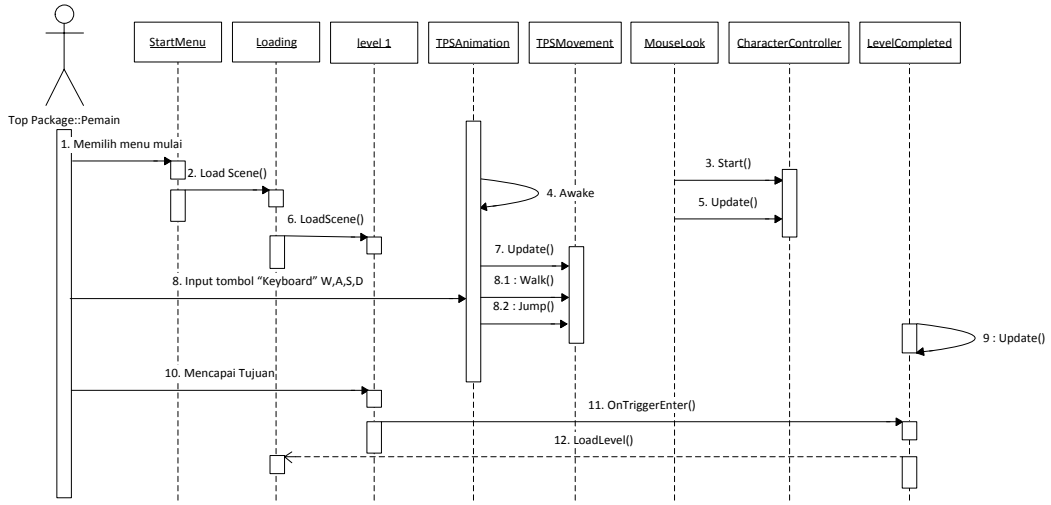
*Sequence diagram* menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).

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

Berikut ini beberapa *sequencial diagram* yang terdapat pada *game save our planet from extinction* yaitu :

### 3.1.11.8.1. *Sequence Diagram* Menampilkan Permainan *Level* 1

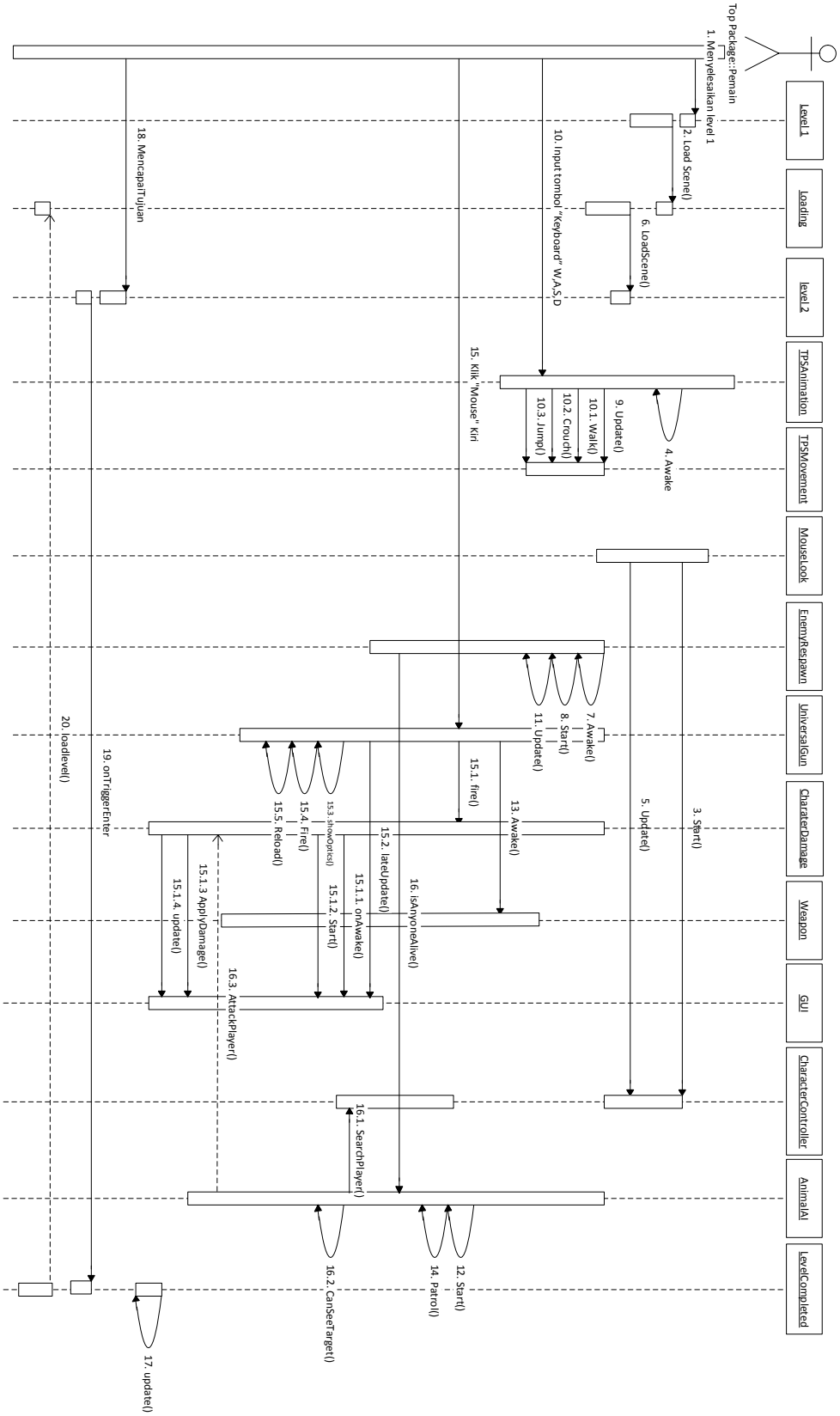
*Sequence diagram* untuk menampilkan permainan *level* 1 bisa dilihat pada gambar 3.26.




**Gambar 3.26 *Sequence Diagram* Menampilkan Permainan *Level* 1**

### 3.1.11.8.2. *Sequence Diagram* Menampilkan Permainan *Level* 2

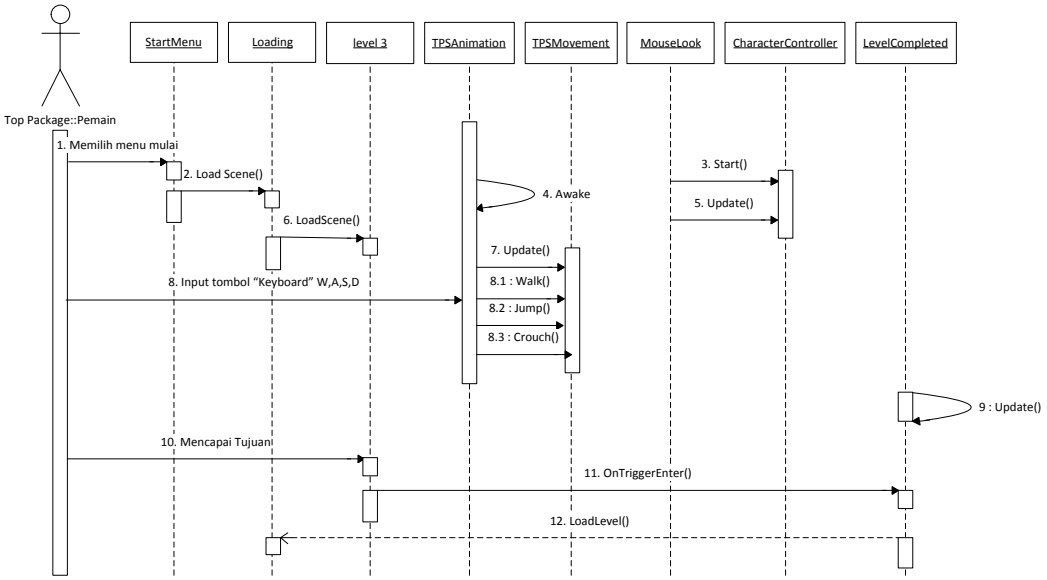
*Sequence diagram* untuk menampilkan permainan *level* 2 bisa dilihat pada gambar 3.27.




**Gambar 3.27 *Sequence* Diagram Menampilkan Permainan *Level* 2**

### 3.1.11.8.3. *Sequence Diagram* Menampilkan Permainan *Level* 3

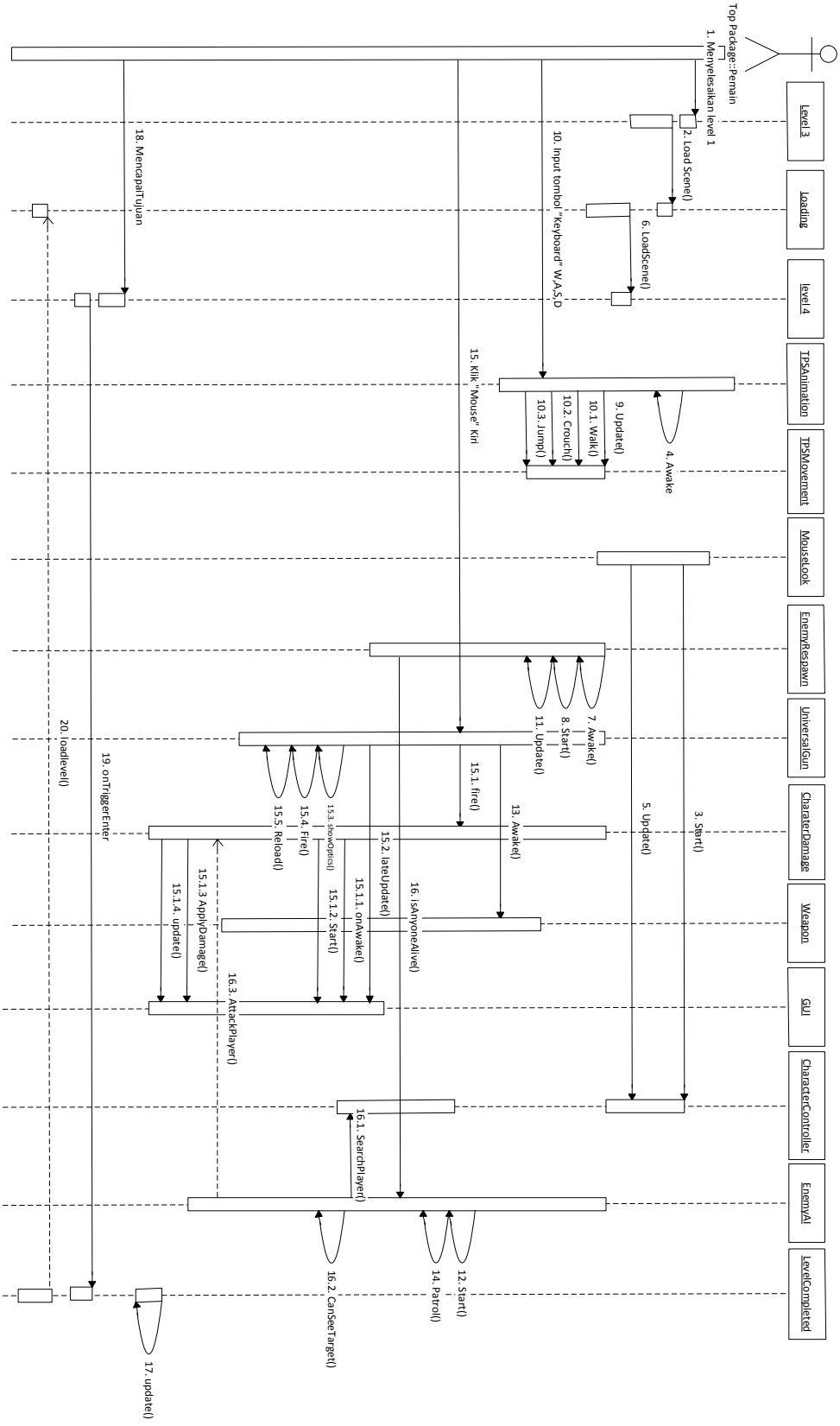
*Sequence diagram* untuk menampilkan permainan *level* 3 bisa dilihat pada gambar 3.28.




**Gambar 3.28 *Sequence* Diagram Menampilkan Permainan *Level* 3**

### 3.1.11.8.4. *Sequence Diagram* Menampilkan Permainan *Level* 4

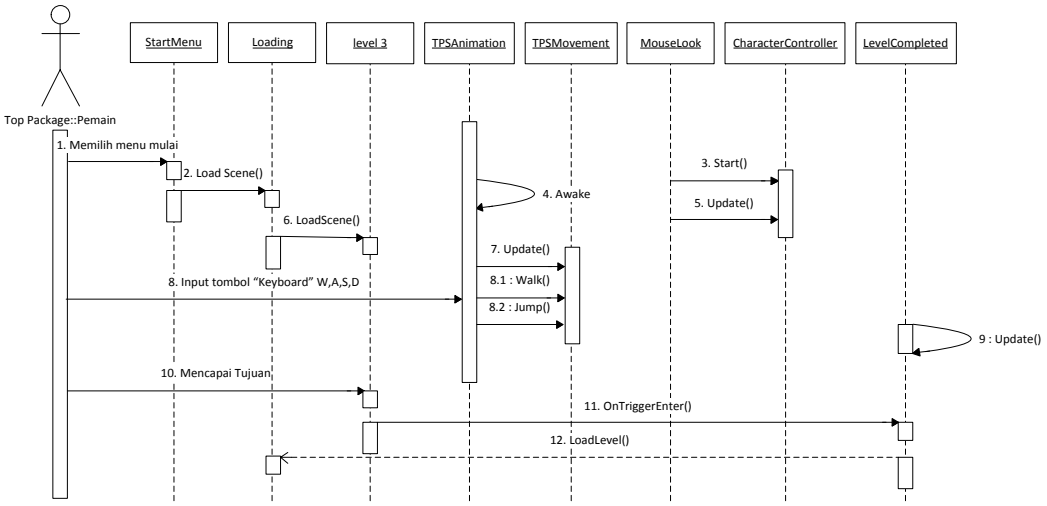
*Sequence diagram* untuk menampilkan permainan *level* 4 bisa dilihat pada gambar 3.29.




**Gambar 3.29 *Sequence* Diagram Menampilkan Permainan *Level* 4**

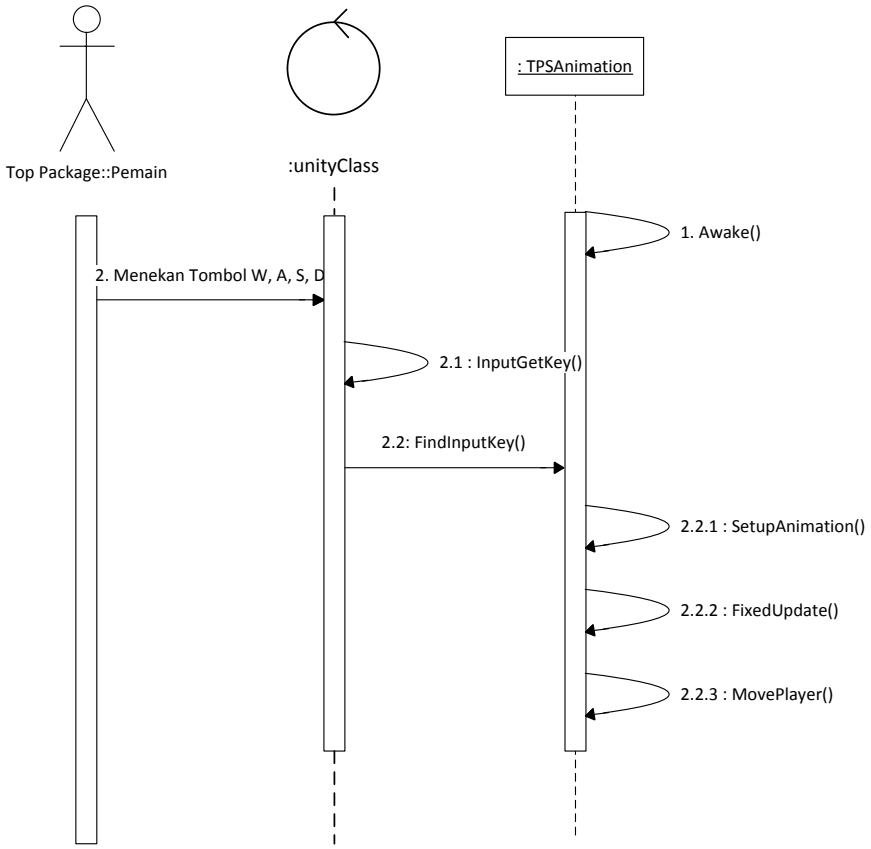
### 3.1.11.8.5. *Sequence Diagram* Menampilkan Permainan *Level* 5

*Sequence diagram* untuk menampilkan permainan *level* 5 bisa dilihat pada gambar 3.30.



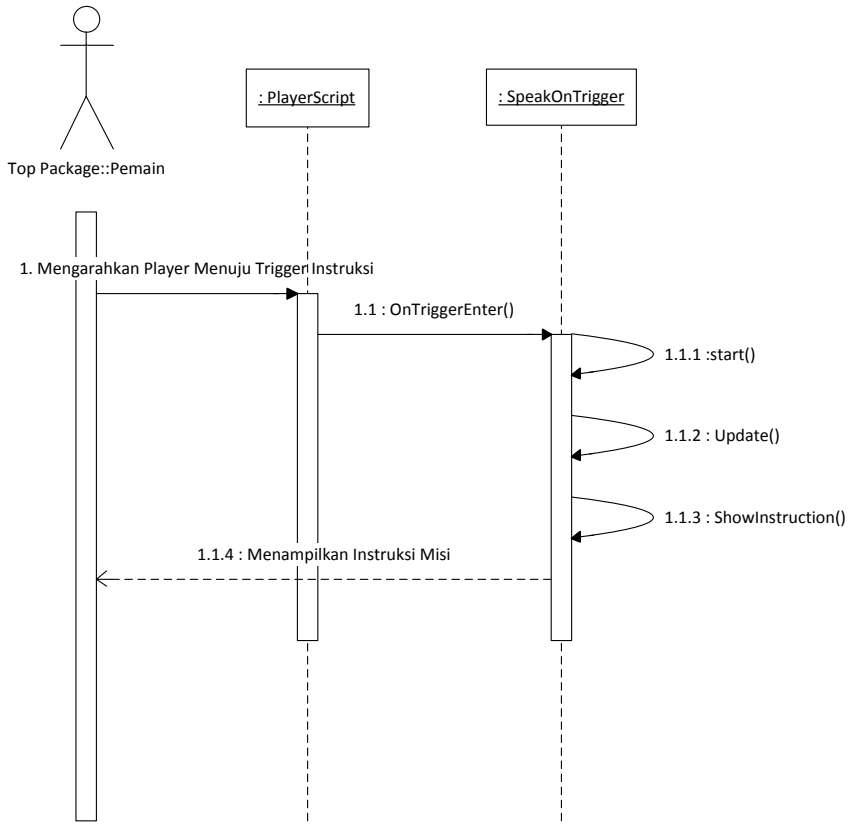

**Gambar 3.30 *Sequence* Diagram Menampilkan Permainan *Level* 5**

### 3.1.11.8.6. *Sequence Diagram* Pergerakan Karakter




**Gambar 3.31 *Sequence Diagram* Pergerakan Karakter**

### 3.1.11.8.7. *Sequence Diagram* Menampilkan Instruksi Misi




**Gambar 3.32 *Sequence* Diagram Menampilkan Instruksi Misi**

## 3.2. Perancangan Sistem

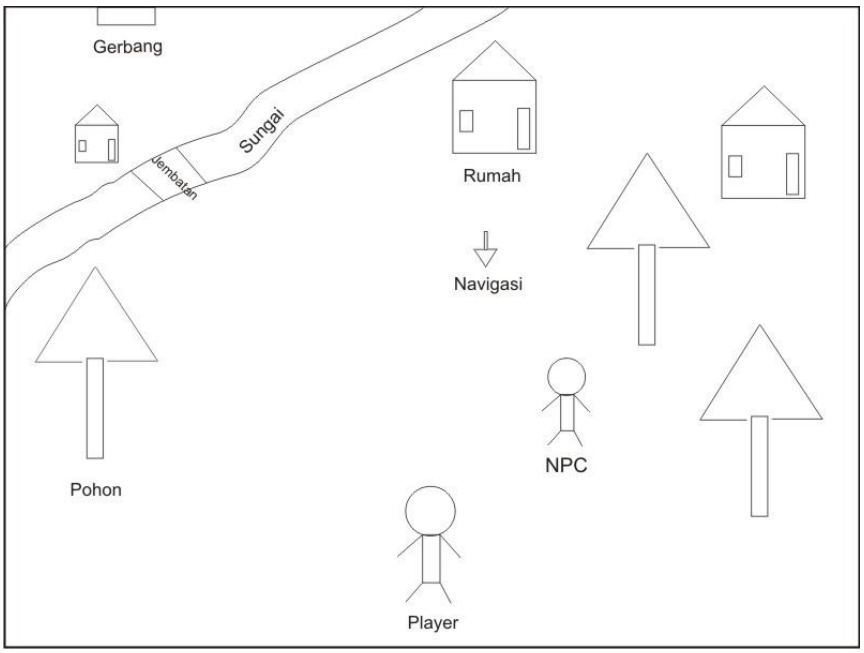
Perancangan adalah suatu bagian dari metodologi pengembangan pembangunan suatu perangkat lunak yang dilakukan setelah tahapan untuk memberikan gambaran secara terperinci. Berdasarkan uraian diatas perancangan sistem merupakan tahapan dari siklus pengembangan sistem yang didefinisikan dari kebutuhan-kebutuhan fungsional dan persiapan untuk rancang bangun implementasi yang menggambarkan bagaimana suatu sistem dibentuk yang dapat berupa penggambaran, perancangan dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah kedalam satukesatuan yang utuh dan berfungsi, juga menyangkut konfigurasi dari komponen-komponen perangkat keras dan perangkat lunak dari suatu sistem.

### 3.2.1. Perancangan Komponen Permainan

Perancangan komponen permainan meliputi karakter dan *storyboard* yang didalamnya menjelaskan karakter beserta senjata apa saya yang dimiliki pemain maupun musuh (NPC) dan *storyboard* sebagai gambar perancangan pembuatan *game save our planet from extinction*

#### 3.2.1.1. *Storyboard*

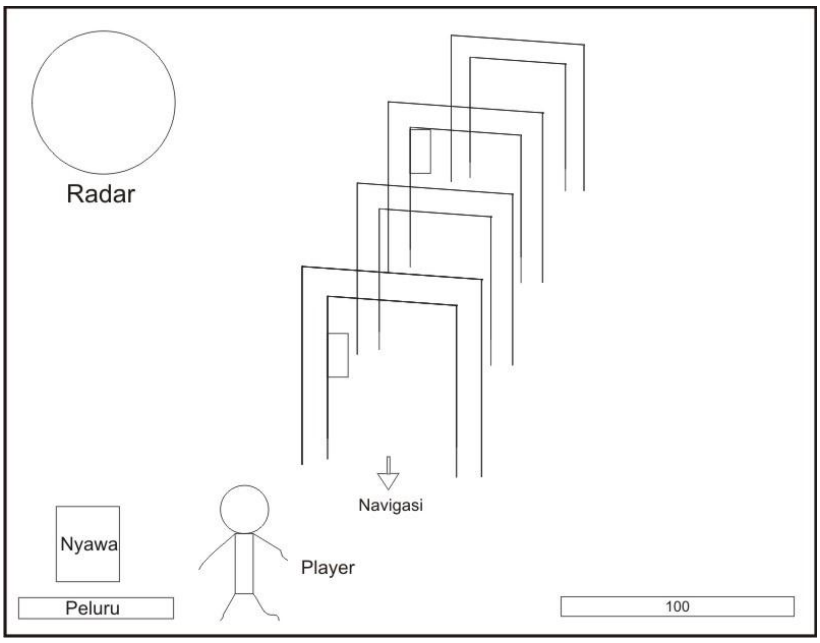
*Storyboard* adalah sketsa gambar yang disusun berurutan sesuai dengan naskah, dengan storyboard dapat menyampaikan ide cerita kepada orang lain dengan lebih mudah, karena dapat menggiring khayalan seseorang mengikuti gambar-gambar yang tersaji, sehingga menghasilkan persepsi yang sama pada ide cerita. Adapun *storyboard* untuk aplikasi *game save our planet from extinction* ini adalah sebagai berikut.




**Gambar 3.33 *Storyboard Map Level* 1**

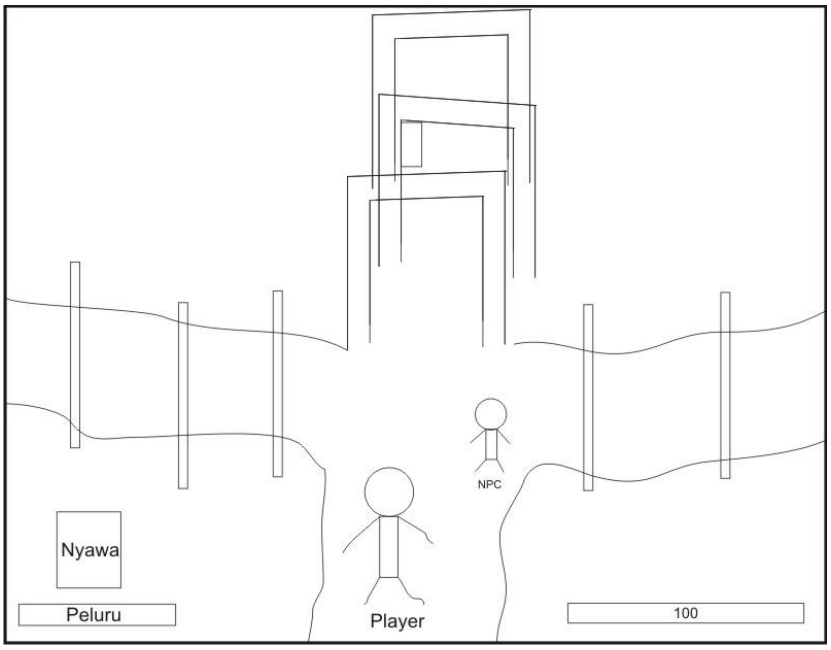
Gambar 3.33 menceritakan ketika pemain memulai permainan di awal *level* dengan berada di suatu perkampungan di daerah Kalimantan dan disinilah awal petualangan dimulai. Gambar 3.33 menceritakan ketika pemain berada tepat di desa tersebut, disinilah pemain harus mencari portal keluar untuk menuju ke

*level* berikutnya dengan mengikuti instruksi yang ada tetapi sebelumnya pemain dituntut untuk mendapatkan informasi dari warga desa.



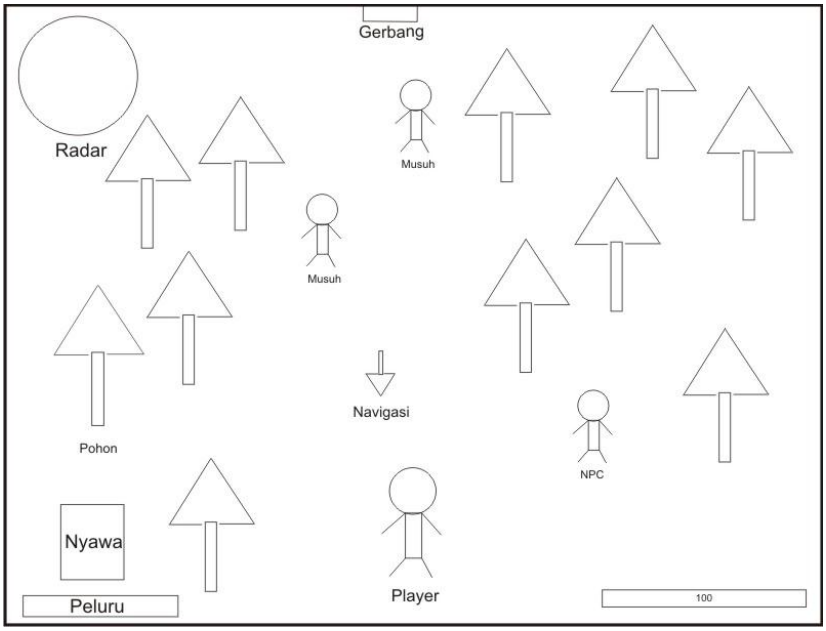

**Gambar 3.34 *Storyboard Map Level* 2**

Gambar 3.34 memiliki *setting* di sebuah goa yang dipenuhi anjing-anjing. Ditempat ini pemain harus melumpuhkan anjing-anjing liar yang ada, suasana ditempat ini sangat gelap sehingga ada obor-obor di pinggir goa untuk menerangi jalan untuk menuju ke *level* berikutnya.



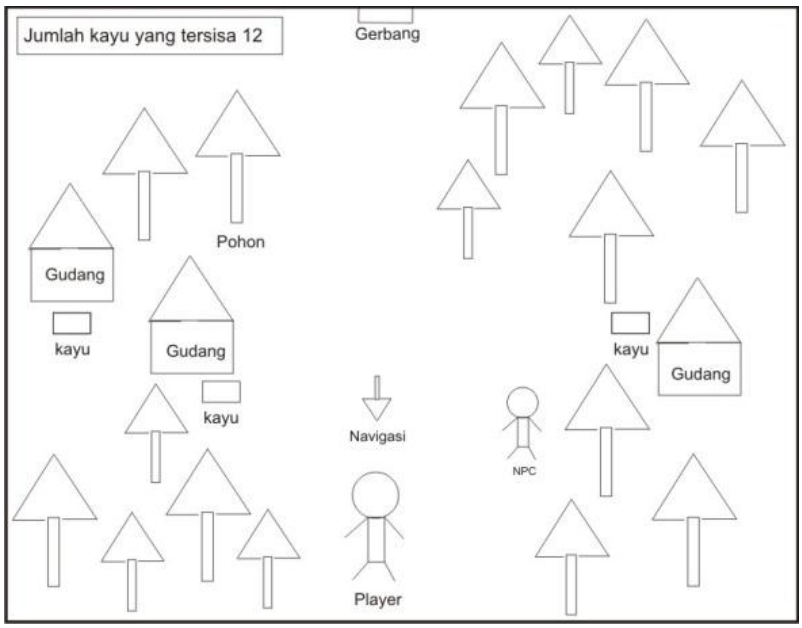

**Gambar 3.35 *Storyboard Map Level* 3**

Gambar 3.35 memiliki *setting* tempat yang sama yaitu di sebuah goa. Di *level* ini pemain dituntut menemukan jalan keluar dari goa tersebut, tetapi hal ini tidak mudah karena banyak sekali jalan yang bercabang dan pemain harus melewati rintangan yang ada di *level* ini agar dapat menuju ke *level* berikutnya.




**Gambar 3.36 *Storyboard Map Level* 4**

Gambar 3.36 menceritakan ketika pemain memulai permainan di *level* 4 dengan berada di suatu hutan belantara. Misi utama di *level* ini yaitu melumpuhkan semua musuh-musuh yang ada. Di *level* 4 ini terjadi pada sore hari. Gambar 3.36 menceritakan ketika pemain berada di hutan.




**Gambar 3.37 *Storyboard* Map *Level* 5**

Gambar 3.37 menceritakan ketika pemain memainkan permainan untuk menyelesaikan misi yang ada pada *level* 5. Misi utama di *level* ini pemain harus mengumpulkan semua kayu-kayu yang tersisa hasil dari penebangan yang disimpan di suatu gudang.

### 3.2.1.2. Pengenalan Karakter

Karakter utama dalam *game* ini bernama Arman, Arman merupakan seorang polisi hutan yang ditugaskan untuk pergi ke suatu pedesaan dalam menelusuri kerusakan hutan yang terjadi di sekitar desa tersebut karena adanya laporan-laporan dari warga desa. Karakter dirancang untuk melawan musuh-musuh yang ada di setiap *level*. Karakter dilengkapi dengan persenjataan berupa senapan laras panjang. *Game* ini merupakan *genre third person shooter* yang mempunyai sudut pandang orang ketiga, bentuk karakter dan properti yang dimiliki karakter utama dapat dilihat pada gambar 3.38, 3.39 dan Tabel 3.40.



**Gambar 3.38 Karakter Utama Tampak Depan**




**Gambar 3.39 Karakter Utama Tampak Belakang**

**Tabel 3.21 Detail Karakter**

| | |
|---|---|
| Nama | Arman |
| Darah | 100 |
| Kemampuan | melompat, menembak, jongkok, |

| | |
|---|---|
| | berjalan |
| Senjata | Senapan |

Senjata yang digunakan oleh Arman adalah senjata senapan laras panjang yang memiliki peluru sebanyak 25 buah yang dapat dilihat pada Gambar 3.40 dan *properties* senjata pada karakter dapat dilihat pada tabel 3.22.

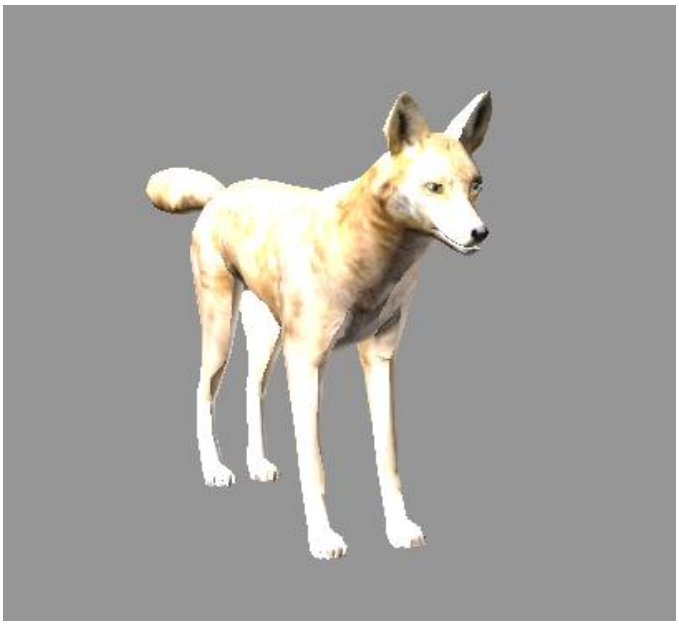


**Gambar 3.40 Senapan Laras Panjang**

**Tabel 3.22 Detail Senjata**

| **Properti** | **Senapan** |
|---|---|
| Damage | 80 |
| Peluru | 5 |
| Jumlah Clip | 5 |

Musuh yang terdapat pada *game* ini adalah anjing dan manusia, anjing dan manusia ini memiliki tingkat kesulitan berbeda-beda untuk dikalahkan. Musuh dapat dilihat pada Gambar 3.41 dan 3.42.



**Gambar 3.41 Karakter Musuh Anjing Liar**



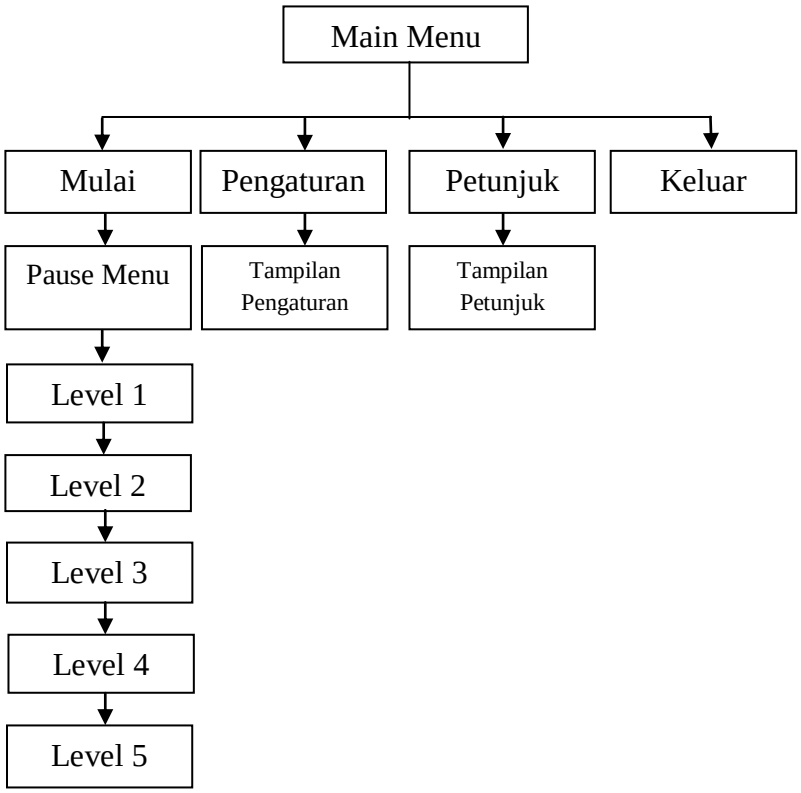
**Gambar 3.42 Karakter Musuh Manusia**

### 3.2.2. Perancangan Arsitektur

Setelah melakukan perancangan data pada sistem yang dibangun, maka dilakukanlah perancangan arsitektur. Perancangan arsitektur yang telah dibuat meliputi beberapa perancangan diantaranya struktur menu, perancangan antarmuka.

### 3.2.2.1. Perancangan Struktur Menu

Perancangan menu dibuat dibuat agar pengguna dapat menggunakan aplikasi Game dengan mudah, sehingga pengguna dapat dengan mudah memilih menu aplikasi yang sedang berjalan. Stuktur menu dapat dilihat pada gambar 3.43




**Gambar 3.43 Struktur Menu Utama**

### 3.2.2.2. Perancangan Antar Muka

Setelah melakukan perancangan data pada sistem yang dibangun, maka dilakukanlah perancangan arsitektur. Perancangan arsitektur yang telah dibuat meliputi beberapa perancangan diantaranya struktur menu, perancangan antarmuka.

#### 3.2.2.2.1. Antarmuka Menu Utama

Perancangan antarmuka menu utama dari *game save our planet from extinction* dapat dilihat pada gambar 3.44.

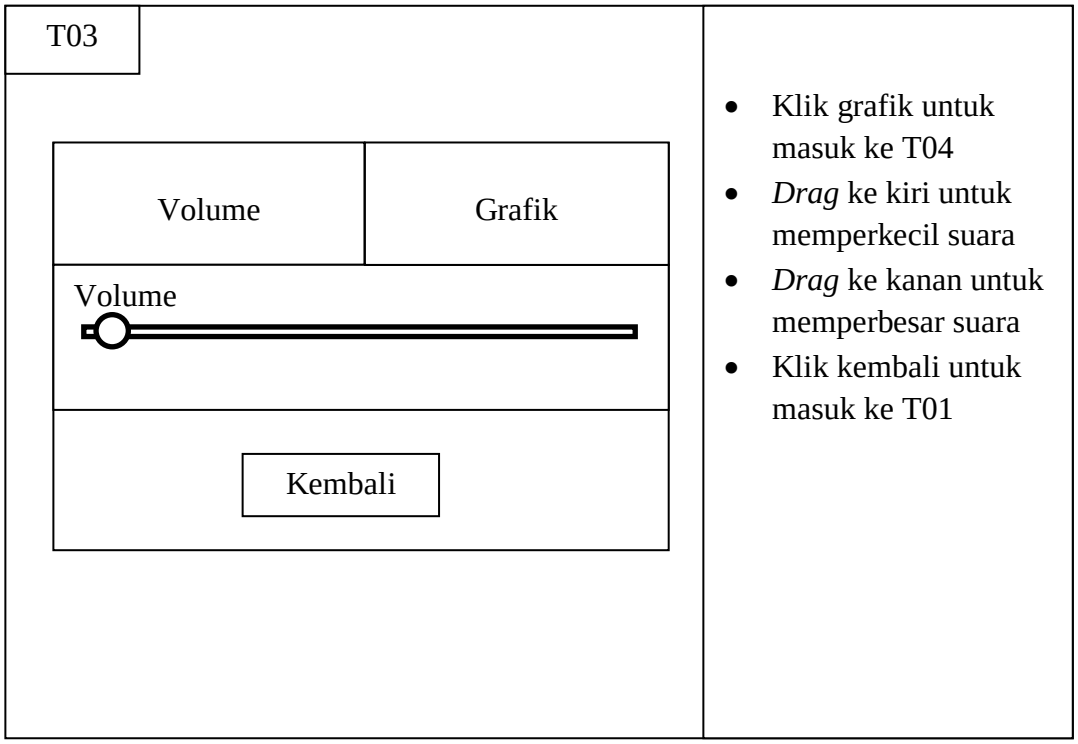



**Gambar 3.44 Antarmuka Tampilan Menu Utama**

**Tabel 3.23 Penggunaan Tombol Menu Utama**

| No | Jenis Objek | Nama Objek | Keterangan |
|---|---|---|---|
| 1 | Button | Mulai | Untuk masuk memulai permainan |
| | | Pengaturan | Untuk menampilkan halaman pengaturan |
| | | Petunjuk | Untuk menampilkan halaman petunjuk |
| | | Keluar | Untuk menutup Aplikasi |

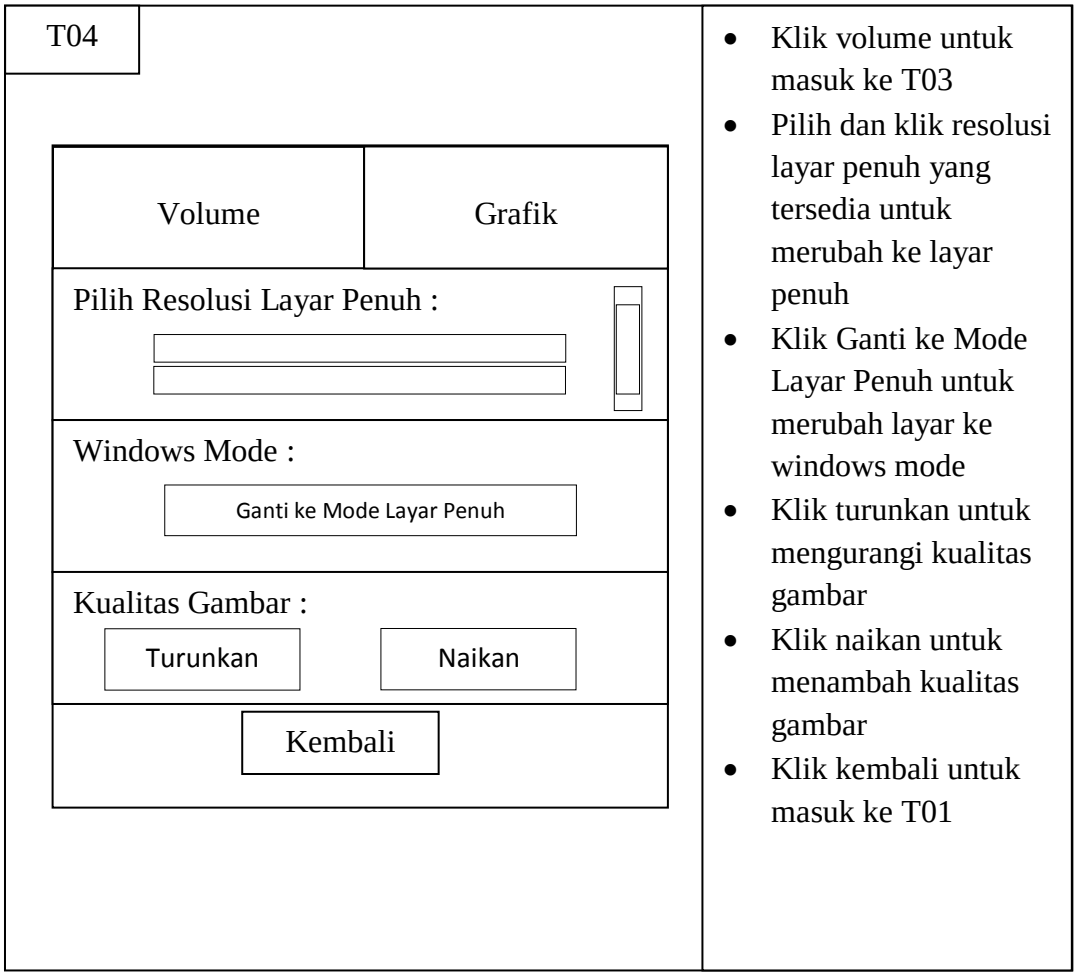
#### 3.2.2.2.2. Antarmuka Pengaturan




**Gambar 3.45 Antarmuka Tampilan Menu Pengaturan Suara**

**Tabel 3.24 Penggunaan Tombol Pengaturan Suara**

| No | Jenis Objek | Nama Objek | Keterangan |
|---|---|---|---|
| 1 | Button | Volume | Untuk pengaturan volume |
| | | Grafik | Untuk pengaturan grafik |
| | | Kembali | Untuk Kembali ke menu utama |



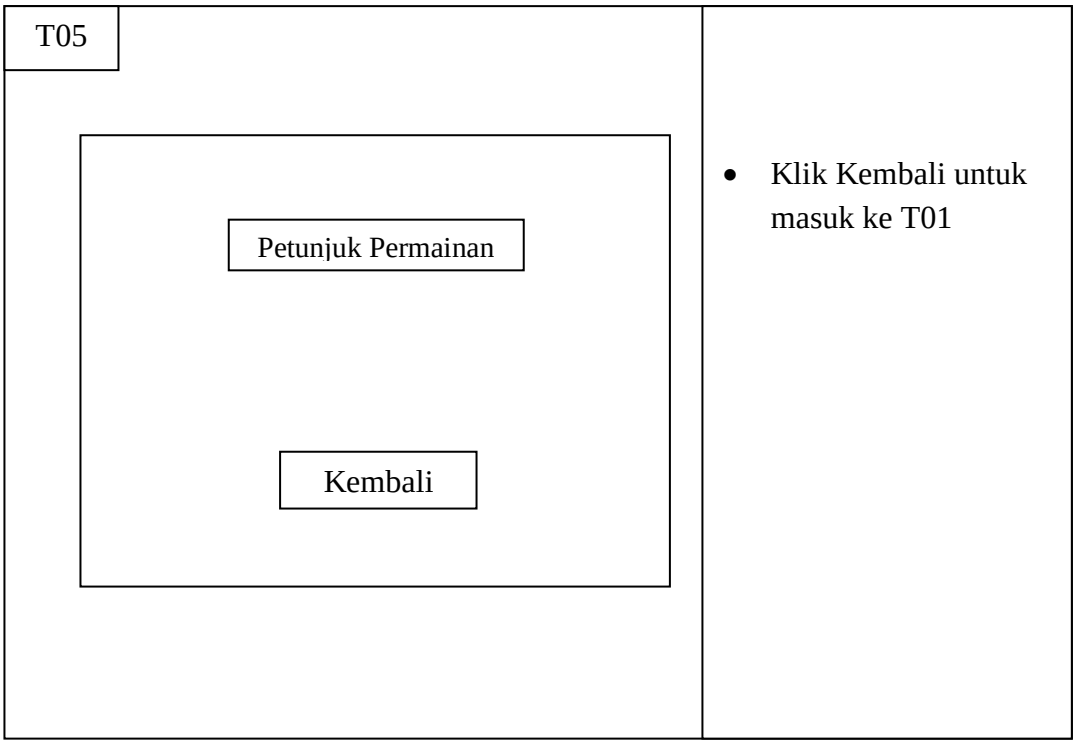

**Gambar 3.46 Antarmuka Tampilan Menu Pengaturan Grafik**

**Tabel 3.25 Penggunaan Tombol Pengaturan Suara**

| No | Jenis Objek | Nama Objek | Keterangan |
|---|---|---|---|
| 1 | Button | Volume | Untuk pengaturan volume |
| | | Grafik | Untuk pengaturan grafik |
| | | Ganti ke Mode Layar Penuh | Untuk Mengganti ke model layar penuh |

| | | | |
|---|---|---|---|
| | | Turunkan | Untuk mengurangi kualitas gambar |
| | | Naikan | Untuk menambah kualitas gambar |
| | | Kembali | Untuk Kembali ke menu utama |

### 3.2.2.2.3. Antarmuka Petunjuk

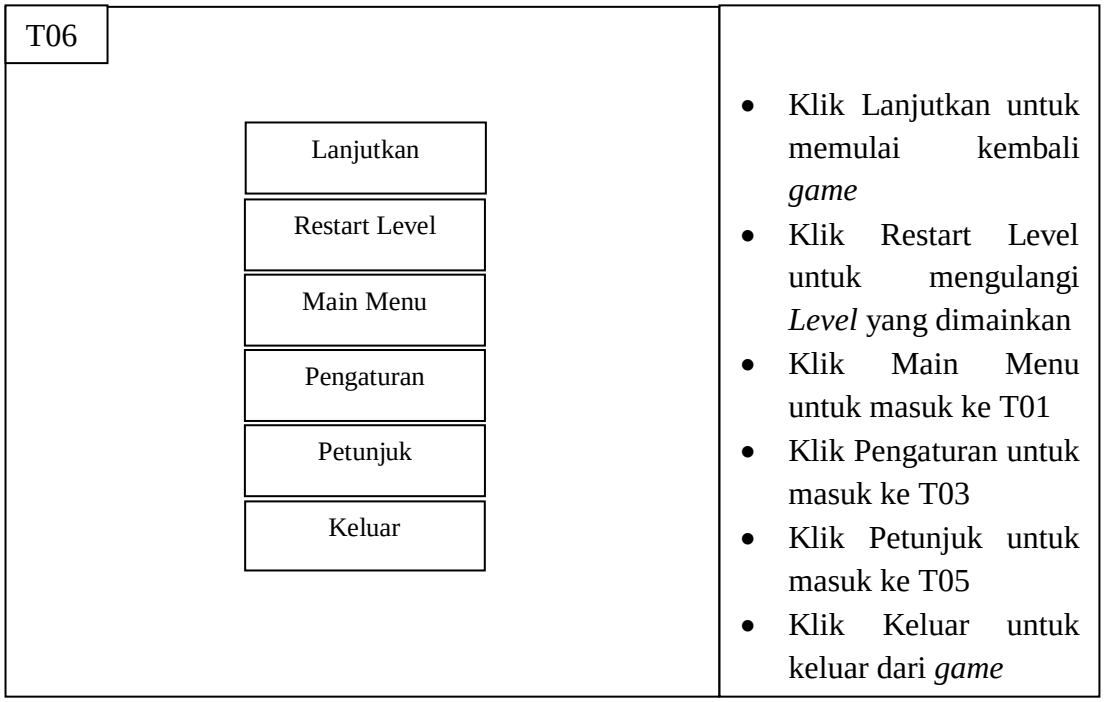



**Gambar 3.47 Antarmuka Tampilan Menu Petunjuk**

**Tabel 3.26 Penggunaan Tombol Petunjuk**

| No | Jenis Objek | Nama Objek | Keterangan |
|---|---|---|---|
| 1 | Button | Kembali | Untuk Kembali ke menu utama |

#### 3.2.2.2.4. Antarmuka Pause Menu



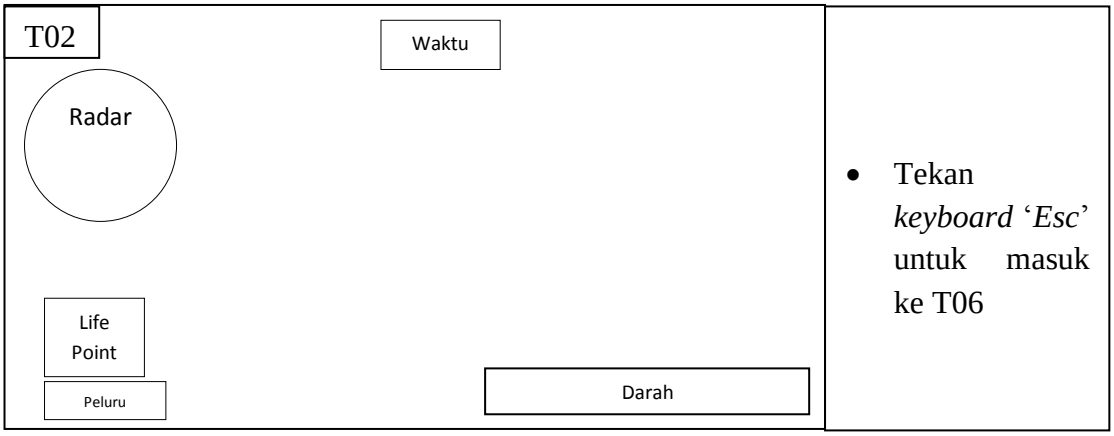

**Gambar 3.48 Antarmuka Tampilan Pause Menu**

**Tabel 3.27 Penggunaan Tombol Pause Menu**

| No | Jenis Objek | Nama Objek | Keterangan |
|---|---|---|---|
| 1 | Button | Lanjutkan | Untuk melanjutkan permainan |
| | | Restart Level | Untuk mengulangi *level game* yang dimainkan |
| | | Main Menu | Untuk kembali ke menu utama |
| | | Pengaturan | Untuk menampilkan pengaturan permainan |

| | | | |
|---|---|---|---|
| | | Petunjuk | Untuk menampilkan petunjuk permainan |
| | | Keluar | Untuk keluar dari permainan |

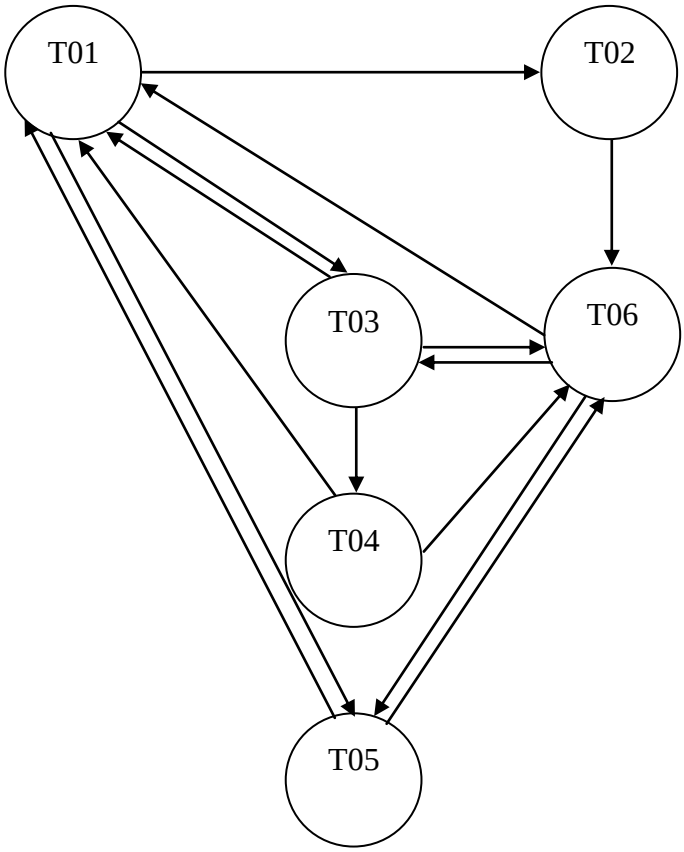
### 3.2.2.2.5. Antarmuka Permainan




**Gambar 3.49 Antarmuka Permainan**

### 3.2.3. Jaringan Semantik

Struktur aplikasi ini dibuat secara moduler, yaitu program dipecah menjadi modul-modul kecil yang mudah dibuat, mudah dites, dan mudah dimodifikasi. Dalam pembuatan *game* ini menggunakan alat bantu bagan struktural yang disebut juga jaringan semantik dapat dilihat pada gambar 3.50.



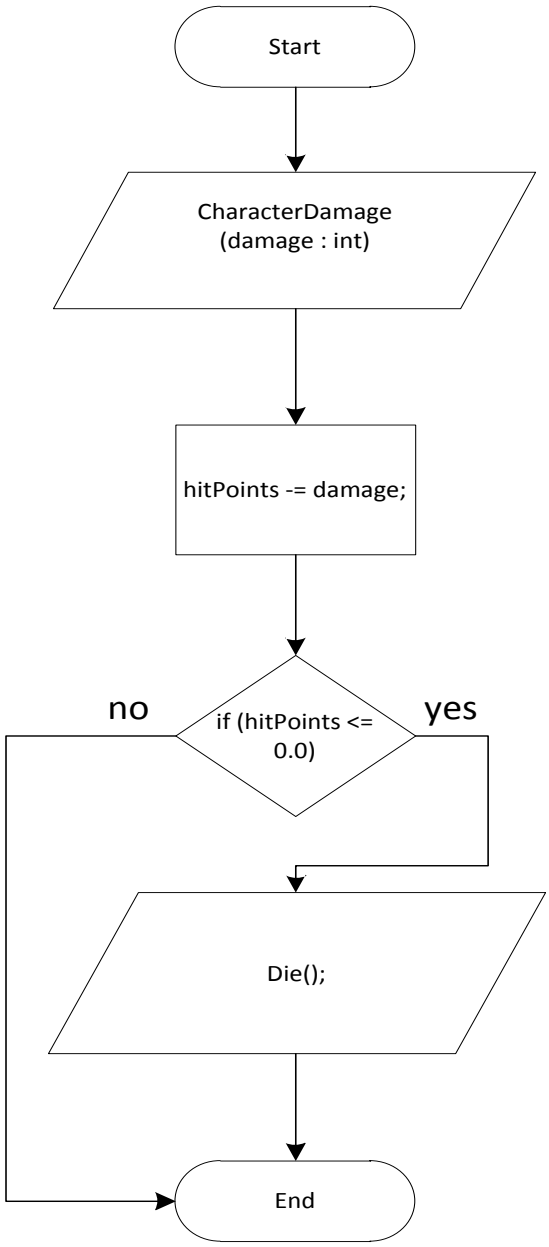

**Gambar 3.50 Jaringan Semantik**

### 3.2.4. Perancangan *Method*

Perancangan *method* merupakan perancangan yang berfungsi untuk mendeskripsikan *method-method* yang berada di dalam aplikasi. *Method* dapat dipanggil dengan menyertakan variabel, baik hanya satu variabel, banyak variabel atau bahkan tidak ada sama sekali. Adapun *method-method* yang terdapat dalam *game save our planet from extinction* yang akan dibangun dijelaskan melalui *flowchart* sebagai berikut.

#### 3.2.4.1. *Method CharacterDamage*

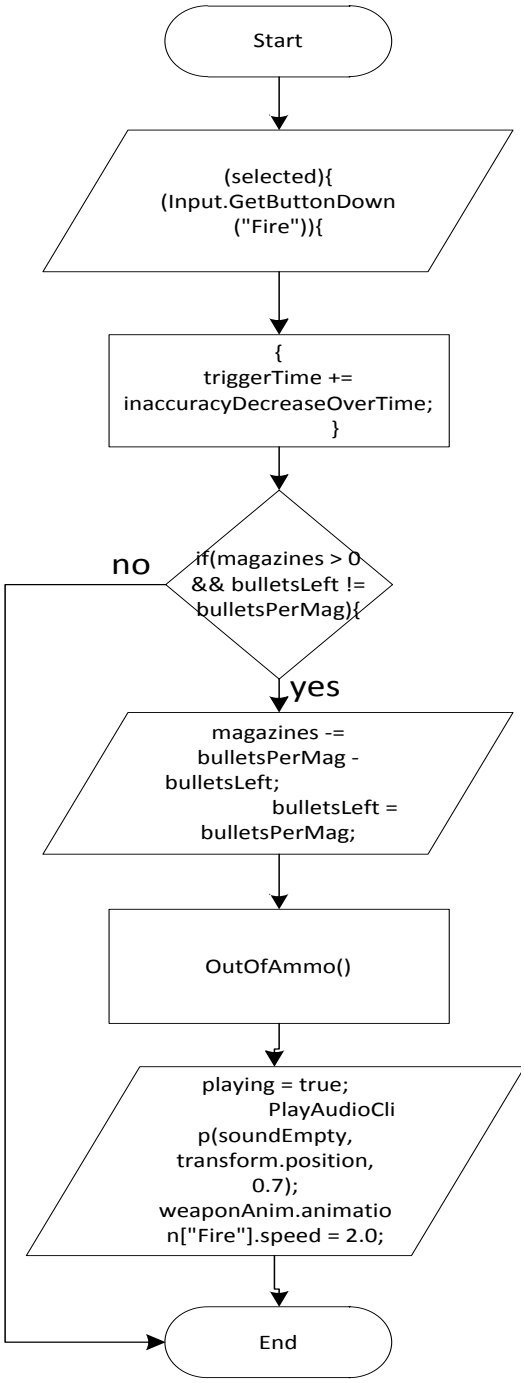
*Method* ini melakukan pengecekan terhadap setiap serangan/*damage* dari musuh yang mengenai pemain seperti pada gambar 3.51.




**Gambar 3.51 *Flowchart* Method *CharacterDamage***

### 3.2.4.2. *Method onGui* (*Weapon*)

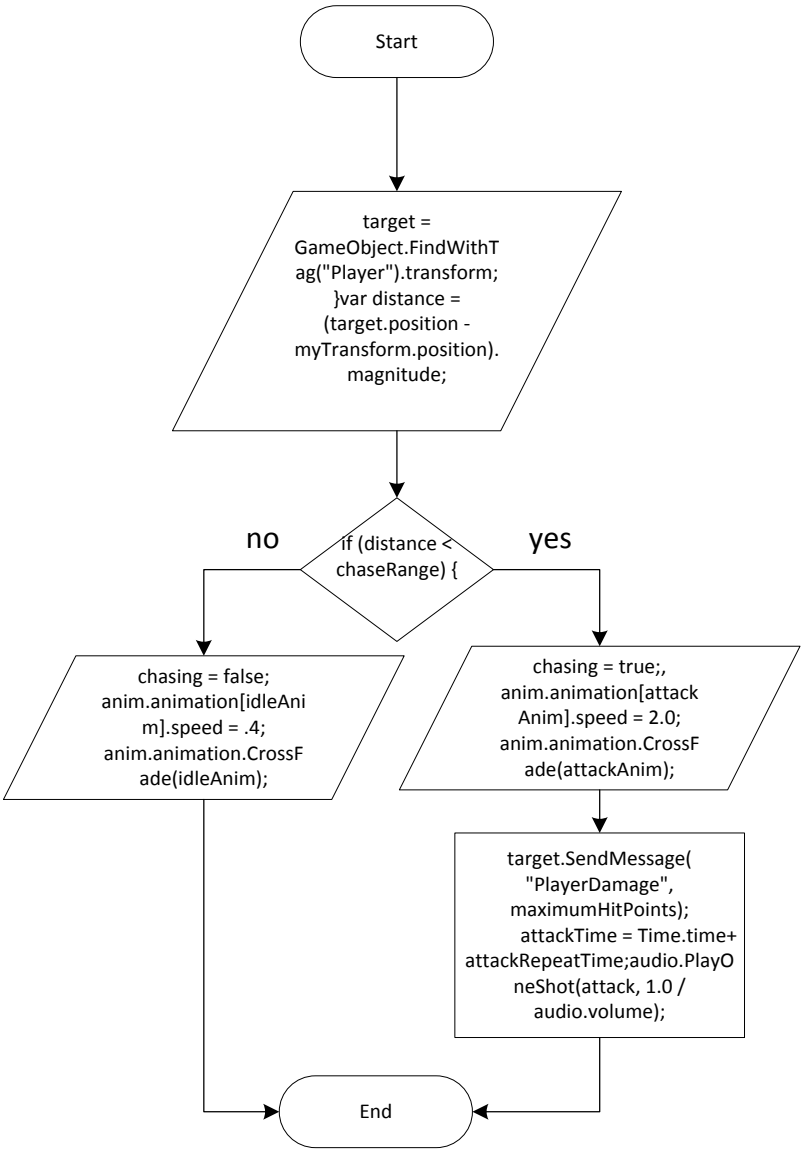
*Method* ini melakukan perubahan terhadap setiap peluru/*ammo* yang digunakan untuk menyerang musuh menggunakan senjata senapan laras panjang seperti pada gambar 3.52.




**Gambar 3.52 *Flowchart Method onGui* (*Weapon*)**

### 3.2.4.3. *Method awake* (AI)

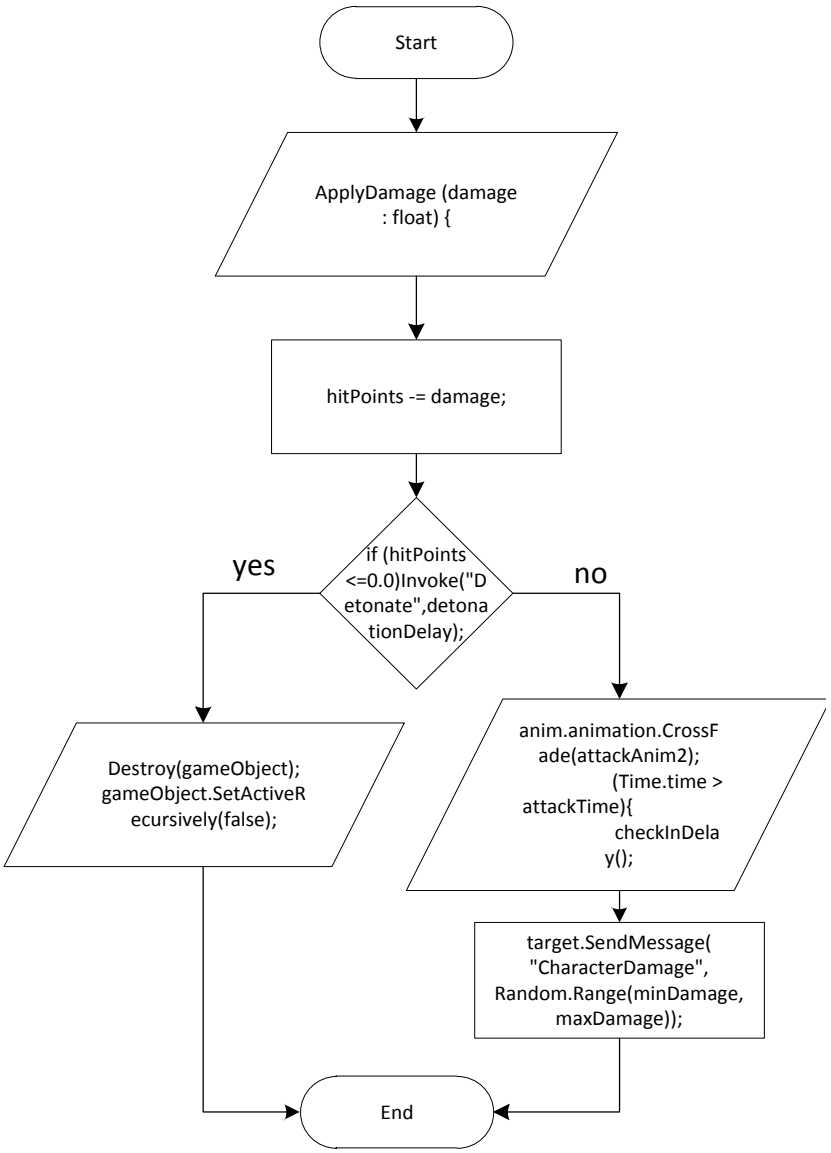
*Method awake* pada AI ini melakukan penyerangan/memberikan damage terhadap target untuk mengurangi jumlah *health point* target sesuai *damage* yang diberikan seperti pada gambar 3.53.




**Gambar 3.53 *Flowchart Method awake*(AI)**

### 3.2.4.4. *Method CharacterDamage.applydamage*()

*Method applydamage* pada EnemyAI ini mendapatkan serangan/menerima *damage* dari target untuk mengurangi jumlah healtpoint sesuai *damage* yang diterima seperti pada gambar 3.54.




**Gambar 3.54** ***Flowchart Method CharacterDamage.applydamage***