BAB III
ANALISIS DAN PERANCANGAN

III.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 dibua.

III.1.1 Analisis Masalah

Seiring dengan berkembangnya peradaban saat ini membuat kebudayaan asing mudah masuk hal ini berdampak kepada berkurangnya kepedulian terhadap pendidikan sejarah. Menariknya budaya masa lalu yang berakibat banyaknya budaya asing yang cenderung lebih menarik sehingga menggeser keberadaan sejarah yang ada [***1***]. Menurut pemerhati budaya dan sejarah Asep Kambali mengatakan jika kesadaran masyarakat Indonesia untuk mengenal dan mencintai sejarah dan budaya, berangsur-angsur mulai meredup menurutnya Kesadaran masyarakat, khususnya pemuda, terhadap budaya dan sejarah dinilai saat ini justru rendah.Kita bisa menilai pertama dari pengetahuan mereka yang kurang terhadap sejarah dan budaya. Kedua, dari cara bersikap. Ketiga, dari cara mereka menghargai dan bertindak terhadap kebudayaan itu sendiri **[*2*]**. Dengan diaplikasikannya ke dalam *game* diharapkan bisa menambah pengetahuan pengguna tentang sejarah khususnya beberapa bangunan bersejarah yang ada di Indonesia seperti Gedung Merdeka, Musium Fatahilah, dan Musium Kereta Api ambarawa, selain itu agar dapat lebih menghargai terhadap sejarah itu sendiri, serta dapat menarik minat para remaja untuk mengunjungi objek-objek wisata yang menyajikan edukasi sejarah.

Berdasarkan uraian masalah dari latar belakang masalah tersebut, maka dapat disimpulkan beberapa masalahnya yaitu :

1. Bagaimana membangun sebuah aplikasi hiburan yang bisa dinikmati oleh para remaja dalam bentuk sebuah *game PC* .
2. Menjadikan *game* sebagai media penyampaian informasi mengenai beberapa bangunan-bangunan bersejarah yang ada di Indonesia

III.1.2 Analisis Game Sejenis

Analisis game sejenis dilakukan untuk memberikan gambaran dari sistem yang akan dibuat yang digunakan sebagai pembanding agar sistem dari *game* yang dibuat semakin baik agar bisa memenuhi keinginan dari pembuat *game*.

III.1.2.1 Soldier Of Fortune payback

Game Soldier Of Fortune Payback ini merupakan game bergenre *first person shooter* yang dibuat oleh Activision dan dirilis pada tanggal 14 Nopember 2007. *Game* ini memiliki grafis yang sangat bagus karena *game* ini ditujukan untuk *console* Xbox 360, Playstation 3, dan PC. Thomas Mason diberikan perintah untuk melaksanakan suatu misi pengawalan dan harus melibatkan dirinya di dalam masalah ketika ada campur tangan kelompok teroris, dan Thomas harus bisa menghentikan kelompok teroris yang bisa menimbulkan kekacauan di seluruh dunia berikut adalah gambar dari *game Soldier Of Fortune : Payback* terlihat pada gambar III.1



Gambar III.1 Screenshot Soldier Of Fortune Payback

Game memiliki grafis dan alur cerita yang cukup baik, namun di dalam *game* terdapat konten-konten yang menunjukan sadisme yaitu apabila *user* menembak kepada musuh pada anggota badan tertentu maka anggota badan yang tertembak akan terpisah dari anggota badan lainnya.

Tabel III.1 analisis game sejenis

Nama *Game*	*Genre Game*	Desain Grafis	*Storyline* dan *gameplay*	*multiplayer*	Basis
Soldier Of Fortune : Payback	*Firrst Person Shooter*	Memiliki gambar 3D yang sangat bagus dan halus serta tidak memiliki spesifikasi yang tinggi	Alur cerita bersifat fiktif bukan diambil dari hal-hal yang pernah terjadi, terdapat konten yang bersifat sadis di dalam game tersebut	tidak	*Desktop*

a. Gameplay

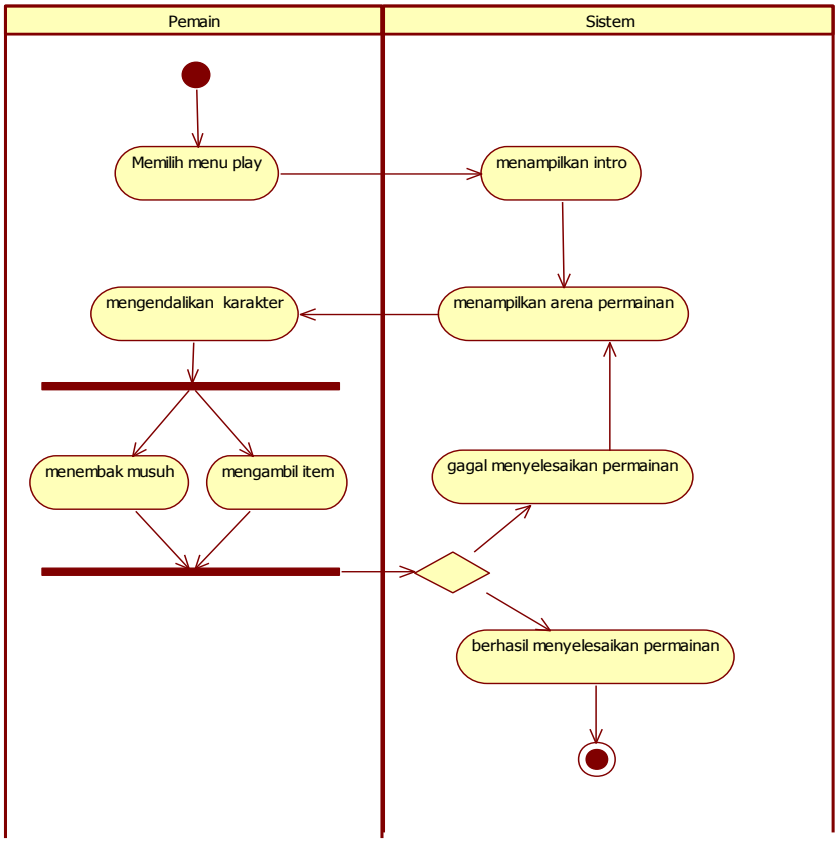
Pada *game Soldier of Fortune*, pemain harus mengalahkan musuh-musuh yang ada di dalam *game*, dan harus sampai ke titik yang telah ditentukan untuk menuju *level* selanjutnya.

b. Skenario *Game*

Skenario dalam permainan *Soldier of Fortune* dapat dilihat alur permainannya sebagai berikut :

a. Pemain berada di sebuah kota yang tandus di tengah gurun
b. Pemain harus mencari sebuah titik yang sudah ditentukan untuk menuju *level* selanjutnya
c. Pemain harus menghindari dan mematikan musuh ketika berada di dalam arena permainan
d. Jika pemain terkena tembakan musuh, maka *health point* akan berkurang dan apabila habis maka permainan diulangi ke awal permainan.

Alur sistem dari *game Soldier of Fortune* ini dapat digambarkan melalui *activity diagram* yang terlihat pada gambar III.2.



Gambar III.2 Alur permainan pada *game Soldier of Fortune*

Game Soldier of Fortune memiliki beberapa keterbatasan yaitu,

1. *Game* ini memiliki sifat sadisme karena banyaknya darah yang diperlihatkan pada game ini.
2. *Game* ini hanya bisa dimainkan oleh satu pemain saja.

Disamping ada beberapa keterbatasan yang dimiliki *game Soldier of Fortune* ada juga beberapa kelebihan yang dimiliki yaitu ;

1. Konten yang disajikan cukup lengkap di dalam game ini
2. Memiliki grafis yang sangat baik
3. Memiliki banyak senjata

III.1.3 Analisis *game* yang dibangun

Game yang akan dibangun berjudul *Takedown The Terrorists*. *Game* ini bergenre *first person shooter, Game* ini memiliki grafis 3D (3 Dimensi). Memiliki alur cerita tentang perlawanan terhadap kelompok teroris.

Game dibangun dengan konsep sudut pandang orang pertama agar permainan yang ditampilkan terkesan lebih nyata, seolah-olah pemain terlibat langsung ke dalam arena permainan

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

1. Grafik *game* 3D
2. *Game* ini memiliki 3 buah *level*
3. *Game* ini dibuat dengan genre *first person shooter*
4. *Game* dapat dihentikan ketika permainan sedang berlangsung
5. Memiliki objek-objek gedung bersejarah dimana info dari objek gedung tersebut bisa ditampilkan pada saat permainan berlangsung
6. *Game* ini memiliki pause menu
7. *Game* dilengkapi dengan menu option untuk pengaturan suara dan resolusi gambar

III.1.4 Storyline

Cerita dimulai tentang maraknya kasus terrorisme yang terjadi di Indonesia dan juga banyaknya kelompok teroris yang ada di Indonesia. Seorang kapten tim dari sebuah pasukan khusus yang bernama Kapten dari pasukan densus 88 yang ditugaskan untuk menangkap kelompok-kelompok teroris di yang ada di Indonesia.

- *Level* pertama misi utama di *level* ini ialah *player* harus menuju mobil pasukannya yang telah menunggu untuk menuju kota selanjutnya di dalam *level* 1 petunjuk keberdaan musuh akan ditunjukkan oleh pasukan yang ada di dalam arena *level* 1. Gedung-gedung yang memiliki sejarah di *level* ini adalah masjid agung bandung, gedung merdeka, gedung savoy homann. Pada *level* ini memiliki tingkat kesulitan mudah dengan *NPC* yang sedikit.

- *Level* kedua misi utama dari *level* ini harus mencari pasukan-pasukan yang diculik oleh para teroris. Karakter harus mengalahkan teroris yang akan menghadang dan membebaskan pasukan yang sedang ditawan oleh teroris. Pada *level* ini karakter akan dihadapkan dengan jumlah musuh yang semakin banyak daripada *level* sebelumnya, petunjuk akan diberikan oleh warga sipil yang akan memberitahu keberadaan player dan karakter akan mendapat NPC bantuan yang membantu karakter. Gedung-gedung bersejarah di *level* ini adalah Bank Indonesia, Stasiun Jatinegara, dan Musium Fatahilah.
- *Level* ketiga, karakter harus menemukan bom yang disembunyikan oleh para teroris untuk menyelesaikan misi terakhir ini. Karakter tidak akan mendapat petunjuk apapun tentang lokasi disimpannya bom-bom tersebut. *Level* ini karakter akan menghadapi musuh yang jauh lebih banyak dan lebih kuat daripada level sebelumnya. Gedung-gedung bersejarah di *level* ini adalah Taman Sari, Lawang Sewu, dan Stasiun Ambarawa.

III.1.5 Gameplay

Misi utama dari game ini adalah mengagalkan rencana terorisme terhadap gedung-gedung bersejarah yang ada di Indonesia sekaligus memberantas komplotan teroris yang telah melakukan teror terhadap masyarakat. Menu utama memiliki beberapa menu yaitu Main, petunjuk, *option*, dan keluar. Di dalam game ini terdapat tiga buah *level*, dua *level* semua *level* memilik *setting* di perkotaan.

Semakin banyak *level* yang dilewati maka semakin sulit musuh yang dihadapi pada setiap *level* player akan mendapatkan beberapa *item* yaitu berupa darah dan amunisi senjata, pada setiap *level* player akan menjumpai gedung-gedung bersejarah, informasi gedung tersebut akan tampil jika aktor utama mendekati objek gedung tersebut.

Senjata yang terdapat di dalam game berupa M4A1 dengan *grenade launcher,*dan *Sniper Dragunov*.

III.1.6 Analisis Tingkat Kesulitan

Tingkat kesulitan akan berbeda di setiap levelnya dilihat dari jumlah musuh, kekuatan musuh, dan selang waktu setiap tembakan yang dilakukan musuh. Di bawah ini adalah perbandingan tingkat kesulitan tiap *level*, dapat dilihat pada tabel III.2

Tabel III.2 Tingkat Kesulitan tiap *Level*

Keterangan	***Level 1***	***Level 2***	***Level 3***
Jumlah musuh	10	*15*	*20*
Jarak tembak musuh	*50*	*75*	*100*
Jarak Pencarian *Player*	*100*	*125*	*150*

III.1.7 Analisis Scoring

Skor yang ada di dalam game Takedown the terrorists ini dilakukan dengan cara menghitung jumlah musuh yang mati akan dikalikan 100 poin skor ti saling berhubungan tiap levelnya.

III.1.8 Analisis Algoritma A star

Algoritma *A** di dalam *game* akan diterapkan pada NPC (*Non Playable Character*) sebagai penentuan jalur tependek atau *pathfinding* menuju *target* yaitu *player* Berikut ini adalah *pseudocode* dari algoritma *A**.

Tabel III.3 Pseudocode Algoritma A*

```
1. public int f {
2. get {
3. return g+h;
4. }
5. }
6. public
7. virtual
8. void UpdateH (Int3 targetPosition, Heuristic
9. heuristic, float scale) {
10. if (heuristic == Heuristic.None) {
11. h = 0;
12. return;
13. }
14. if (heuristic == Heuristic.Manhattan) {
15. h = Mathfx.RoundToInt  (
16.(Abs (position.x-targetPosition.x) +
17. Abs (position.y-targetPosition.y) +
18. Abs (position.z-targetPosition.z))
19. * scale
20. );
21. } else if (heuristic ==
22. Heuristic.DiagonalManhattan) {
23. int xDistance = Abs (position.x-targetPosition.x);
24. int zDistance = Abs (position.z-targetPosition.z);
25. if (xDistance > zDistance) {
26. h = (14*zDistance + 10*(xDistance-zDistance))/10;
27. } else {
28. h = (14*xDistance + 10*(zDistance-xDistance))/10;
29. }
30. h = Mathfx.RoundToInt (h * scale);
31. } else {
32. h = Mathfx.RoundToInt ((position-
33. targetPosition).magnitude*scale);
34. }
35. }
```

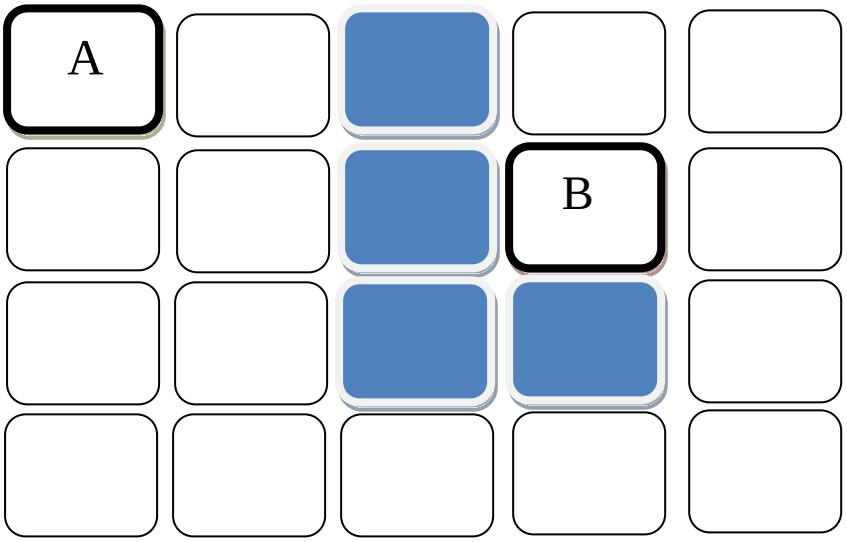
Kamus global yang digunakan ditampung dalam *class node* yang terdapat pada *script astarclasses*.

{Kamus Global}
public class Node
type:
{
public Int3 position;
public Node parent;
public int pathIDx;

```
public static Path activePath;
public int g;
public int h;
public int cost;
public int penalty;
public int tags = 1;
}
public Node[] connections;
public int[] connectionCosts;
```

script di atas merupakan kamus global yang akan dimasukkan ke dalam banyaknya *node* yang saling berhubungan, juga untuk menentukan nilai dari tiap-tiap *node*

Cara kerja algoritma *A** di dalam *game* lebih jelasnya kita lihat pada gambar III.3



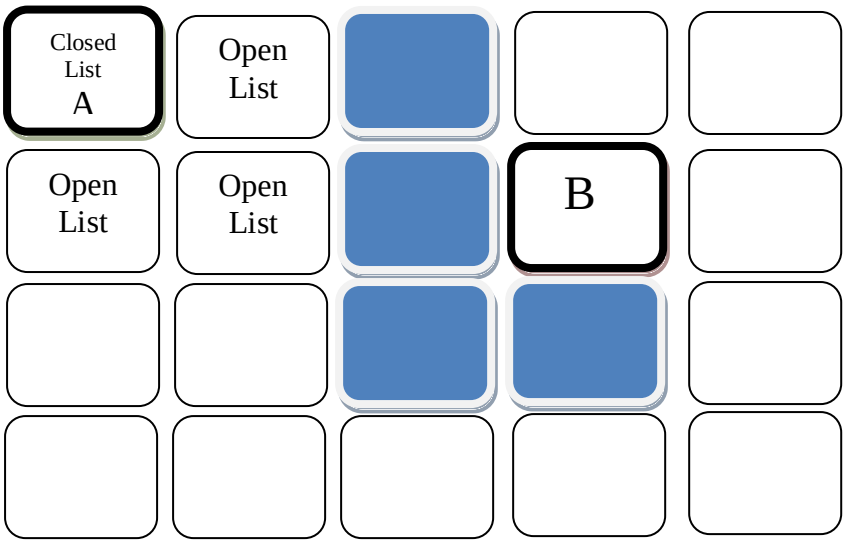
Gambar III.3 Tahap awal pencarian Algoritma A*

Pada gambar III.3 pencarian wilayah dibagi ke dalam *node-node* yang terdiri dari node berwarna hijau merupakan node awal (node A), *node* berwarna merah merupakan *node* akhir (node B) dan yang berwarna biru 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 biru 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 III.4



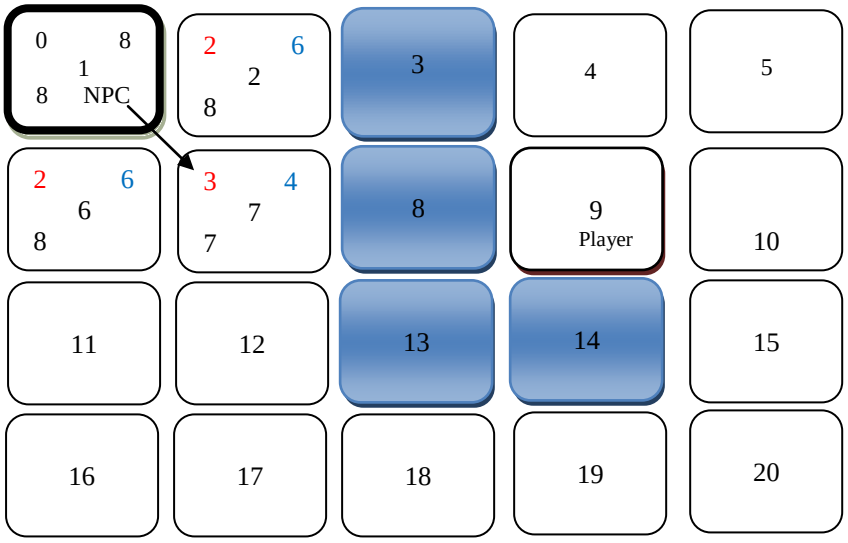
Gambar III.4 Ilustrasi *open list* dan *closed list*

Pada gambar III.4 diatas *node* awal yang berwarna hijau 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 disekitarknya. 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 III.5.



Gambar III.5 Perhitungan menentukan nilai F

Gambar III.5 menunjukkan pada *open list* dari *node* awal yaitu NPC nilai F yang dihasilkan adalah 4 yang didapatkan dari menambahkan nilai G = 2 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 dirumuskan dengan:

H = 2*(abs(n.x-tujuan.*x*) + abs(n.y-tujuan.y))

Sebagai contoh *node* yang terletak disebelah kanan dari *node* awal diberikan nilai H = 6 yang didapatkan dari perhitungan 6 *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.

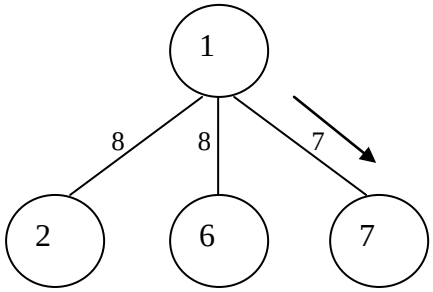
1. Hapus *current node* dari *open list* dan masukkan ke dalam *closed list.*
2. Periksa semua *node* yang berdekatan dengan *current node*, abaikan *node* penghalang. Jika tidak ada pada *open list,* tambahkan ke dalam *open list*, lalu tambahkan "*parent*" untuk *node* tersebut.
3. Jika *node* sudah 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 kita 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 III.3

Tabel III.4 Tabel Perhitungan Langkah Pertama

Open List	2, 6, 7
Closed List	1
Parent	1
Current Node	7

Dibawah ini adalah penjelasan langkah pertama algoritma *astar* dengan menggunakan *tree*

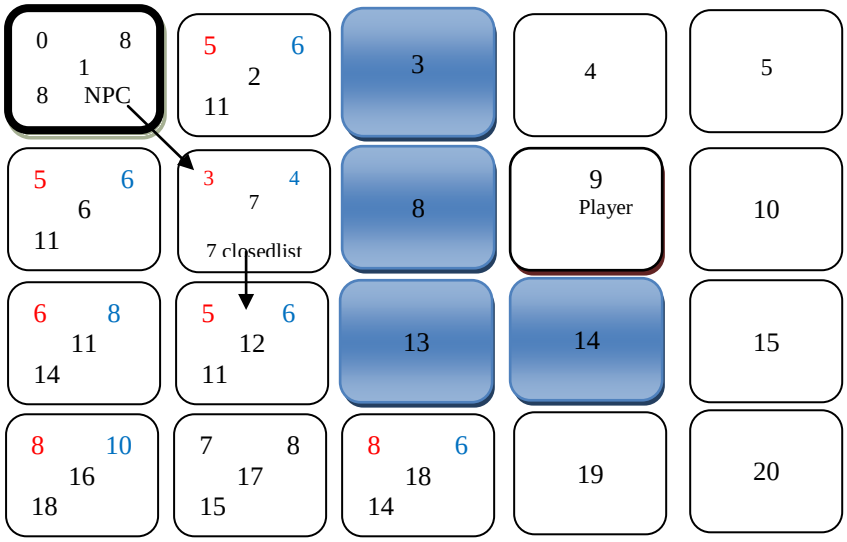


Gambar III.6 Penjelasan A* menggunakan *tree* pada langkah pertama

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 menemuka *node* dengan nilai F yang paling rendah yang selanjutnya akan dijadikan *current node* (1,1), 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



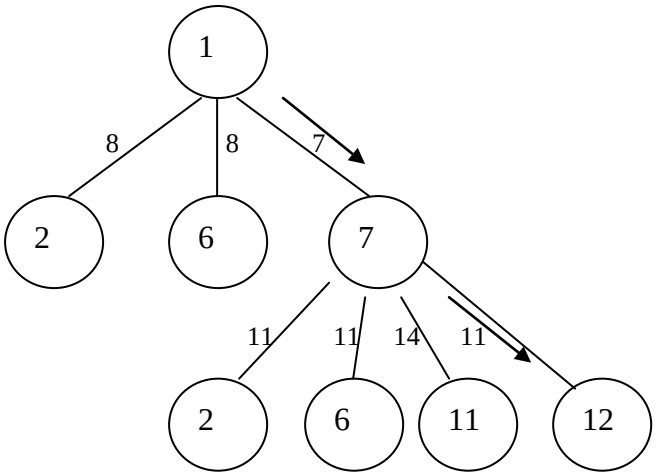
Gambar III.7 Penjelasan Astar pada langkah kedua

Tabel III.5 menunjukan penjelasan tentang pencarian jalur terpendek untuk langkah kedua dapat dilihat pada tabel III.5

Tabel III.5 Perhitungan Langkah Kedua

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

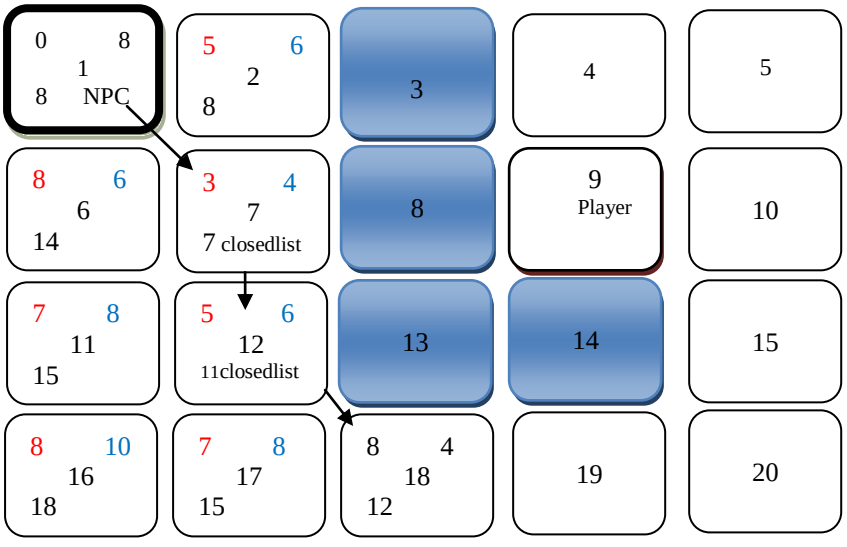
Gambar III.8 menunjukkan penjelasan melalui *tree* untuk pencarian jalan terpendek pada langkah kedua



Gambar III.8 Penjelasan Astar dengan tree pada langkah kedua

Pada langkah kedua terdapat node yang memiliki nilai F yang sama pada kasus seperti ini kita dapat memilih jalur yang mana saja tinggal kita menyesuaikan dengan node mana yang mendekat menuju 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 III.9



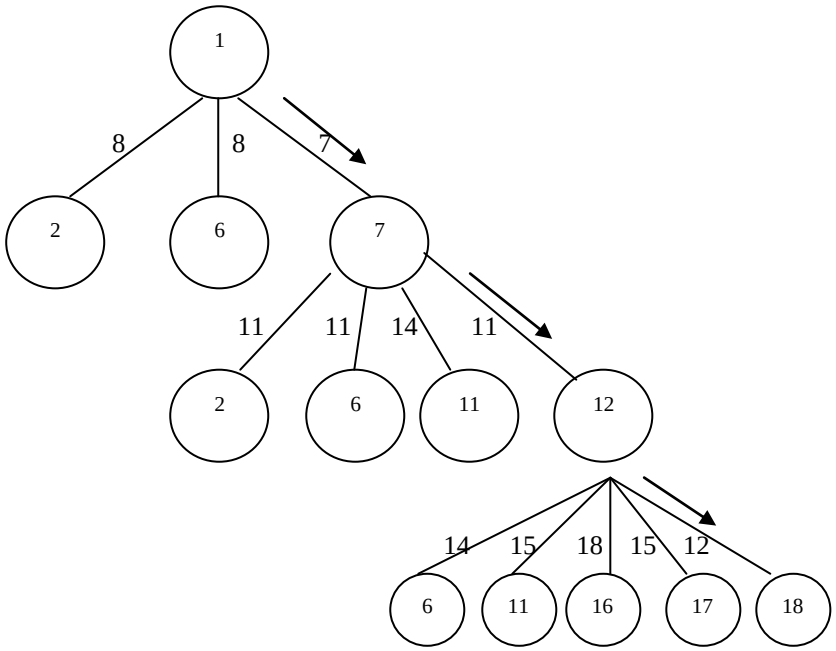
Gambar III.9 Penjelasan Astar pada langkah ketiga

Tabel III.6 menunjukan penjelasan tentang pencarian jalur terpendek untuk langkah kedua dapat dilihat pada tabel III.6

Tabel III.6 Tabel Perhitungan langkah ketiga

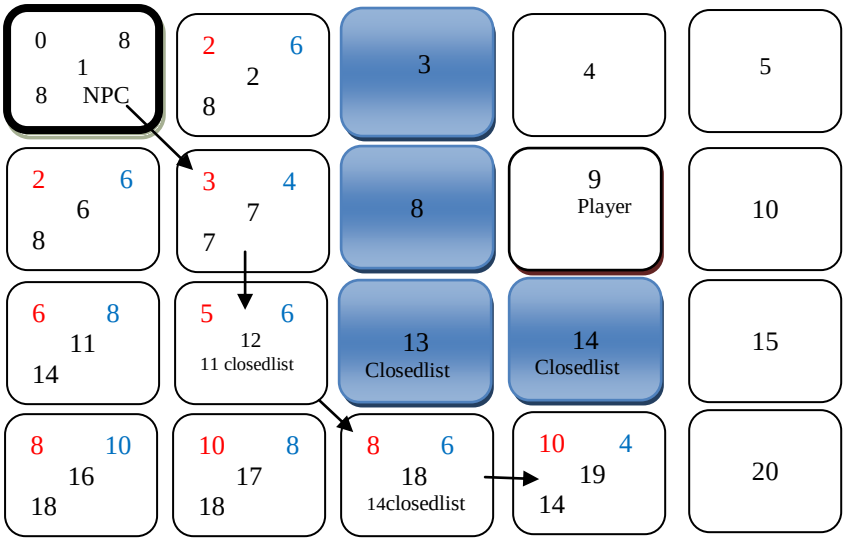
Open List	6, 11, 16, 17, 18
Closed List	12, 13, 8, 7
Parent	12
Current Node	18

Gambar III.10 menunjukkan penjelasan melalui *tree* untuk pencarian jalan terpendek pada langkah ketiga.



Gambar III.10 Penjelasan Astar dengan tree pada langkah ketiga

Perhitungan langkah keempat yang berada pada *node* dengan indeks 18 dapat dilihat pada gambar III.11



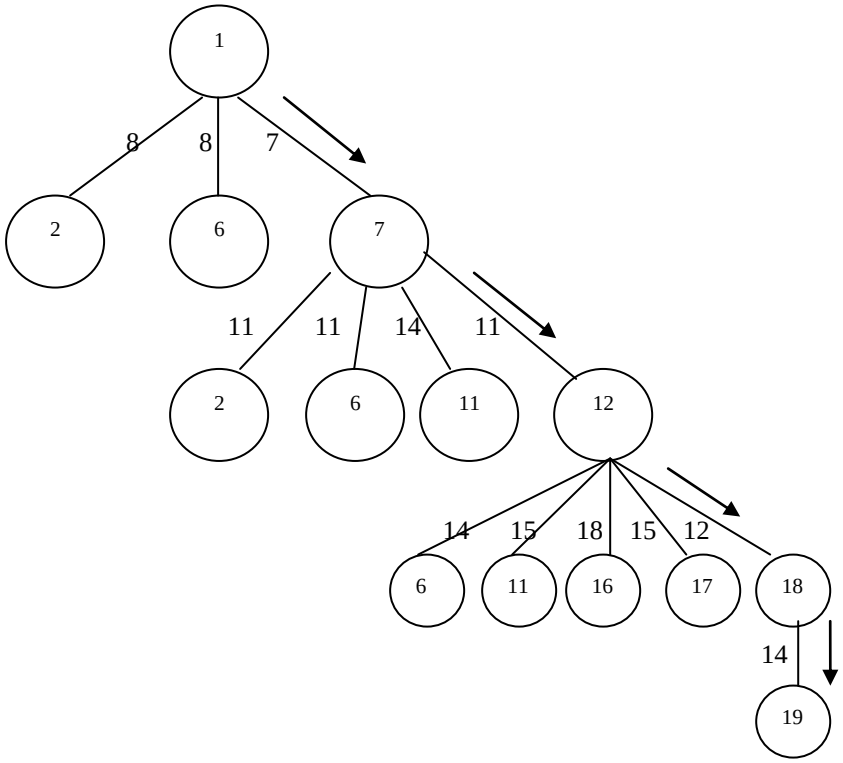
Gambar III.11 penjelasan astar pada langkah keempat

Tabel III.7 menunjukan penjelasan tentang pencarian jalur terpendek untuk langkah kedua dapat dilihat pada tabel III.7

Tabel III.7 Perhitungan langkah keempat

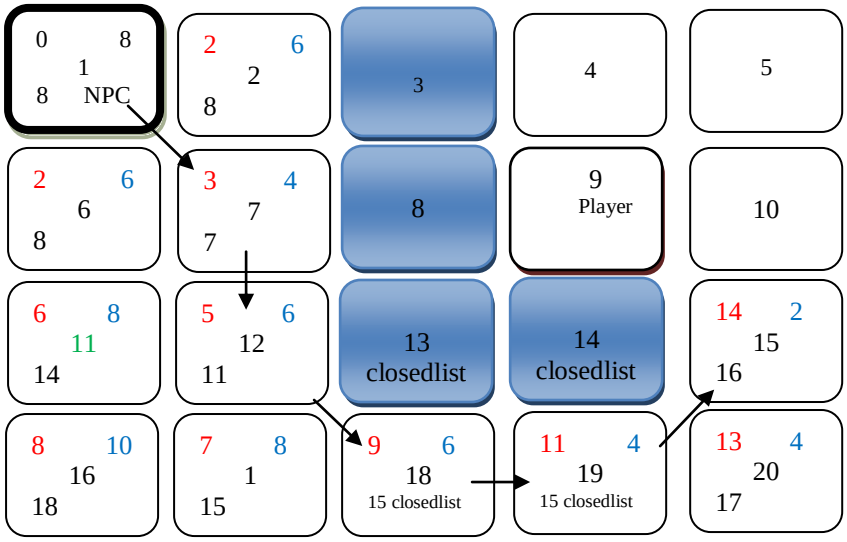
Open List	17, 19
Closed List	12, 13, 14, 18
Parent	18
Current Node	19

Gambar III.12 menunjukkan penjelasan melalui *tree* untuk pencarian jalan terpendek pada langkah keempat.



Gambar III.12 Penjelasan Astar dengan tree pada langkah keempat

Perhitungan pada langkah kelima yang berada pada *node* dengan indeks 15 yang memiliki nilai F = 16 bisa dilihat pada gambar III.13



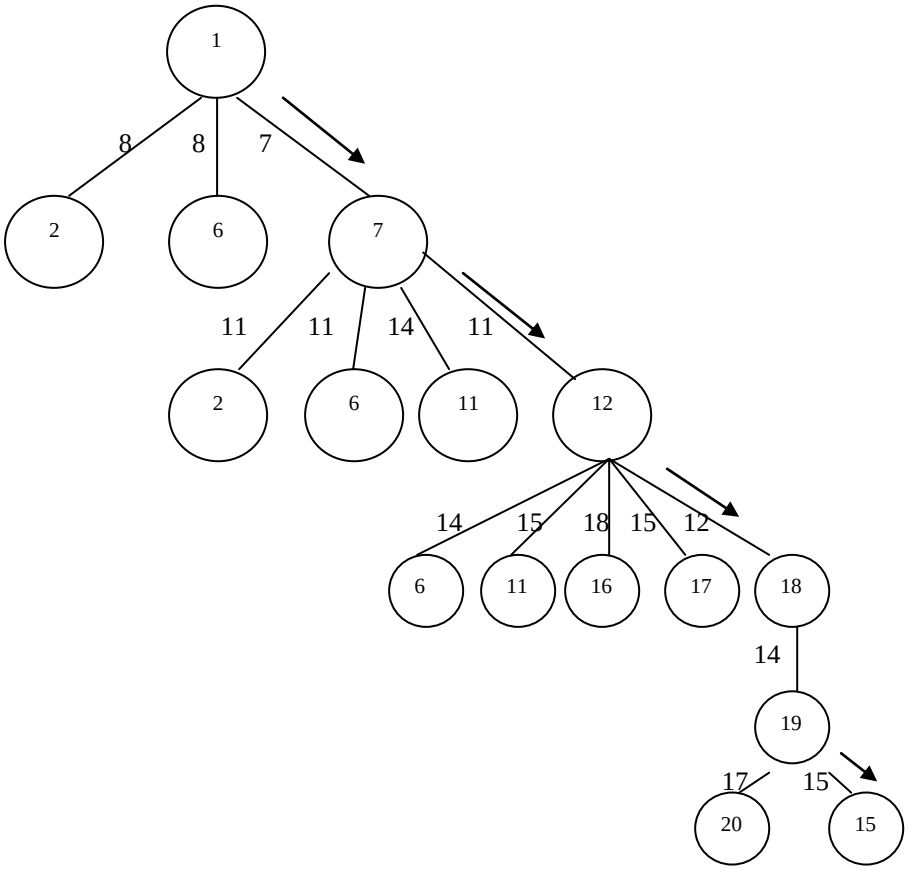
Gambar III.13 Penjelasan astar pada langkah kelima

Pada tabel III.8 penjelasan melalui tabel untuk pencarian jalur terpendek pada langkah keenam.

Tabel III.8 Tabel perhitungan langkah kelima

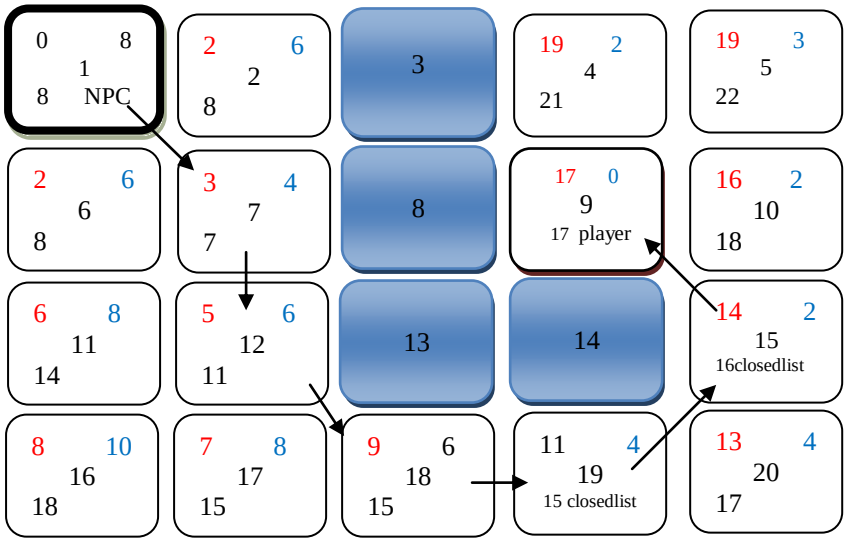
Open List	15, 20
Closed List	13, 14, 18
Parent	19
Current Node	15

Gambar III.14 menunjukkan penjelasan melalui *tree* untuk pencarian jalan terpendek pada langkah kelima.



Gambar III.14 Penjelasan Astar dengan tree pada langkah kelima

Perhitungan pada langkah keenam yang berada pada *node* dengan indeks ke 15 yang memiliki nilai F = 16 bisa dilihat pada gambar III.15



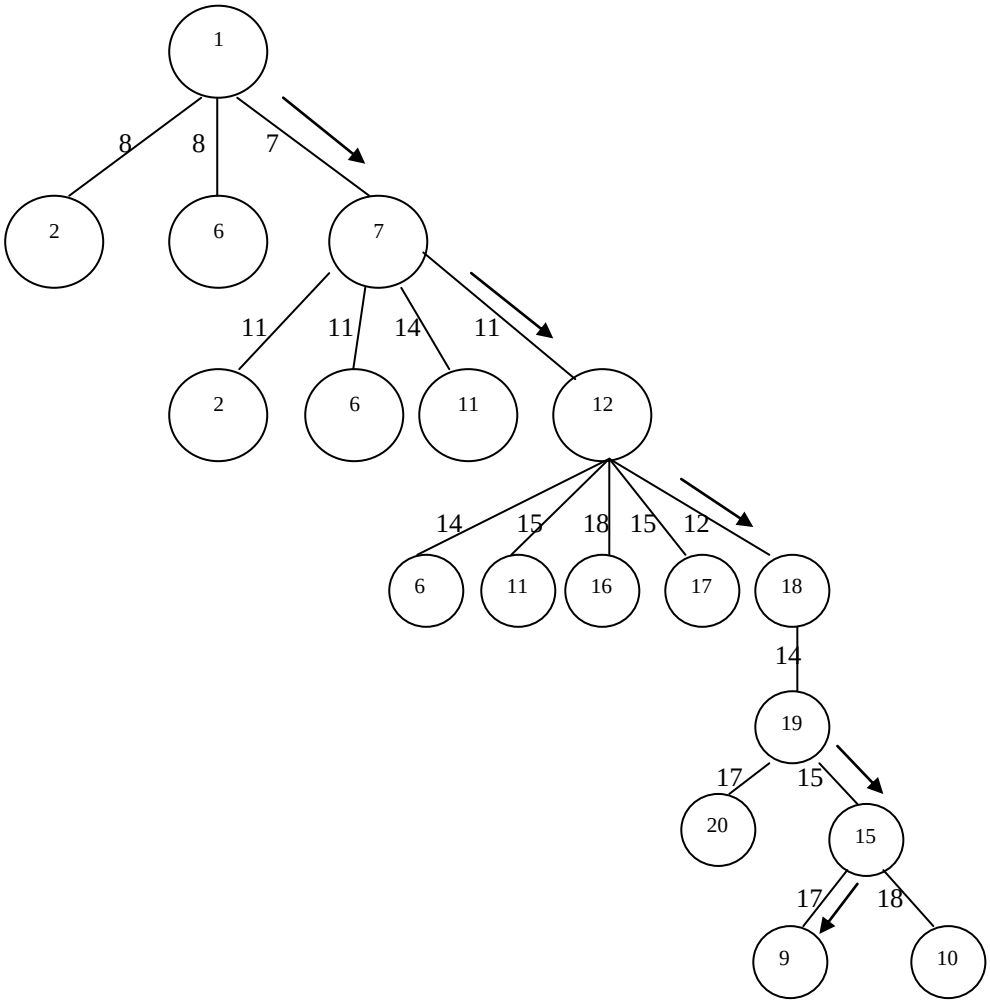
Gambar III.15 Penjelasan astar pada langkah keenam

Pada tabel III.9 penjelasan melalui tabel untuk pencarian jalur terpendek pada langkah keenam.

Tabel III.9 Perhitungan langkah keenam

Open List	9, 10
Closed List	19, 14
Parent	15
Current Node	9

Gambar III.16 menunjukkan penjelasan melalui *tree* untuk pencarian jalan terpendek pada langkah keenam.



Gambar III.16 Penjelasan Astar dengan tree pada langkah keenam

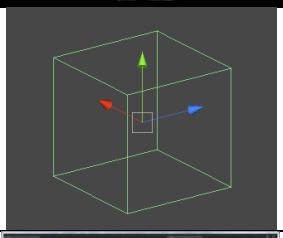
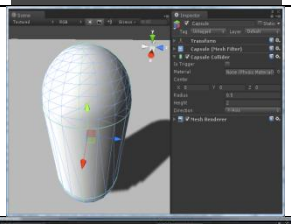
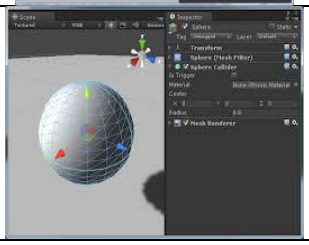
Karena *node* telah sampai ke tujuan (*player*) maka pencarian dihentikan lalu hubungkan *list* yang telah tersusun (indeks 19) dengan *parent* dari *node* tujuan (indeks 15) yang sebelumnya indeks 9 telah dimasukkan ke dalam *closed list*.

III.1.9 Analisis *Collision Detection*

Collision detection adalah pendektesian 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 colider yang digunakan di

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

Tabel III.10 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

III.2 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional dilakukan untuk mengetahui spesifikasi kebutuhan untuk sistem. Spesifikasi kebutuhan melibatkan analisis perangkat keras/*hardware*, analisis perangkat lunak/*software*, analisis pengguna/*user*.

III.2.1 Analisis Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak digunakan untuk mengetahui perangkat lunak apa saja yang akan diperlukan dalam proses pembuatan *game* dan proses implementasi program game bisa dilihat pada tabel III.11

Tabel III.11 Spesifikasi Perangkat Lunak

Nama Perangkat Lunak	Jenis
Sistem Operasi	Microsoft Windows 7
IDE	Unity Developer
Animasi	3ds Max 2009
Design	Adobe Photoshop CS 5

III.2.2 Analisis Kebutuhan Perangkat Keras (*Hardware*)

Spesifikasi perangkat keras yang digunakan untuk membuat *game takedown the terrorists* dan spesifikasi perangkat keras yang diperlukan untuk menjalankan *game takedown the terrorists.* Tabel III.12 menunjukan perangkat keras yang digunakan untuk membuat *game takedown the terrorists*

Tabel III.12 Spesifikasi Perangkat Keras Pembuat

Nama Perangkat Keras	Spesifikasi minimum
Processor	AMD C60, Dual Core
Memory	2 GB
Harddisk	320 GB
VGA	ATI Radeon HD 6290, 256 MB

Menjalankan *game takedown the terrorists* dibutuhkan spesifikasi minimal perangkat untuk user agar bisa menjalankan *game takedown the terrorists* dengan baik. Tabel III.13 menunjukan spesifikasi kebutuhan perangkat keras agar bisa menjalankan *game takedown the terrorists.*

Tabel III.13 Spesifikasi Minimum Perangkat Keras User

Nama Perangkat Keras	Spesifikasi minimum
Processor	Intel Dual Core
Memory	512 MB
Harddisk	2 GB
VGA	ATI Radeon, GeForce 512 MB

Kebutuhan spesifikasi minimum perangkat keras diperlukan untuk mengimpelementasikan game yang sudah dibangun apakah perangkat keras yang digunakan sudah memiliki standar atau tidak, jika tidak maka aplikasi tidak akan berjalan dengan baik biasanya ditandai dengan gerakan game yang melambat dikarenakan perangkat keras yang digunakan tidak memenuhi standar minimum dari game tersebut, jika spesifikasi perangkat keras sudah memenuhi spesifikasi minimumnya maka game akan berjalan dengan baik

III.2.3 Analisis Pengguna (Pemain)

Pada analisis pengguna ini akan mencakup beberapa parameter terhadap calon pengguna dari aplikasi ini

III.2.3.1 *User Knowledge and Experience*

Game takedown the terrorists ini hanya diperuntukan untuk kalangan remaja saja selain tahu bagaimana cara memainkannya juga bisa menyaring hal-hal apa saja yang boleh diambil dan tidak dari game tersebut tabel III.14 menjelaskan anasisis dari *user knowledge and experience*

Tabel III.14 user and knowledge experience

Educational *level*	**Reading *level***	**Typing Skills**
Hanya bisa digunakan oleh kalangan remaja saja yaitu 13 tahun ke atas	Reading *level* yang dimiliki *user* yaitu sedang	Tidak perlu memerlukan *typing skills* yang tinggi yaitu di *level* rata-rata 40 kata permenit
Computer Literacy	**Task Experience**	**System Experience**
Moderate (menengah)	Bisa digunakan oleh pengguna yang bisa menggunakan komputer dengan tingkat sedang	Bisa digunakan oleh pengguna yang bisa menggunakan komputer dengan tingkat sedang
Application Experience	***Native Language***	***Use Of Other System***
Bisa digunakan dalam sistem operasi windows	Menggunakan bahasa Indonesia	Bisa dijalankan tanpa perlu meng-install aplikasi lain

III.2.3.2 *User Physical Characteristic*

Ada hal-hal yang harus diperhatikan dari fisik karakteristik *user* untuk dapat menggunakan aplikasi ini seperti yang terlihat pada tabel III.15.

Tabel III.15 Analisis *User Physical Characteristic*

Age	13 tahun ke atas
Gender	Pria dan Wanita
Handedness	Kanan dan kiri
Color Blind	User yang buta warna mampu memainkan aplikasi ini, karena tidak ada fungsi warna yang dipakai untuk membuat *game* ini.

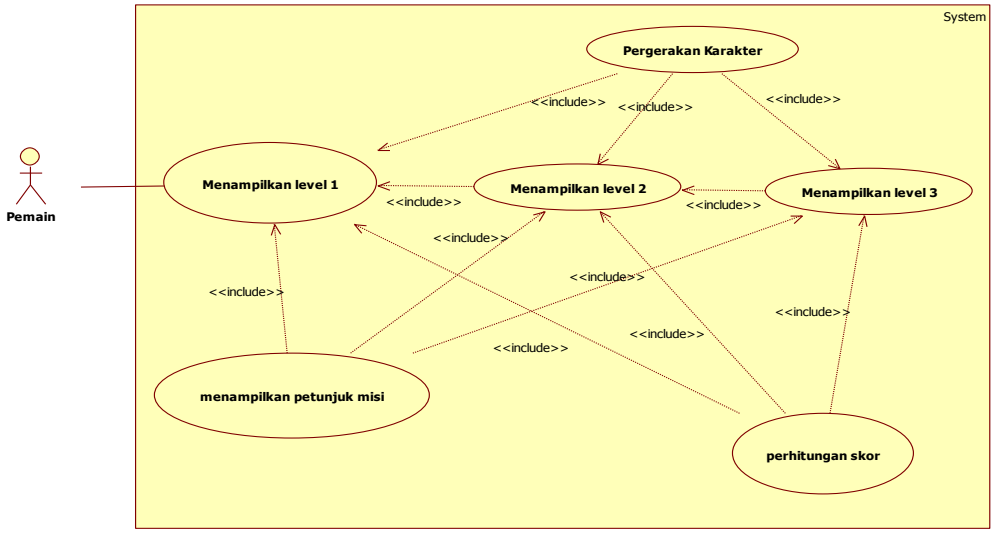
III.3 Analisis Kebutuhan Fungsional

Penggunaan *UML* merupakan tahap pertama yang harus dilakukan untuk menganalisis pemrograman berorientasi objek, penggunaan *UML* digunakan untuk mengetahui apa saja yang menjadi aktor dan sistem, dan agar lebih mudah dilakukan pengembangan terhadap aplikasi yang akan dibuat

III.3.1 *Use Case Diagram*

Use Case diagram merupakan pemodelan untuk memodelkan *behavior* dari sistem informasi yang akan dibuat. *Use Case* sendiri digunakan untuk mendeskrisipkan interaksi antara *actor* dengan sistem dari sistem informasi yang akan dibuat.

Berikut adalah proses-proses yang terjadi di dalam *game takedown the terrorists*, yang digambarkan di dalam diagram *use case* pada gambar III.6



Gambar III.17 Use Case diagram

III.3.2 Definisi Actor

Definisi *Actor* berfungsi untuk menjelaskan *Actor* itu sendiri yang terdapat di dalam *Use Case Diagram.*tabel III.18 merupakan tabel definisi *actor*

Tabel III.16 Definisi Actor

No	*Actor*	Deskripsi
1	Pemain	Orang yang akan memainkan *game*

III.3.3 Definisi Use Case

Definisi *Use Case* berfungsi untuk menjelaskan fungsi apa saja yang bisa dijalankan yang terdapat di dalam *Use Case Diagram.* Definisi *Use Case* diterangkan pada Tabel III.19

Tabel III.17 Definisi Use Case

No	*Use Case*	Deskripsi
1	Menampilkan Level 1	Untuk menampilkan permainan
2	Menampilkan Level 2	Untuk menampilkan nilai tertinggi
3	Menampilkan Level 3	Untuk menggerakkan karakter
5	Pergerakan karakter	Menampilkan petunjuk misi tiap level
6	Perhitungan skor	Untuk menghitung skor
7	Menampilkan petunjuk misi	Untuk menampilkan petunjuk misi

III.3.4 Skenario Use Case

Use Case Skenario berisi alur dari event-event yang terjadi di dalam game, yang menggambarkan urutan interaksi *actor* dengan *use case* dari awal sampai akhir. Berikut ini adalah scenario *use case* yang terdapat pada pembangunan *game takedown the terrorists*

III.3.4.1 Skenario Use Case Menampilkan Level 1

Skenario *Use Case Level* 1 berfungsi untuk menampilkan level 1, skenario *Use Case level 1* diterangkan pada tabel III.20.

Tabel III.18 Skenario Use Case Level 1

Identifikasi	
Nomor	1
Nama	Menampilkan *Level* 1
Tujuan	Untuk menampilkan *scene level* 1
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	Menu Utama ditampilkan
Aksi actor	**Reaksi Sistem**
1. Memilih main pada menu utama	2. Menampillkan video permainan level 1
	3. Menempatkan *player* pada area permainan
	4. Menentukan lokasi pemunculan musuh
	5. Memunculkan musuh
	6. Melakukan Pencarian
	7. Menembak karakter
	8. Mengurangi peluru musuh
	9. Terkena karakter
	10. Mengurangi darah karakter
11. Input tombol w,a,s,d	12. Menggerakkan karakter utama
	13. Menyentuh trigger
	14. Menampilkan misi
15. Menekan klik kiri	16. Mengurangi jumlah peluru
	17. Tembakkan terkena musuh
	18. Menghitung darah
	19. Menghitung skor
20. Mencapai tujuan	21. Menyentuh trigger tujuan
	22. Menghentikan pencarian jalur
	23. Menuju level berikutnya

III.3.4.2 Skenario Use Case Menampilkan Case Level 2

Skenario *Use Case Level* 2 berfungsi untuk menampilkan level 2, skenario *Use Case level* 2 diterangkan pada tabel III.19.

Tabel III.19 Skenario Use Case Level 2

Identifikasi	
Nomor	2
Nama	Menampilkan *Level* 2
Tujuan	Untuk menampilkan *scene level* 2
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	Menu Utama ditampilkan
Aksi actor	**Reaksi Sistem**

1. Memilih main pada menu utama	2. Menampillkan video permainan level 2
	3. Menempatkan *player* pada area permainan
	4. Menentukan lokasi pemunculan musuh
	5. Memunculkan musuh
	6. Melakukan Pencarian
	7. Menembak karakter
	8. Mengurangi peluru musuh
	9. Terkena karakter
	10. Mengurangi darah karakter
11. Input tombol w,a,s,d	12. Menggerakkan karakter utama
	13. Menyentuh trigger
	14. Menampilkan misi
15. Menekan klik kiri	16. Mengurangi jumlah peluru
	17. Tembakkan terkena musuh
	18. Menghitung darah
	19. Menghitung skor
20. Mencapai tujuan	21. Menyentuh trigger tujuan
	22. Menghentikan pencarian jalur
	23. Menuju level berikutnya

III.3.4.3 Skenario Use Case Menampilkan Level 3

Skenario *Use Case Level* 3 berfungsi untuk menampilkan level 3, skenario *Use Case level* 3 diterangkan pada tabel III.20.

Tabel III.20 Skenario Use Case Level 3

Identifikasi	
Nomor	3
Nama	Menampilkan *Level* 3
Tujuan	Untuk menampilkan *scene level* 3
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	Menu Utama ditampilkan
Aksi actor	**Reaksi Sistem**
1. Memilih main pada menu utama	2. Menampillkan video permainan level 3
	3. Menempatkan *player* pada area permainan
	4. Menentukan lokasi pemunculan musuh
	5. Memunculkan musuh
	6. Melakukan Pencarian

	7. Menembak karakter
	8. Mengurangi peluru musuh
	9. Terkena karakter
	10. Mengurangi darah karakter
11. Input tombol w,a,s,d	12. Menggerakkan karakter utama
	13. Menyentuh trigger
	14. Menampilkan misi
15. Menekan klik kiri	16. Mengurangi jumlah peluru
	17. Tembakkan terkena musuh
	18. Menghitung darah
	19. Menghitung skor
20. Mengarahkan player	21. Mengambil item
	22. Item tersisa = 0
	23. Mengaktifkan trigger tujuan
	24. Mencapai trigger tujuan
	25. Menghentikan pencarian jalur
	26. Memanggil scene berikutnya

III.3.4.4 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 III.21.

Tabel III.21 Skenario Use Case Pergerakan Karakter

Identifikasi	
Nomor	3
Nama	Pergerakan karakter
Tujuan	Untuk menggerakan karakter yang dimainkan
Aktor	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
	3. Karakter bergerak maju
	4. Karakter bergerak ke kiri
	5. Karakter bergerak mundur
	6. Karakter bergerak ke kanan
Kondisi Akhir	Karakter bergerak

III.3.4.5 Skenario Use Case Perhitungan Skor

Skenario*Use Case* perhitungan skor berfungsi untuk menghitung skor yang didapat oleh pemain, tabel perhitungan skor dapat dilihat pada tabel III.22.

Tabel III.22 Skenario Use Case Perhitungan Skor

Identifikasi	
Nomor	4
Nama	Perhitungan Skor
Tujuan	Untuk menghitung skor yang didapat
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	Menampilkan skor dengan nilai 0
Aksi actor	**Reaksi Sistem**
1. Membunuh musuh	2. Mengubah skor sesuai dengan musuh yang dibunuh
Kondisi Akhir	Hasil perhitungan skor

III.3.4.7 Skenario Use Case Menampilkan Petunjuk Misi

Skenario *Use Case* menampilkan petunjuk misi berfungsi untuk menampilkan misi yang ada di dalam *game*, tabel menampilkan petunjuk misi dapat dilihat pada tabel III.23.

Tabel III.23 Skenario Use Case Menampilkan Petunjuk Misi

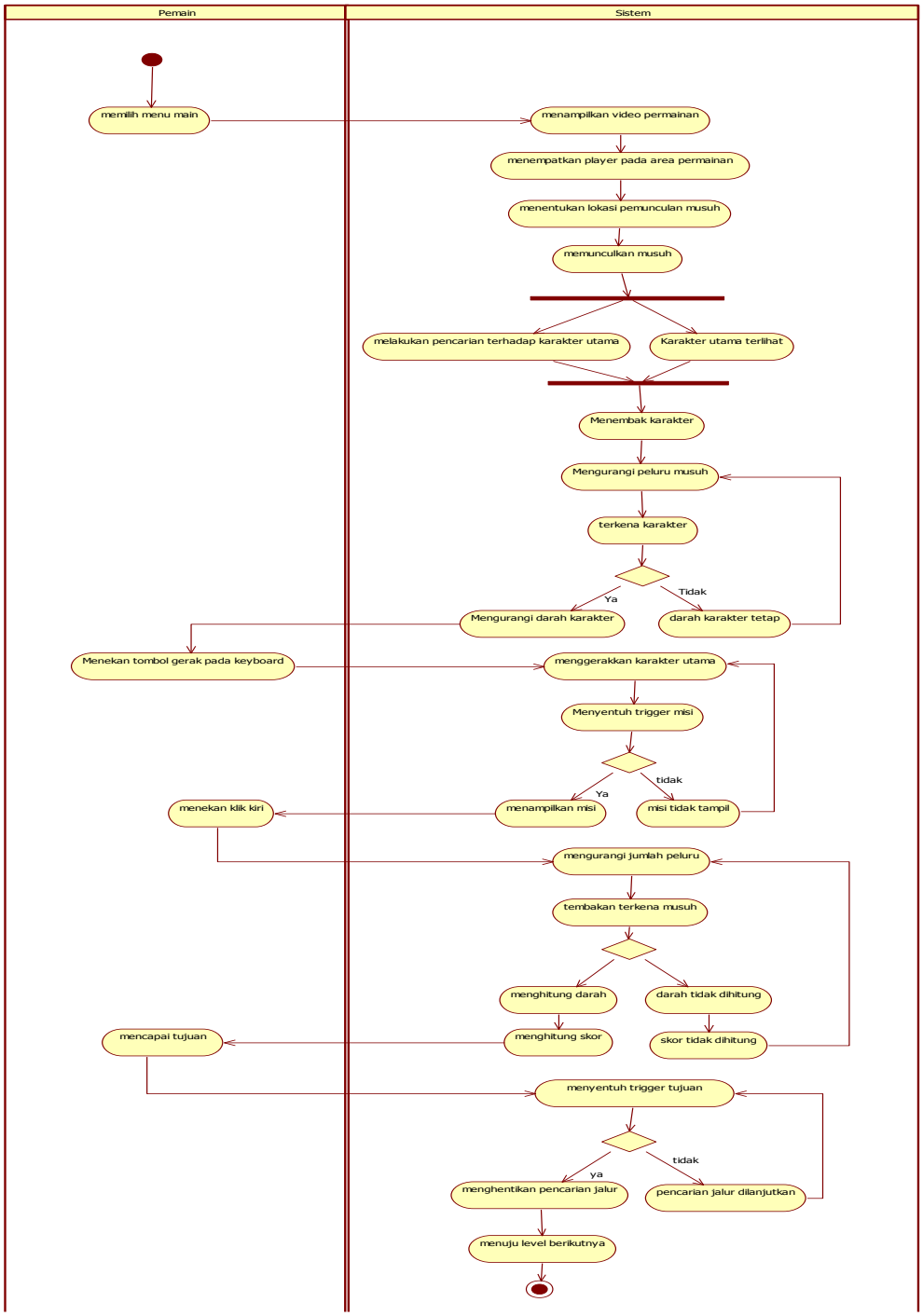
Identifikasi	
Nomor	5
Nama	Menampilkan Petunjuk Misi
Tujuan	Untuk menampilkan misi yang ada pada *game*
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	Menyajikan permainan
Aksi actor	**Reaksi Sistem**
1. Menggerakkan karakter menuju trigger	2. Menghilangkan trigger
	3. Memanggil petunjuk misi
Kondisi Akhir	Menampilkan petunjuk misi

III.3.5 Activity Diagram

Activitiy diagram menggambarkan alur dari berbagai aktifitas dalam sistem yang sedang dibangun, untuk menerangkan bagaimana alur dari masing-masing proses secara mendetail. Activity diagram ini diawali dengan *initial node* hingga mencapai *final state*, dan di dalam activity diagram ini bisa melakukan behavior parallel.

III.3.5.1 Activity Diagram Menampilkan Level 1

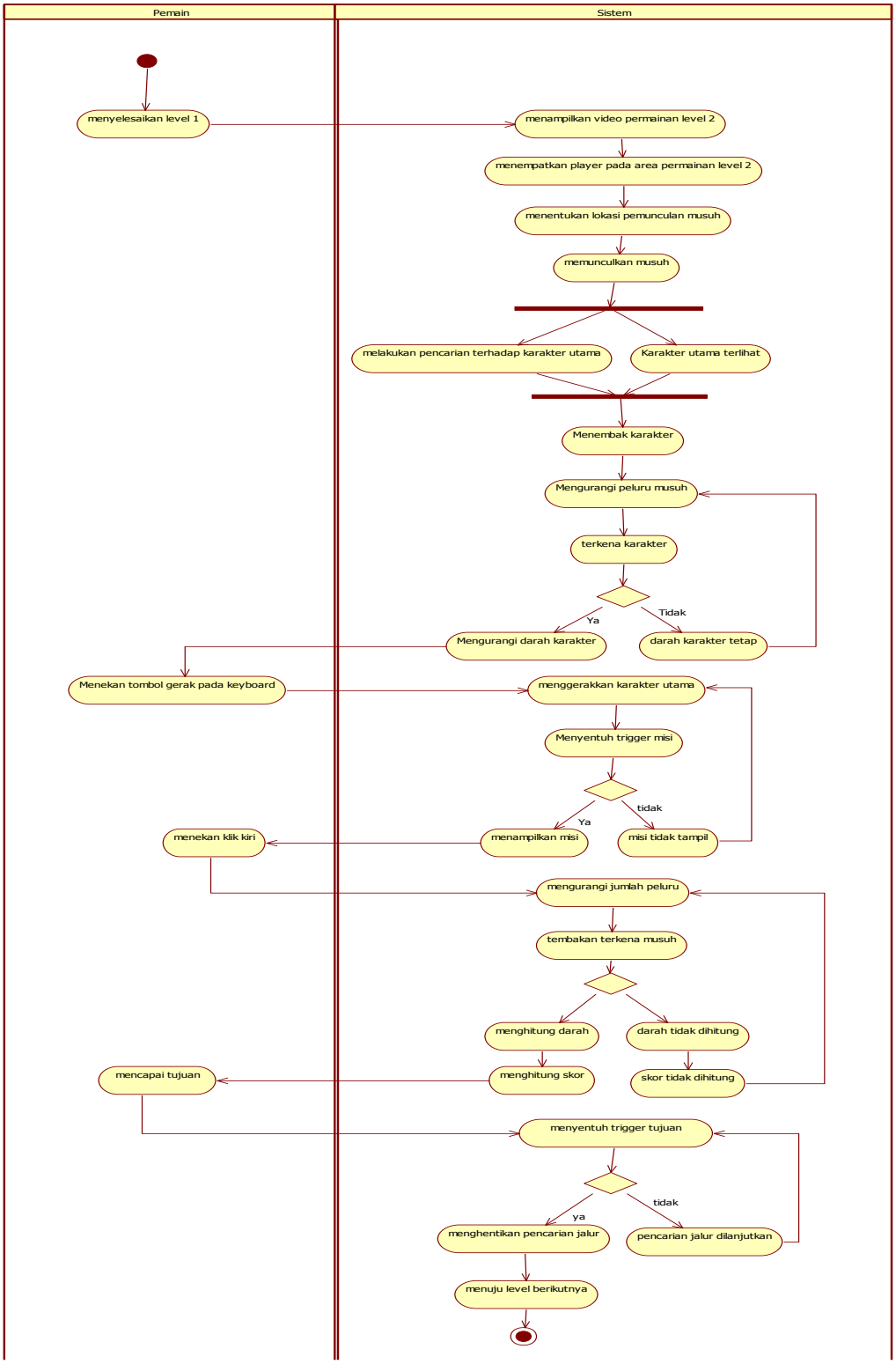
Activity diagram menampilkan *scene level 1* merupakan aktivitas diagram untuk menyajikan seluruh konten permainan yang ada di dalam *level* 1 pada game *activity diagram* menampilkan penyajian permainan dapat dilihat pada gambar III.18



Gambar III.18 Activity Diagram Menampilkan Level 1

III.3.5.2 Activity Diagram Menampilkan Level 2

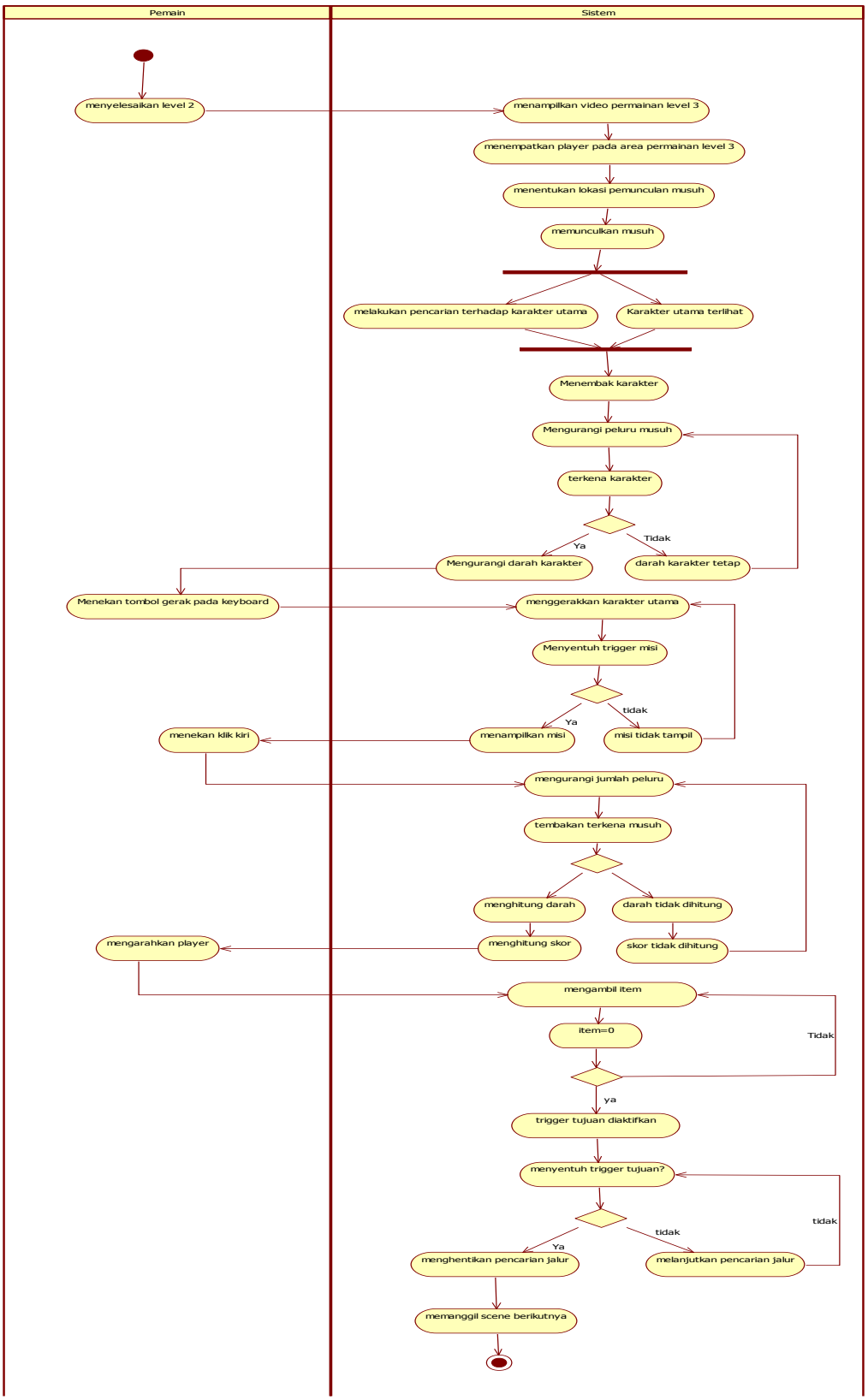
Activity diagram menampilkan *scene level* 2 merupakan aktivitas diagram untuk menyajikan seluruh konten permainan yang ada di dalam *level* 2 pada game *activity diagram* menampilkan penyajian permainan dapat dilihat pada gambar III.19



Gambar III.19 Activity Diagram Menampilkan Level 2

III.3.5.3 Activity Diagram Menampilkan Level 3

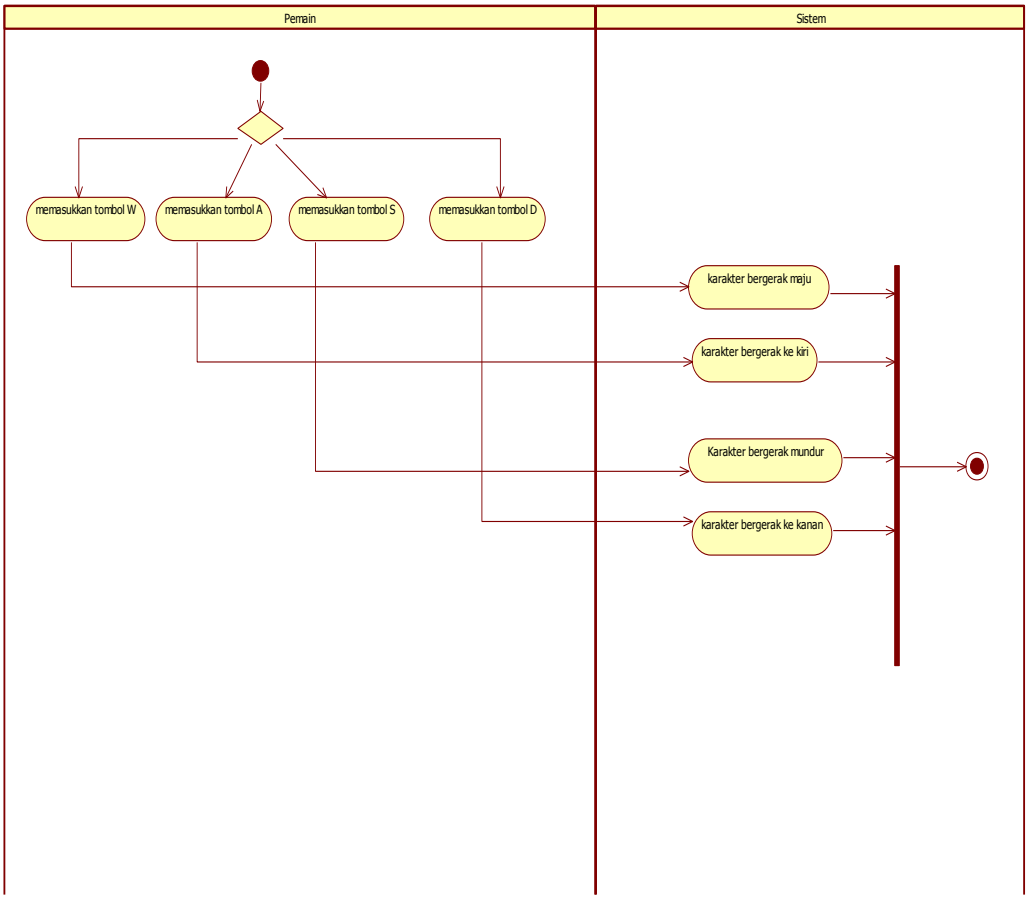
Activity diagram menampilkan *scene level* 3 merupakan aktivitas diagram untuk menyajikan seluruh konten permainan yang ada di dalam *level* 3 pada game *activity diagram* menampilkan penyajian permainan dapat dilihat pada gambar III.20



Gambar III.20 Activity Diagram Menampilkan Level 3

III.3.5.4 Activity Diagram Pergerakkan Karakter

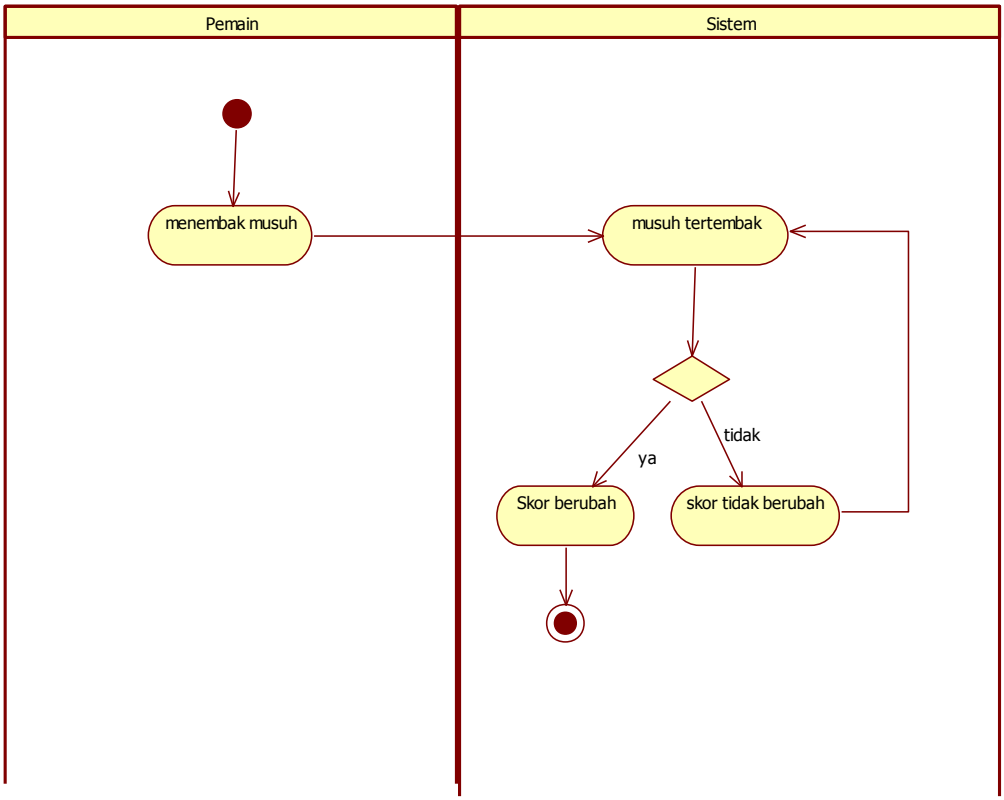
Activity diagram pergerakkan karakter digunakan untuk menampilkan menggerakkan karakter utama yang ada di dalam *game*. *Activity diagram* pergerakkan karakter bisa dilihat pada gambar III.21



Gambar III.21 Activity Pergerakkan Karakter

III.3.5.5 Activity Diagram Perhitungan Skor

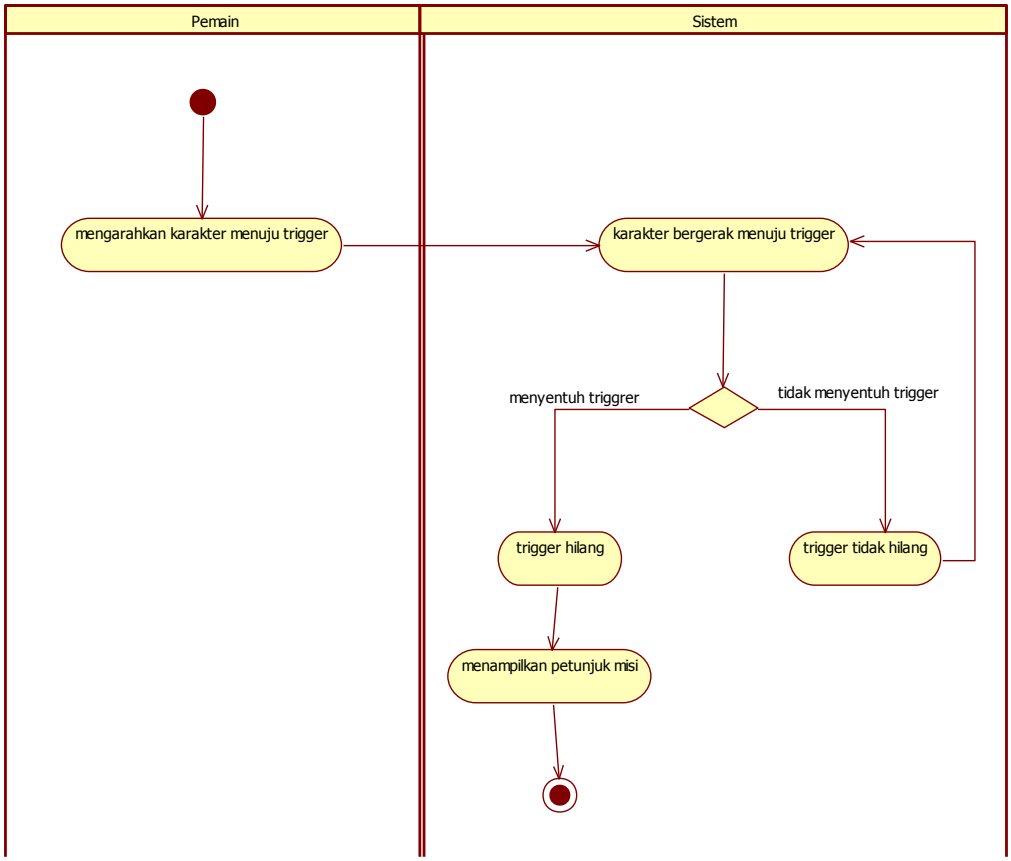
Activity diagram perhitungan skor digunakan untuk menghitung skor yang didapat ketika sedang memainkan *game*. *Activity diagram* perhitungan skor bisa dilihat pada gambar III.22.



Gambar III.22 Activity Diagram Perhitungan Skor

III.3.5.6 Activity Diagram Menampilkan Petunjuk Misi

Activity diagram menampilkan petunjuk misi digunakan untuk menampilkan misi yang harus dijalankan ketika *player* berada di dalam arena *game*. *Activity diagram* perhitungan skor bisa dilihat pada gambar III.23.



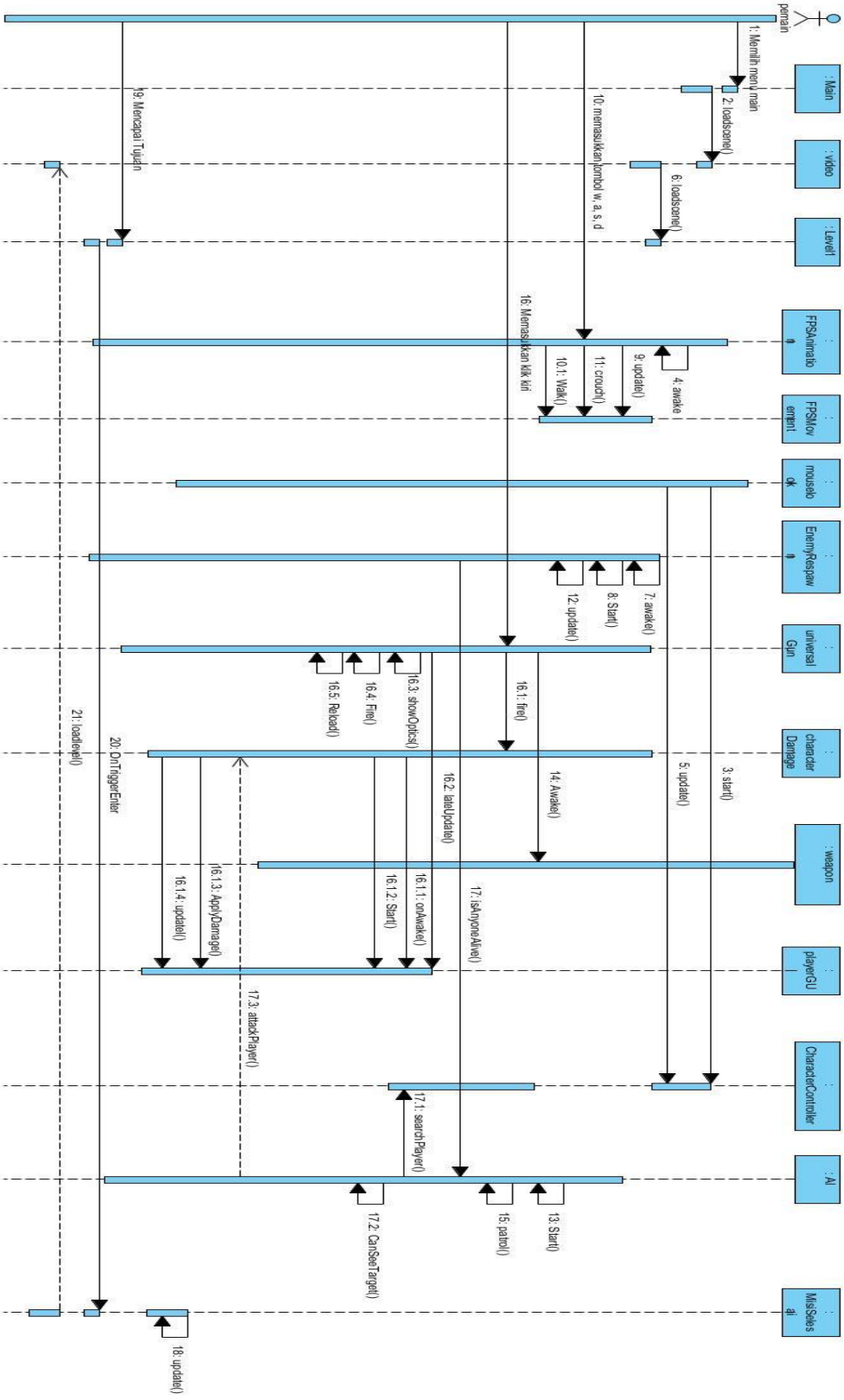
Gambar III.23 Activity Diagram Menampilkan Petunjuk Misi

III.3.6 Sequence Diagram

Sequence diagram berfungsi untukmenggambarkan perilaku objek pada use case dengan mendeskripsikan garis hidup (*lifeline*) dan pesan yang dikirimkan dan di terima antar objek.

III.3.6.1 Sequence Diagram Menampilkan Level 1

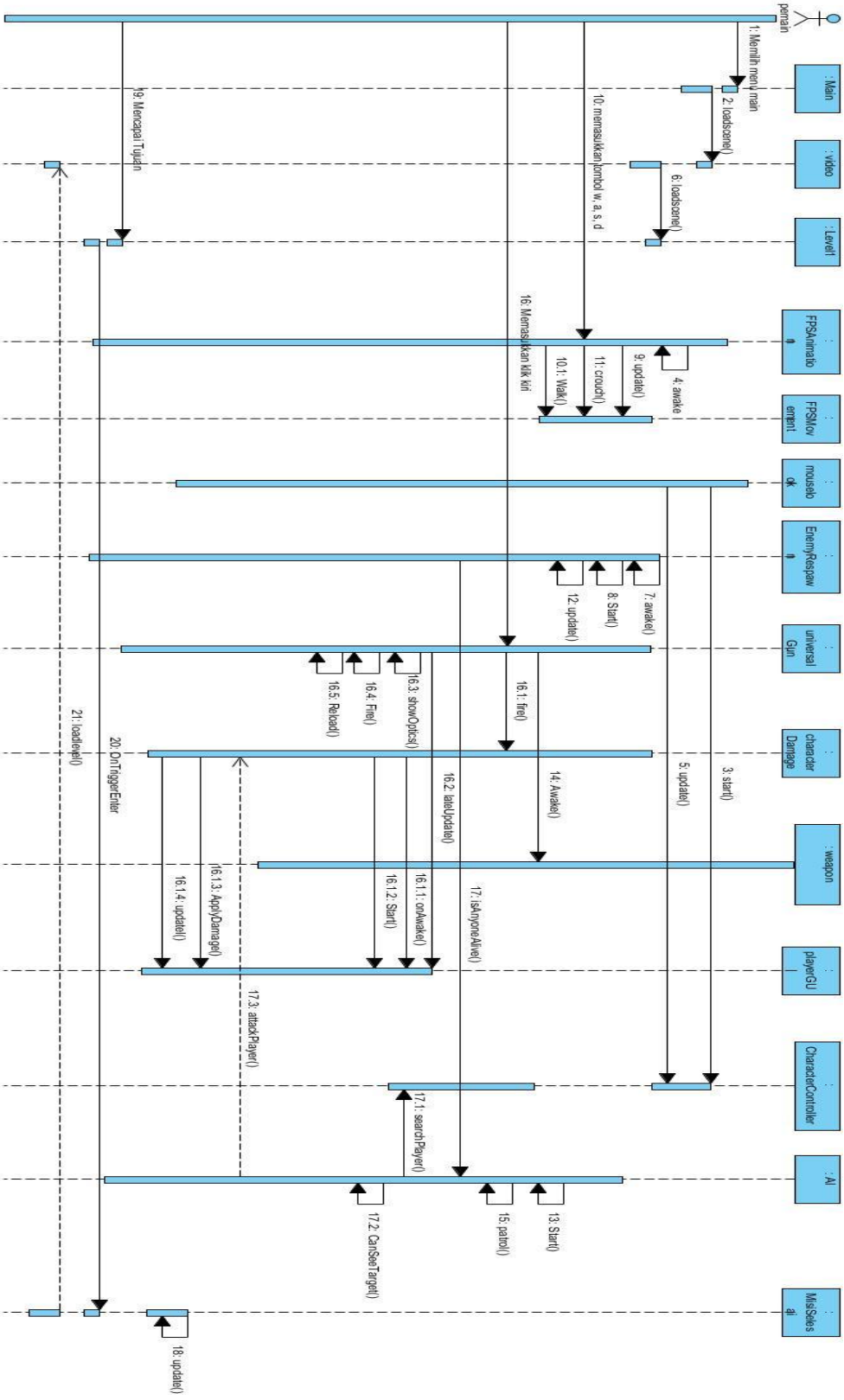
Sequence diagram untuk menampilkan *level* 1 bisa dilihat pada gambar III.24.



Gambar III.24 Sequence Diagram Menampilkan Level 1

III.3.6.2 Sequence Diagram Menampilkan Level 2

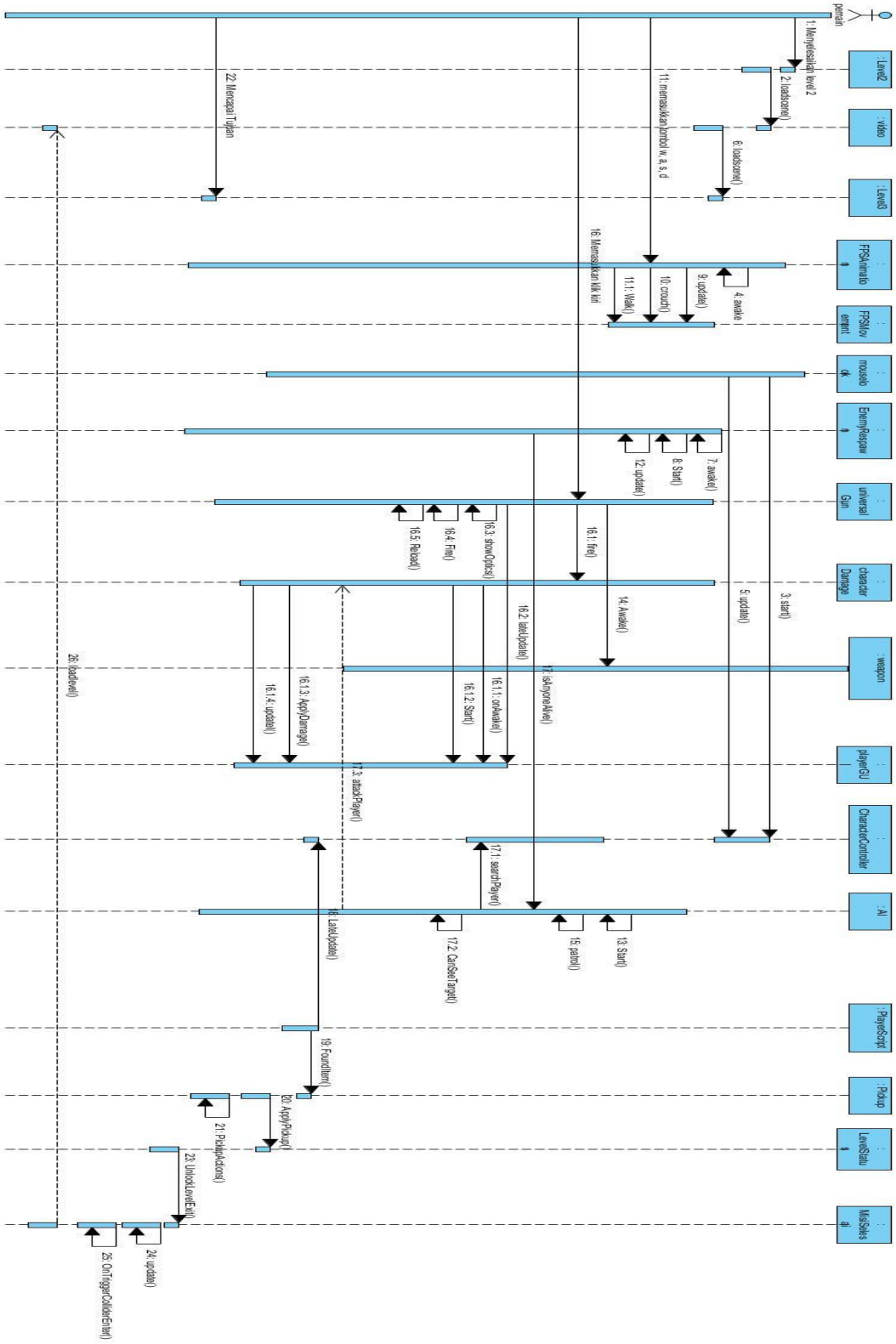
Sequence diagram untuk menampilkan *level* 2 bisa dilihat pada gambar III.25.



Gambar III.25 Sequence Diagram Menampilkan Level 2

III.3.6.3 Sequence Diagram Menampilkan Level 3

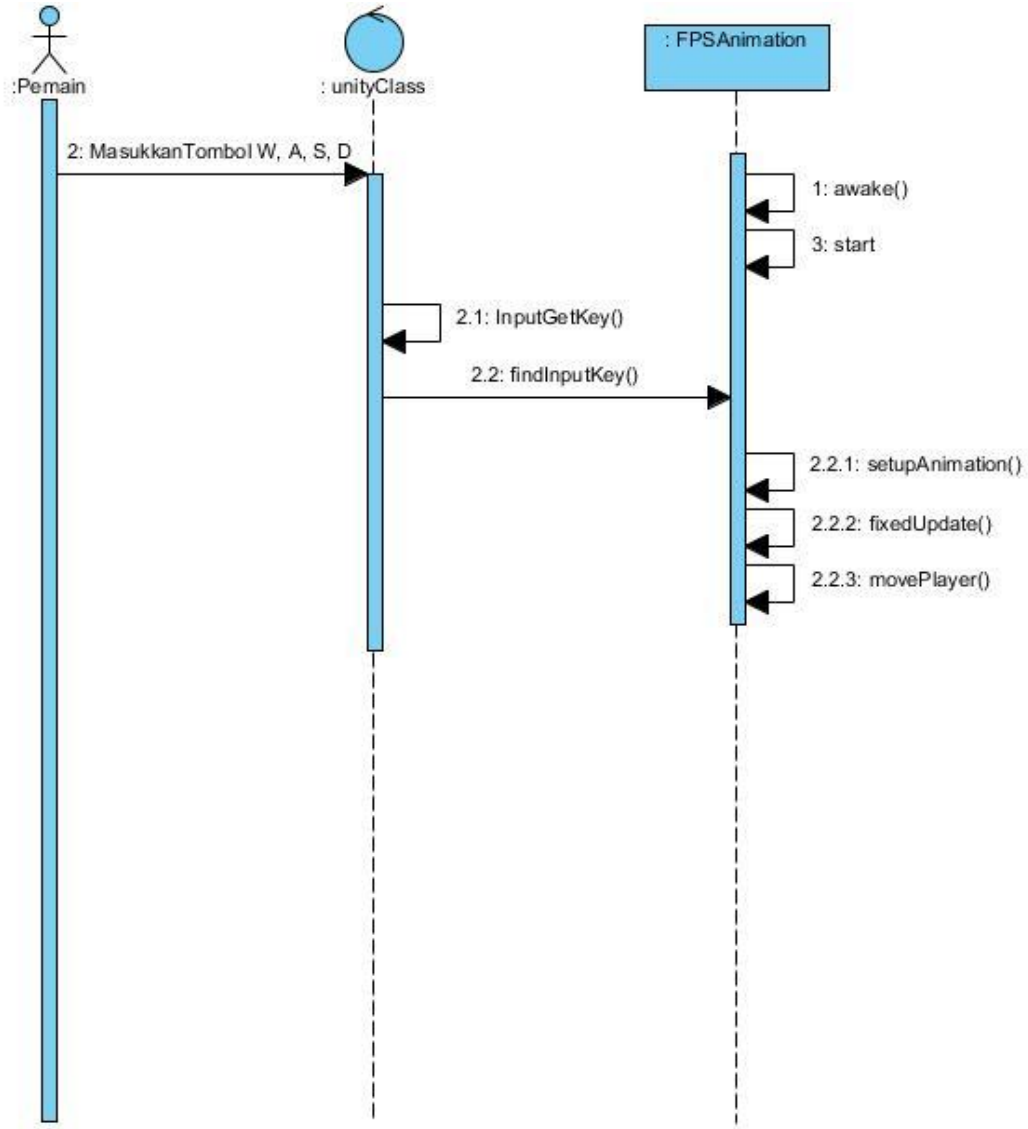
Sequence diagram untuk menampilkan *level* 3 bisa dilihat pada gambar III.



Gambar III.26 Sequence Diagram Menampilkan Level 3

III.3.6.4 Sequence Diagram Pergerakan Karakter

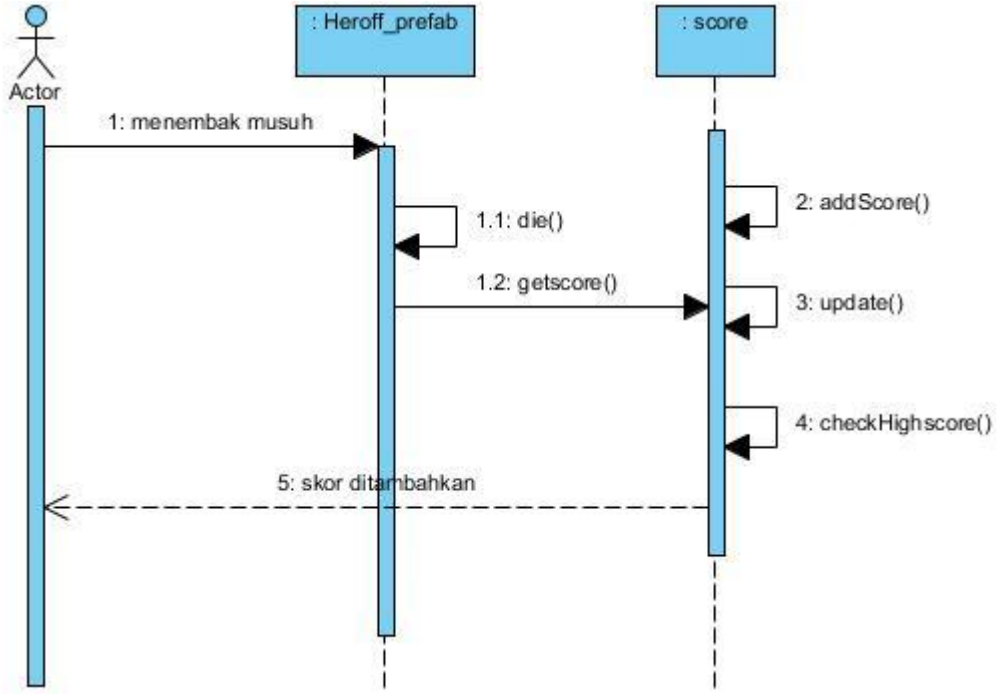
Sequence diagram untuk melakukan pergerakan karakter bisa dilihat pada gambar III.27



Gambar III.27 Sequence Diagram Pergerakan Karakter

III.3.6.5 Sequence Diagram Perhitungan Skor

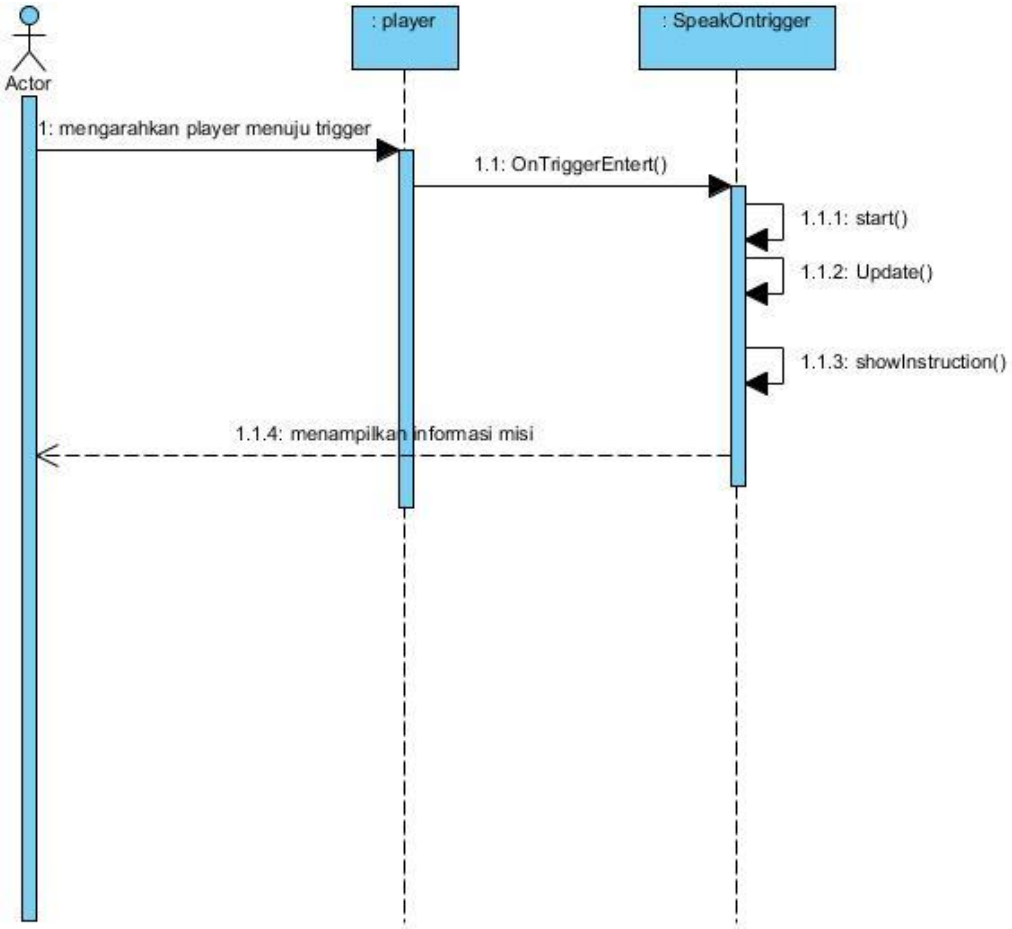
Sequence diagram perhitungan skor bisa dilihat pada gambar III.28.



Gambar III.28 Sequence Diagram Perhitungan Skor

III.3.6.6 Sequence Diagram Menampilkan Petunjuk Misi

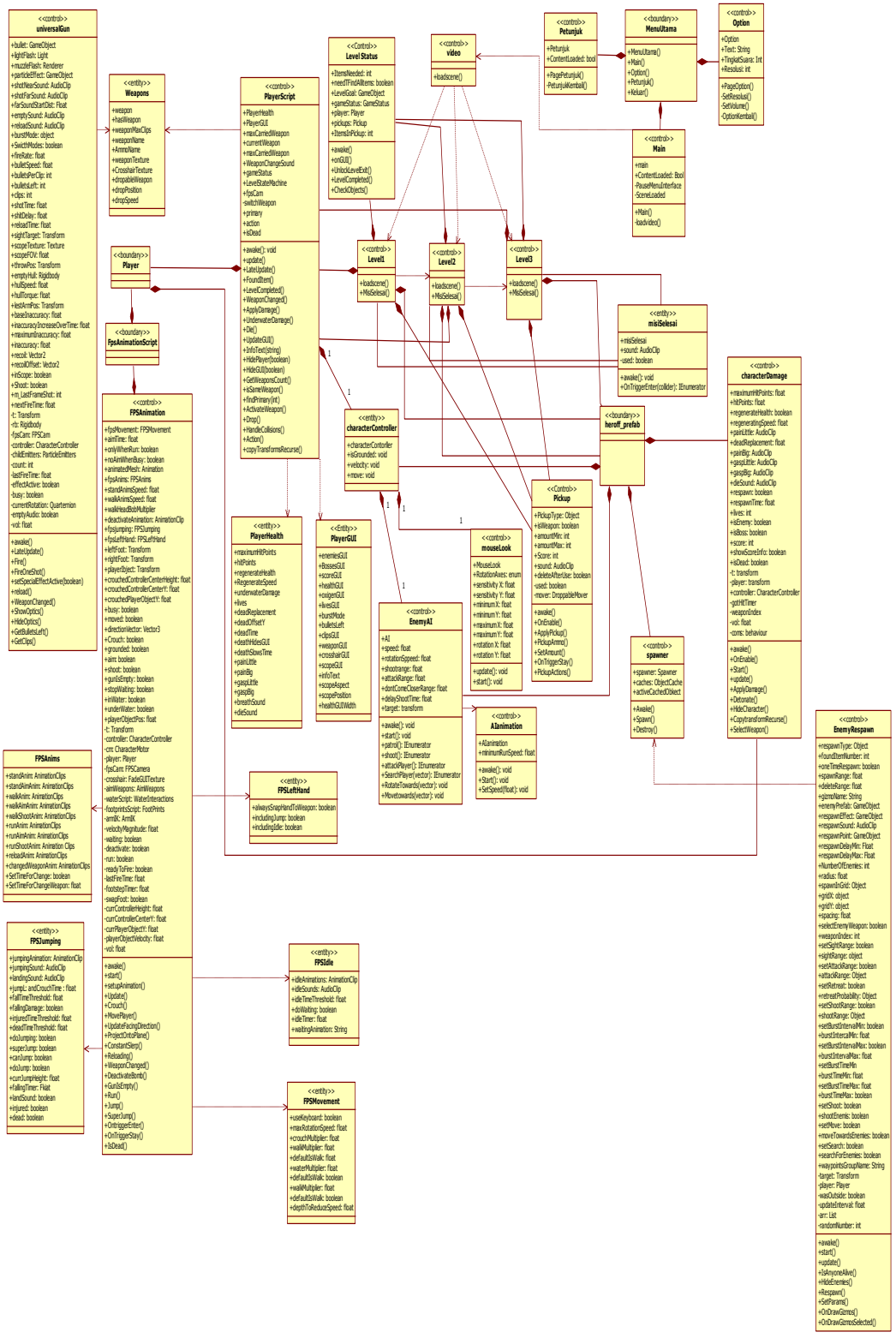
Sequence diagram menampilkan petunjuk misi bisa dilihat pada gambar III.29.



Gambar III.29 Sequence Diagram Menampilkan Petunjuk Misi

III.3.7 Class Diagram

Class Diagram menggambarkanstruktur sistem dengan mendefinisikan class-class yang ada di dalam sistem, *class diagram* dari *game takedown the terrorists* bisa dilihat pada gambar III.30



Gambar III.30 Class Diagram Game Takedown The Terrorists

III.4 Perancangan sistem

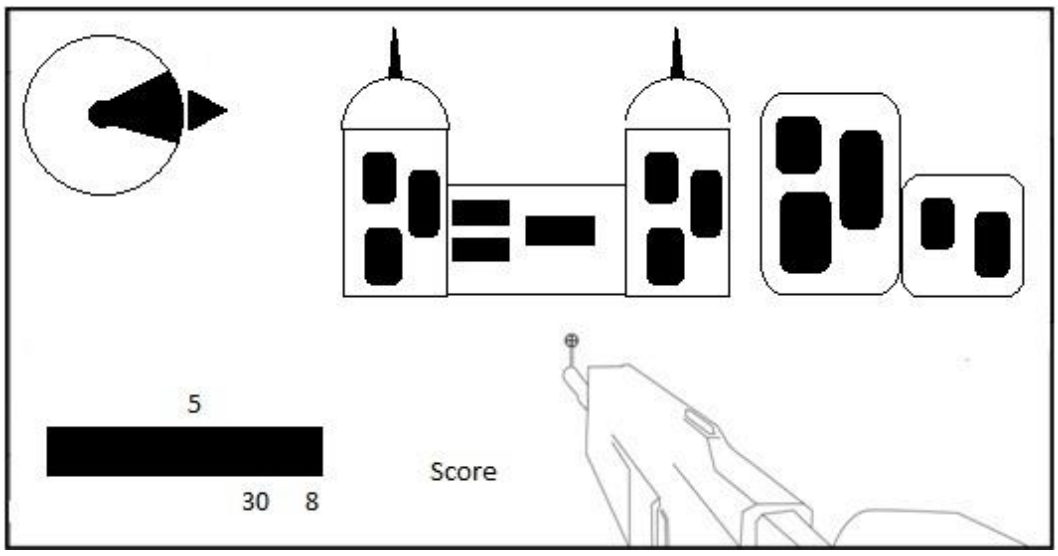
Perancangan sistem adalah merancang atau mendesain suatu sistem yang baik, yang isinya adalah langkah-langkah operasi dalam proses pengolahan data dan prosedur untuk mendukung operasi sistem.

III.4.1 Perancangan Komponen Permainan

Perancangan komponen permainan meliputi karakter dan *storyboard* berisi tentang penjelasan karakter dan *storyboard* sebagai penjelasan latar yang digunakan, berupa gambar sebagai perancangan latar yang akan digunakan dalam *game takedown the terrorists* ini.

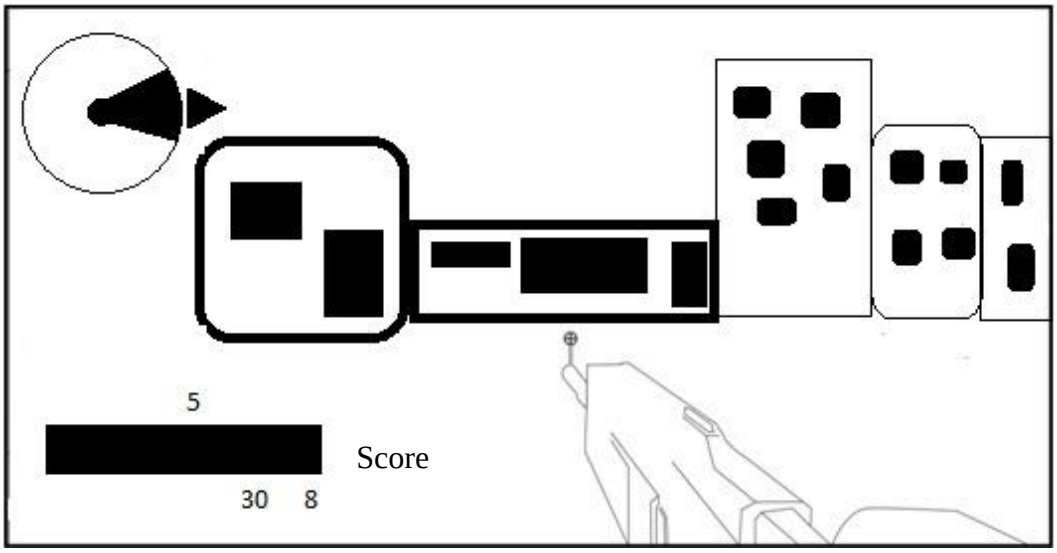
III.4.1.1 Storyboard

Storyboard adalah gambar atau *prototype* dari permainan yang digunakan sebagai perancangan awal sebelum diimplementasikan ke dalam *game*



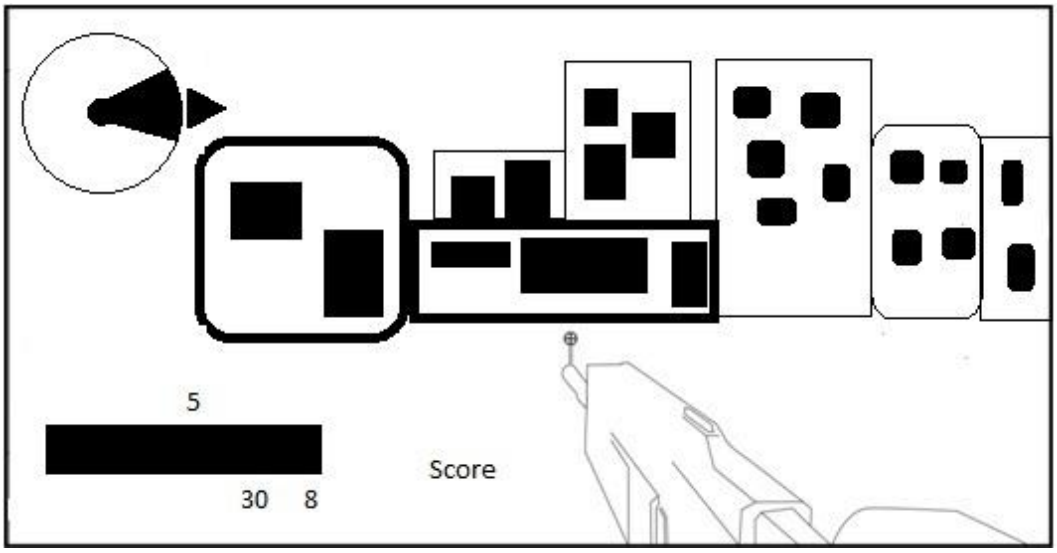
Gambar III.31 Storyboard *level* 1

Pada gambar III.31 di atas menunjukkan *level* 1 dimana cerita dimulai di sebuah perkotaan dimana di kota ini *player* harus menemukan mobil gegana untuk menuju kota selanjutnya



Gambar III.32 Storyboard *level* 2

Gambar III.32 memiliki setting disebuah perkotaan yang telah dikuasai oleh para kelompok teroris, disini *player* akan dihadapkan dengan jumlah musuh yang lebih banyak dan beragam dan harus menyelamatkan pasukannya yang disandera.



Gambar III.33 Storyboard *Level* 3

Gambar III.33 di atas memiliki setting yang sama di sebuah perkotaan *player* diharuskan mencapai taman sari untuk menyelamatkan bangunan tersebut

dari ancaman para teroris, pada *level* ini musuh akan lebih kuat tetapi lebih dan jarak tempuh *player* untuk mencapai tujuan yang semakin jauh.

III.4.1.2 Pengenalan Karakter

Karakter utama dalam *game* ini bernama Kapten Zulfikar seorang kapten dari sebuah pasukan khusus anti teroris. Kapten Zulfikar dilengkapi berbagai macam senjata yang berbeda di setiap *level*nya. Properti yang dimiliki karakter utama bisa dilihat pada tabel III.26

Tabel III.24 properti karakter utama

Nama	Kapten Zulfikar
Darah	200
Kemampuan	Berlari, melompat, menembak, mengganti senjata, jongkok, berjalan
Senjata	M4A1 dengan Grenade Launcher dan Dragunov

Senjata yang digunakan oleh Kapten Zulfikar ada dua macam, pertama adalah senjata M4A1 yang kedua adalah Dragunov. Untuk penggunaan senjata M4 yang disertai dengan grenade launcher dapat digunakan hanya pada *level* terakhir gambar dari masing-masing dari senjata tersebut bisa dilihat pada gambar III.34 dan III.35



Gambar III.34 Senjata M4A1 dengan Grenade Launcher



Gambar III.35 Senjata Dragunov

Tabel III.25 tabel senjata

Properti	M4A1	Dragunov	Grenade Launcher
Damage	20	20	100
Peluru	30	5	1
Jumlah Clip	5	5	8

Musuh yang terdapat di dalam game ini berupa seorang pria yang menggunakan rompi hitam berikut gambarnya ada pada gambar III.36



Gambar III.36 musuh yang terdapat di dalam game

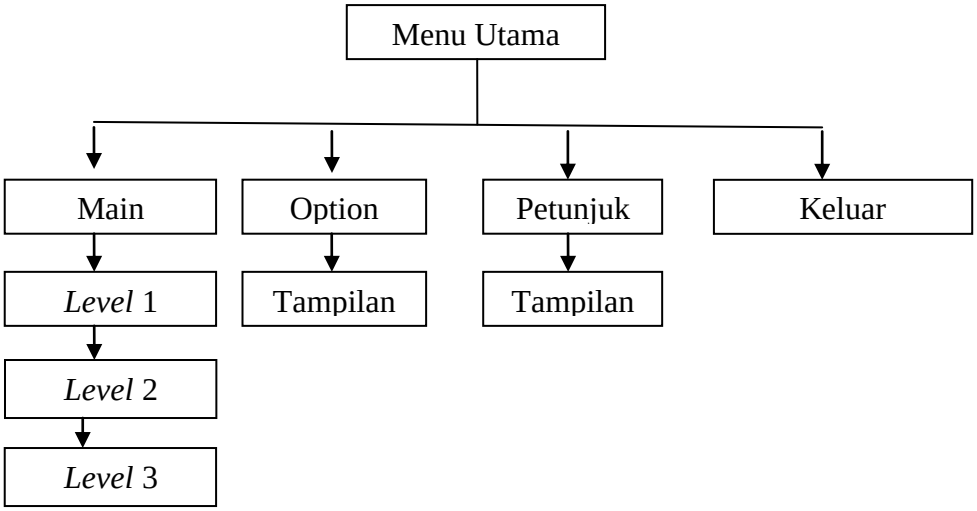
NPC teman yang digunakan pada *game* ini adalah karakter laki-laki dengan menggunakan baju hitam, dengan rompi anti peluru



Gambar III.37 Karakter teman yang terdapat di dalam game

III.4.2 Perancangan Struktur Menu

Perancangan Struktur Menu dibuat untuk mempermudah pembuat dalam membuat rancangan menu agar bisa dengan mudah digunakan oleh *user* `dalam memilih menu aplikasi yang sedang berjalan struktur menu dapat dilihat pada gambar III.38



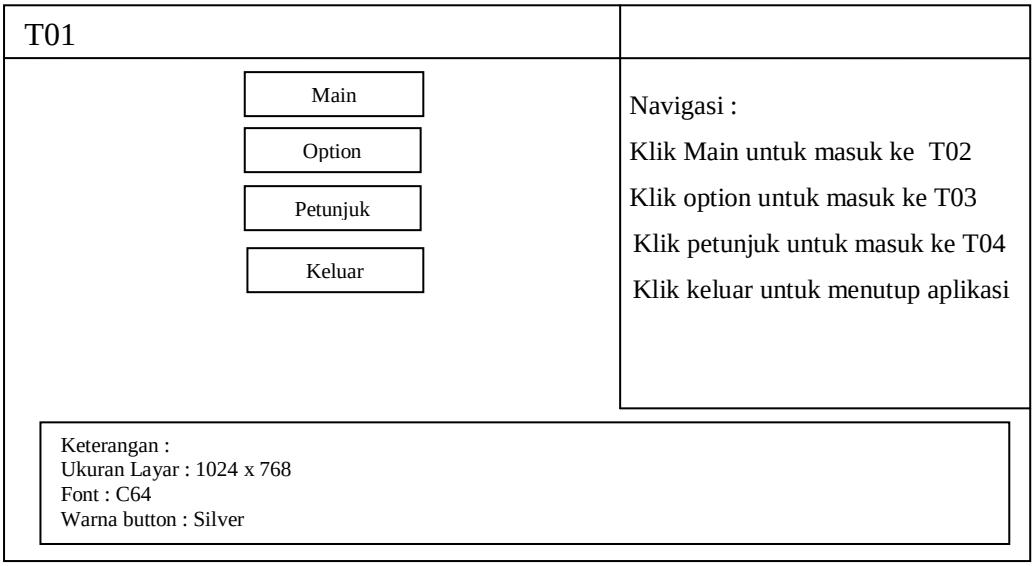
Gambar III.38 Struktur Menu Utama

III.4.3 Perancangan antarmuka

Perancangan antarmuka digunakan untuk menggambarkan gambaran masing-masing dari struktur menu.Perancangan antarmuka yang dibuat meliputi perancangan antar muka menu utama, perancangan antarmuka *option*, perancangan antarmuka petunjuk.

III.4.3.1 Perancangan Antarmuka Menu Utama

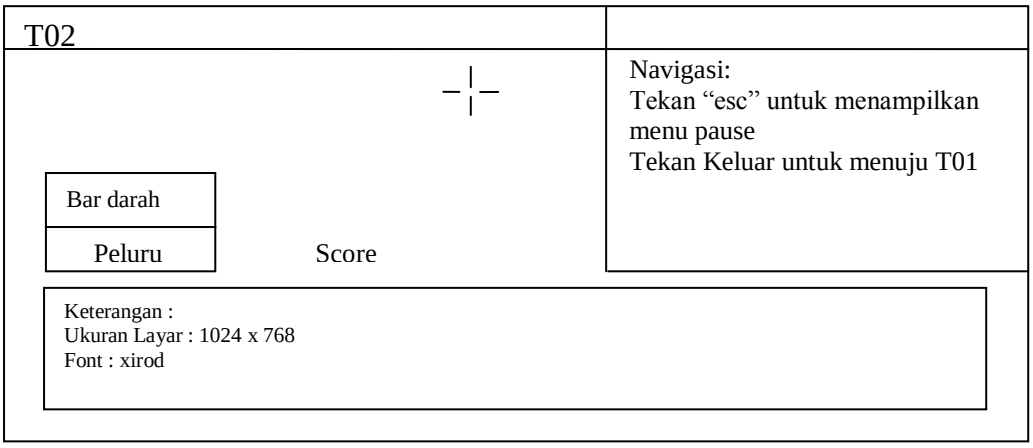
Masuk ke Menu Utama menampilkan lima tombol pada layar yang bisa dipilih oleh *user* sebelum bermain *game*. Antarmuka Menu Utama dapat dilihat pada gambar III.39



Gambar III.39 Antarmuka Menu Utama

III.4.3.2 Perancangan Antarmuka Permainan

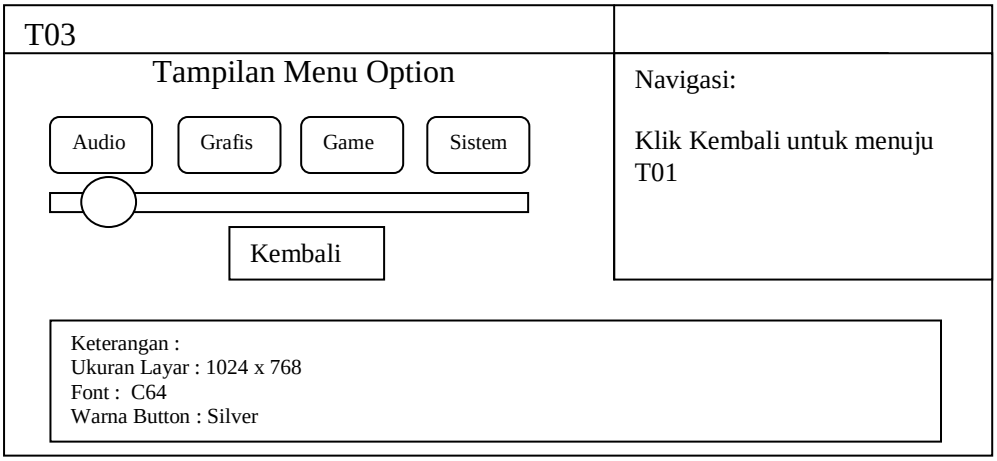
Perancangan antarmuka permainan dari *game takedown the terrorists* dapat dilihat pada gambar III.40



Gambar III.40 Antarmuka permainan

III.4.3.3 Perancangan Antarmuka Menu Option

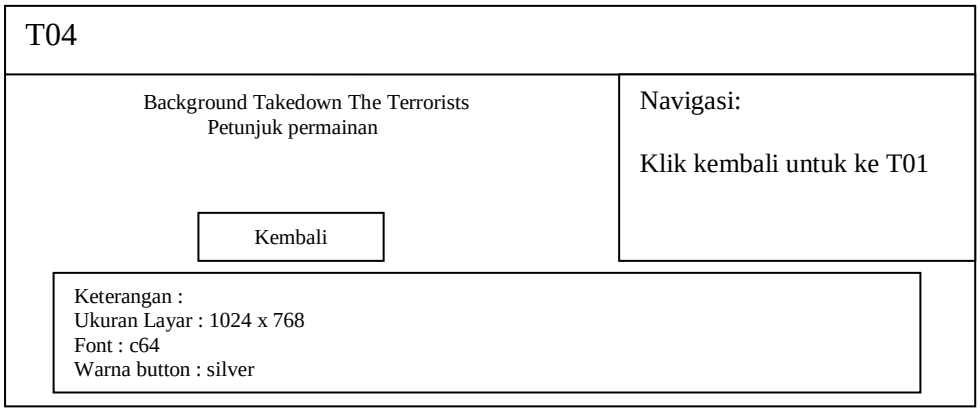
Perancangan antarmuka menu option dari *game takedown the terrorists* dapat dilihat pada gambar III.41



Gambar III.41 Antarmuka Menu Option

III.4.3.4 Perancangan Antarmuka petunjuk

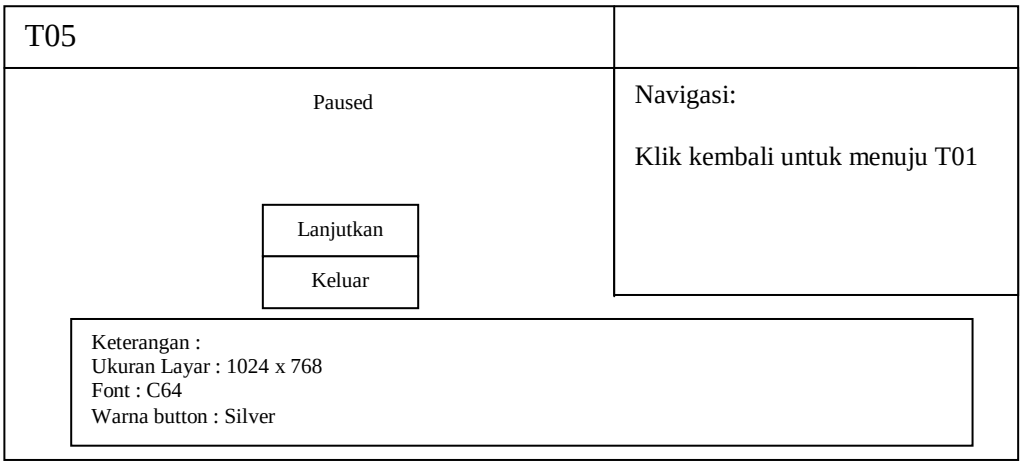
Perancangan antarmuka menu petunjuk dari *game takedown the terrorists* dapat dilihat pada gambar III.42



Gambar III.42 Antarmuka Petunjuk

III.4.3.5 Perancangan Menu Pause

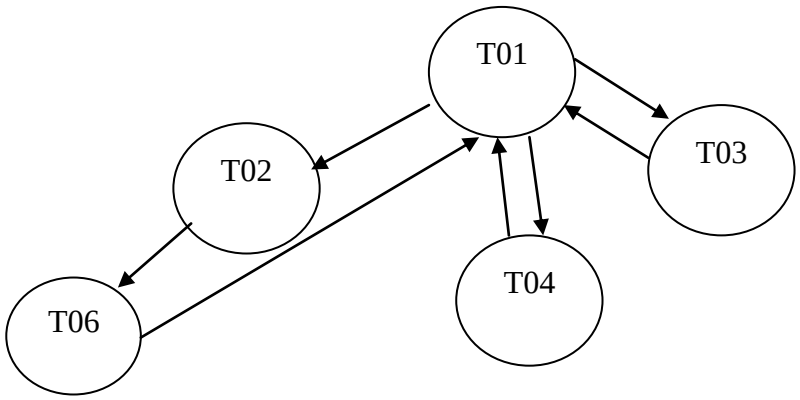
Perancangan antarmuka menu *pause* dari *game takedown the terrorists* dapat dilihat pada gambar III.43



Gambar III.43 Antarmuka Menu Pause

III.4.4 Jaringan Semantik

Gambaran pengetahuan grafis yang menunjukkan hubungan antar berbagai objek, terdiri dari lingkaran-lingkaran yang dihubungkan dengan anak panah yang menunjukkan objek dan informasi tentang objek-objek tersebut disebut jaringan semantik.Jaringan semantik dari *game takedown the terrorists* ini bisa dilihat pada gambar III.44.



Gambar III.44 Jaringan Semantik *Takedown The Terrorists*

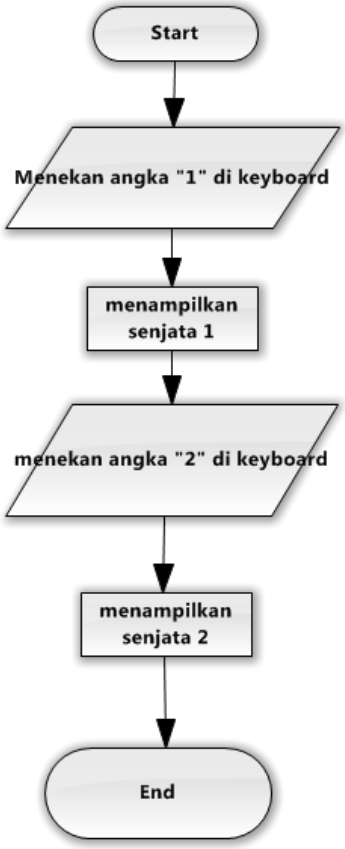
III.4.5 Perancangan Method

Perancangan komponen *method* merupakan perancangan untuk mendeskripsikan *method-method* yang ada di dalam sistem dari aplikasi yang akan dibangun. *Method* dapat dipanggil dengan menyertakan variabel atau tanpa

menyertakan variabel. Berikut ini adalah *method-method* yang digunakan di dalam *game takedown the terrorists*.

III.4.5.1 *PlayerWeapons.SelectWeapon*()

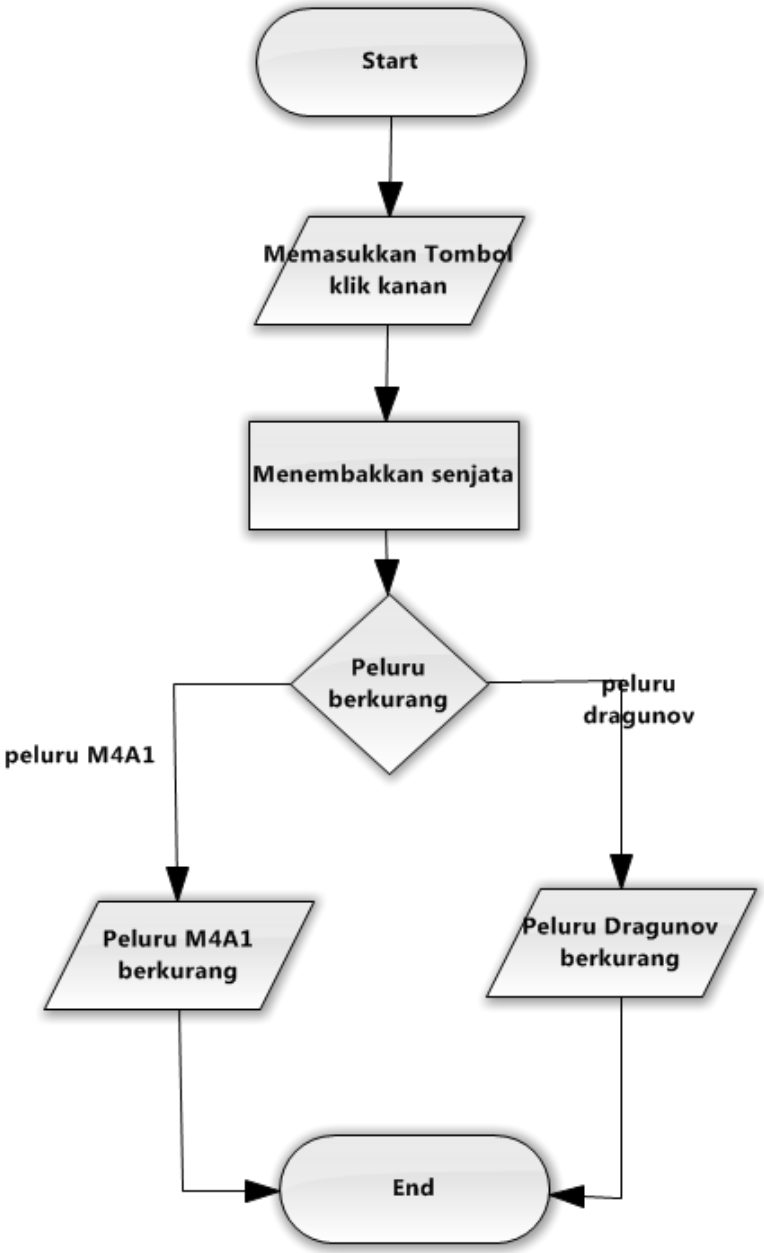
Method ini berada pada *class PlayerWeapons* digunakan untuk melakukan update penggantian senjata dengan menekan tombol “1” atau “2” pada *keyboard*, *method,* bisa dilihat pada gambar III.45



Gambar III.45 Flowchart Method PlayerWeapons.SelectWeapon

III.4.5.2 Method Enemy.*UpdateGUI*()

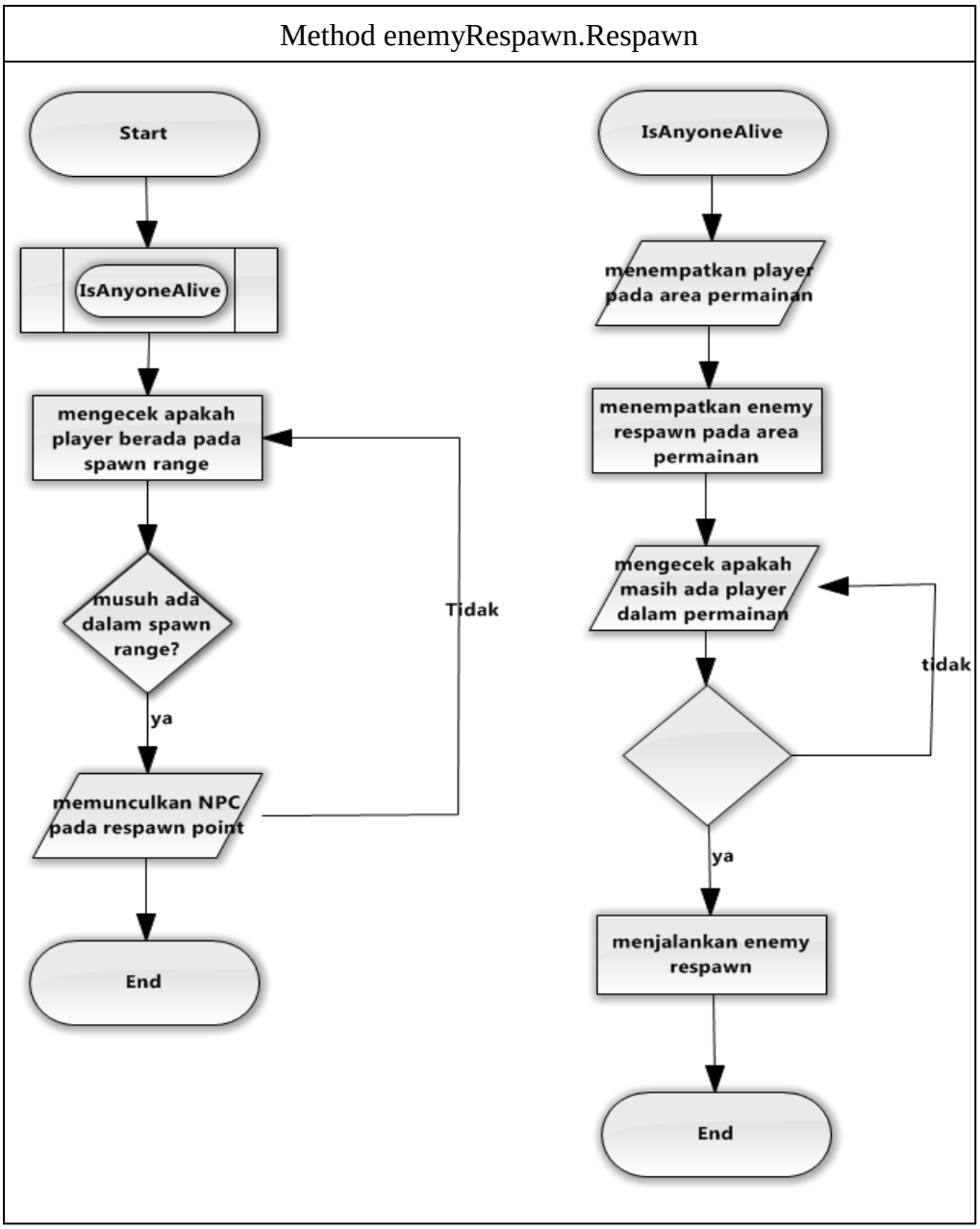
Method ini berada pada *class FPSPlayer* yang digunakan sebagai GUI jumlah peluru yang tersisa dari masing-masing senjata, *method updateGUI* dapat dilihat pada gambar III.46



Gambar III.46 Method FPSPlayer.UpdateGUI

III.4.5.3Method EnemyRespawn.Respawn()

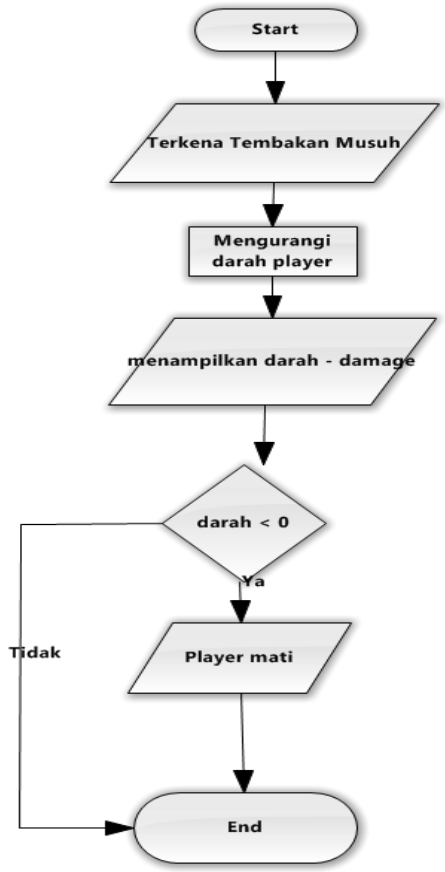
Method berada pada class Respawn, *method* ini digunakan untuk menentukan letak pemunculan musuh pada area permainan *method respawn* bisa dilihat pada gambar III.47



Gambar III.47 Method EnemyRespawn.Respawn

III.4.5.4 Method *CharacterDamage.ApplyDamage*()

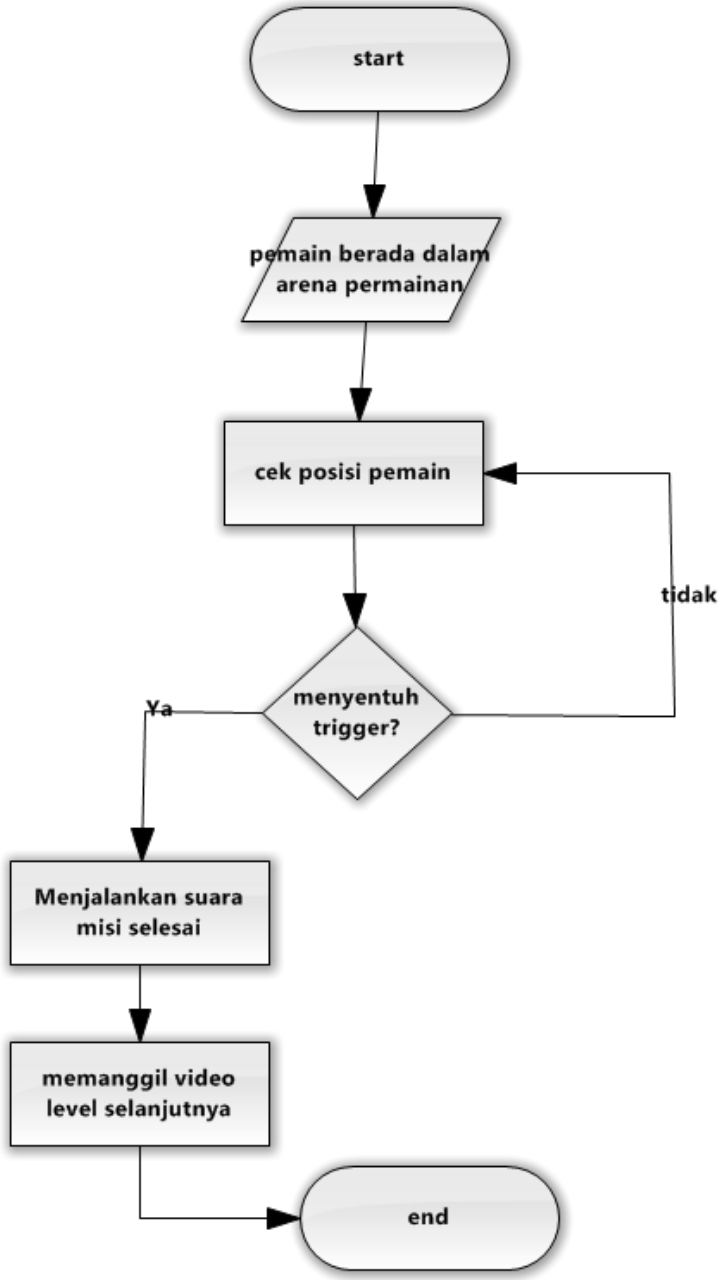
Method ini terdapat pada *class CharacterDamage method* ini digunakan untuk mengurangi darah atau *healthbar* dari player jika terkena oleh serangan musuh *method damage* bisa dilihat pada gambar III.48



Gambar III.48 Method CharacterDamage.ApplyDamage

III.4.5.5 Method misiSelesai.misiSelesai()

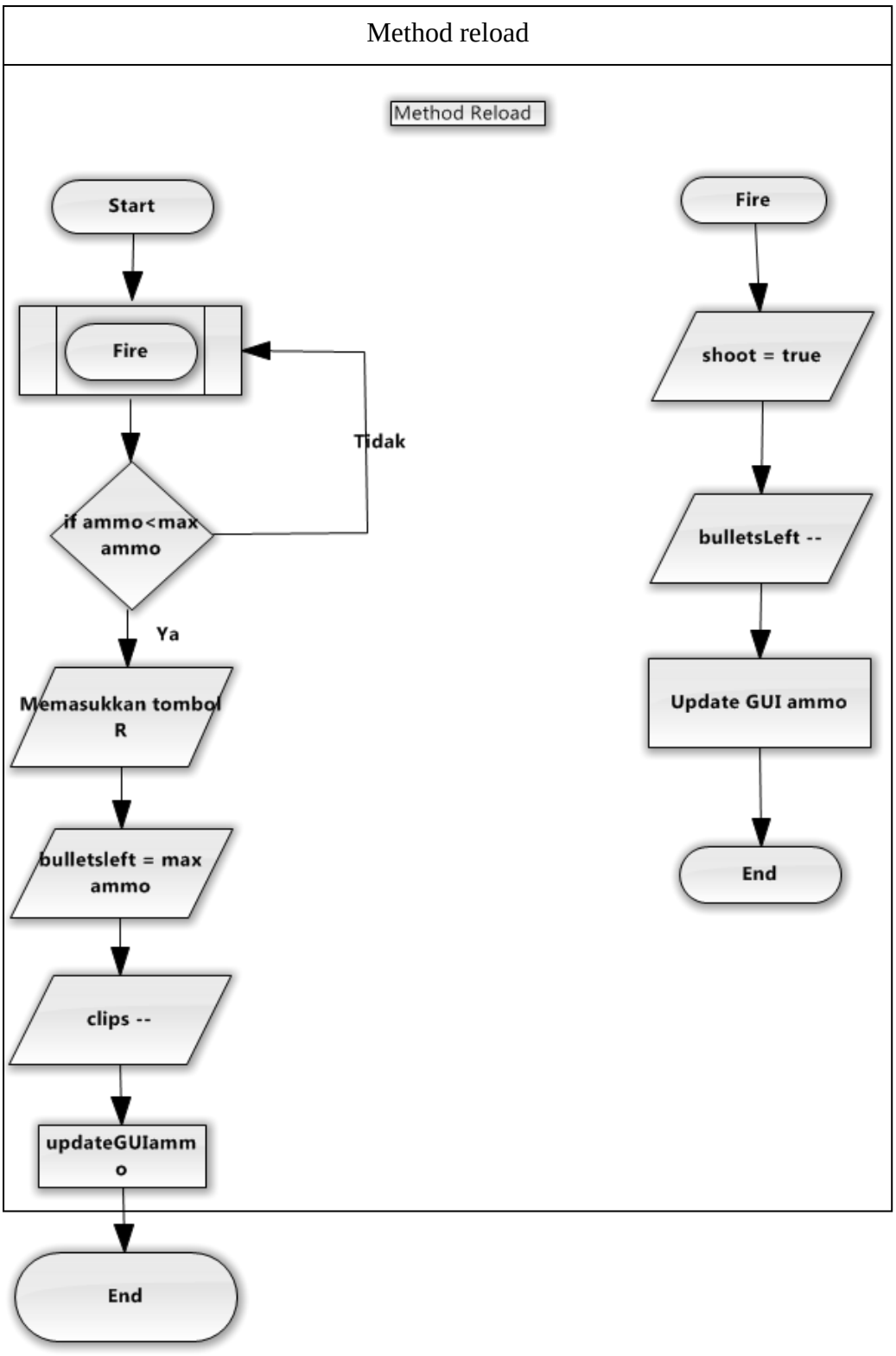
Method ini berada pada beberapa *class* misi selesai *method* ini digunakan untuk melakukan pengecekan terhadap posisi pemain yang berada di arena misi selesai maka akan melakukan *load level* ke *level* selanjutnya. gambar *method* ini bisa dilihat pada gambar III.49



Gambar III.49 Method misiSelesai.MisiSelesai

III.4.5.6 Method *UniversalGun.reload*()

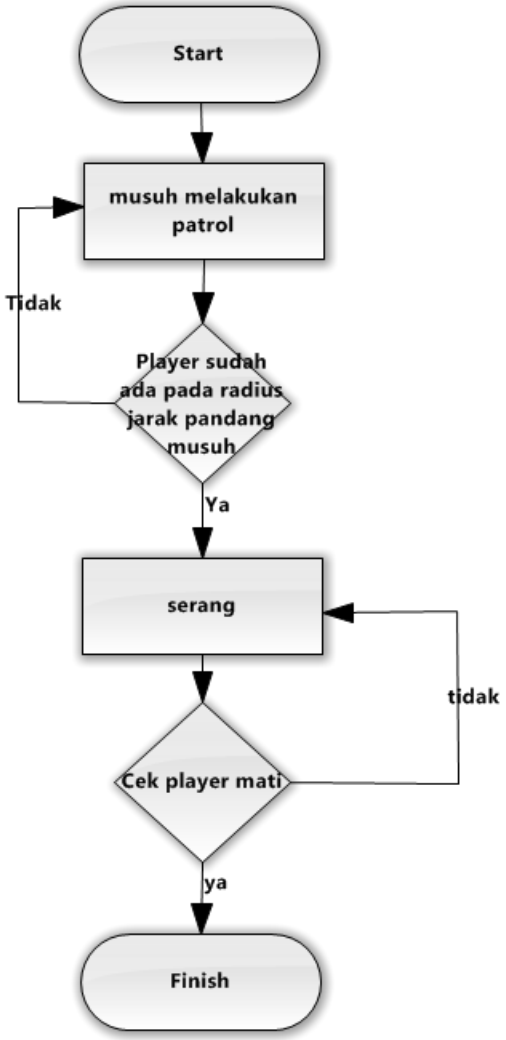
Method ini berada pada *class universal gun,* digunakan untuk melakukan pengisian otomatis terhadap amunisi senjata jika telah habis *method* bisa dilihat pada gambar III.50



Gambar III.50 Method Universal Gun.reload

III.4.5.7 Method AI.attackPlayer()

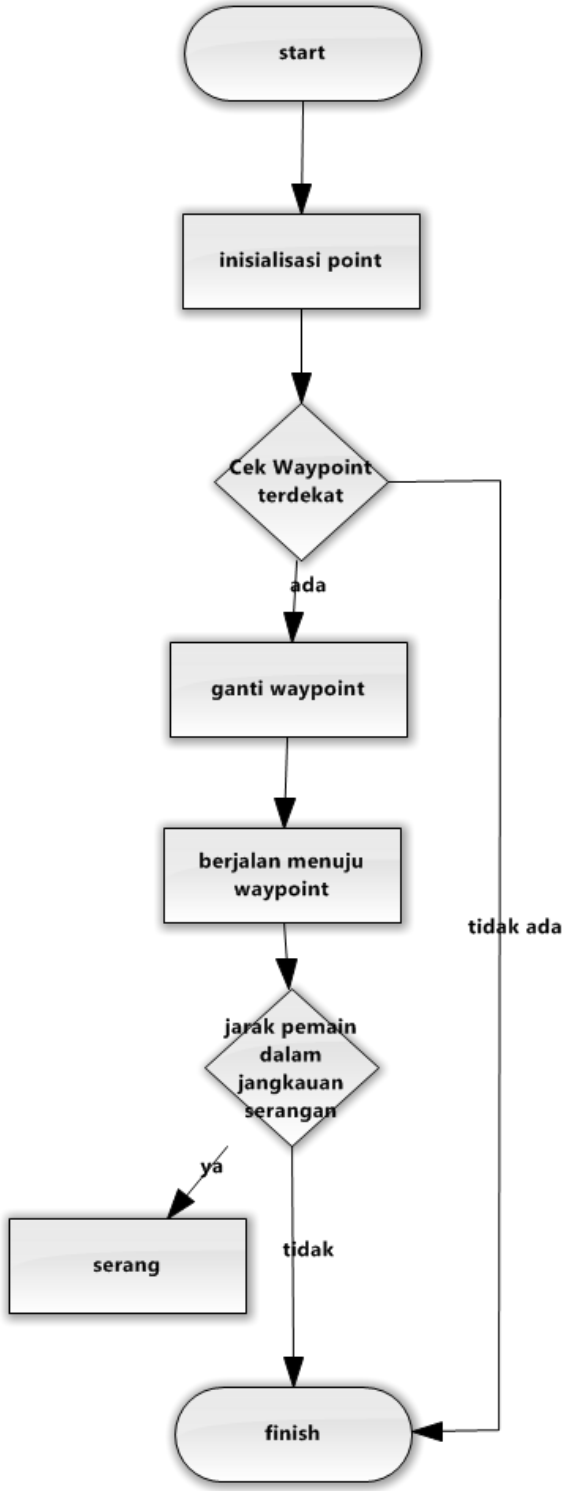
Method ini berada pada *class AI*, digunakan *NPC* untuk menyerang terhadap *player method* bisa dilihat pada gambar III.51



Gambar III.51 Method AI.attackplayer

III.4.5.8 Method AI.*patrol*()

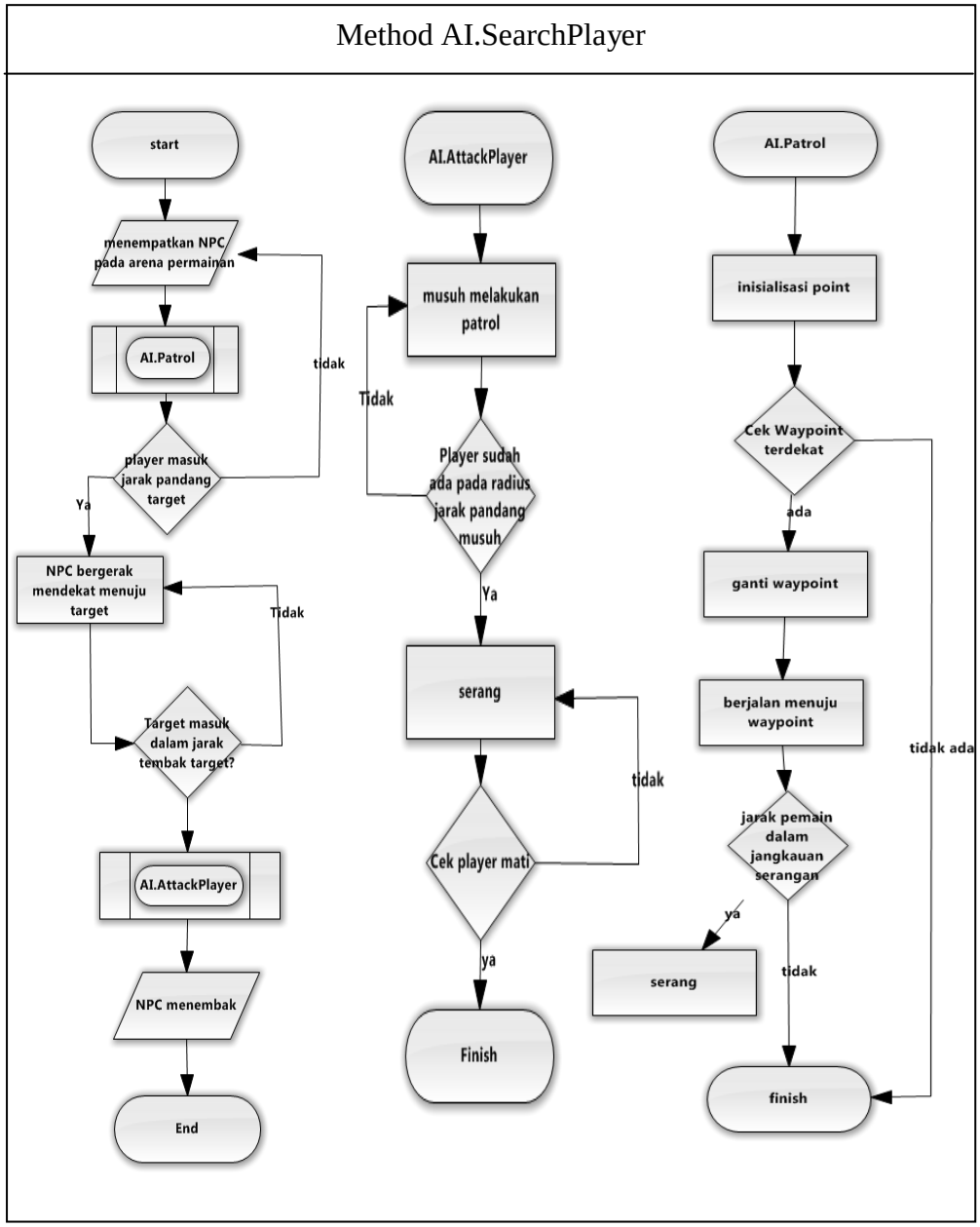
Method ini berada pada *class AI*, digunakan *NPC* untuk melakukan patroli pada titik-titik yang telah ditentukan dengan menggunakan *waypoint. Method* bisa dilihat pada gambar III.52



Gambar III.52 Method AI.patrol

III.4.5.9 Method AI.*searchPlayer*()

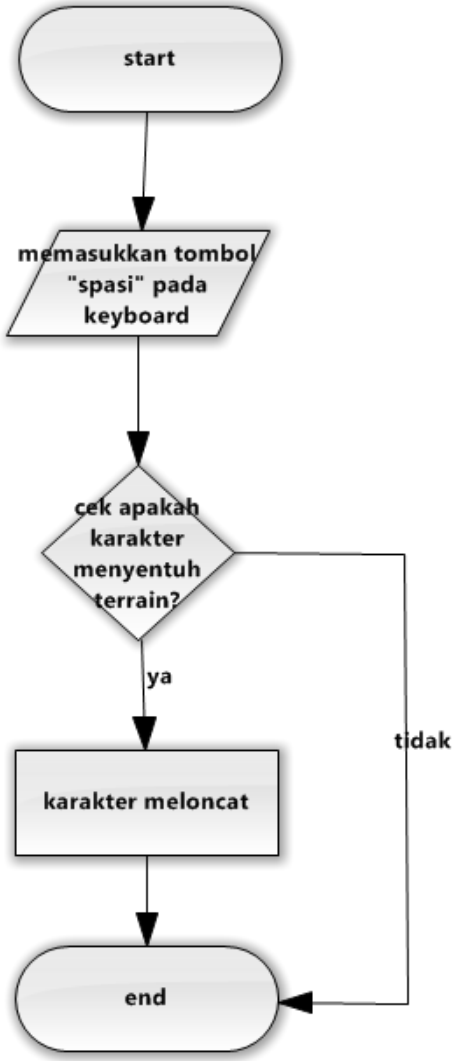
Method ini berada pada *class AI*, digunakan *NPC* untuk melakukan mencari keberadaan player. *Method* bisa dilihat pada gambar III.53



Gambar III.53 Method AI.SearchPlayer

III.4.5.10 Method FPSwalker.FixedUpdate()

Method ini berada pada *class FPSWalker*, *Method* ini untuk menggerakkan karakter melakukan update berupa pergerakan lompatan terhadap karakter ketika karakter telah menyentuh tanah. *Method* bisa dilihat pada gambar III.54



Gambar III.54 Method FPSWalker.FixedUpdate