BAB 3
ANALISIS DAN PERANCANGAN

3.1 Analisis

Pada proses analisis ini mencakup berbagai analisis diantaranya analisis masalah, analisis *game*, analisis masukan, analisis algoritma, analisis kebutuhan perangkat lunak yang mencakup analisis kebutuhan fungsional dan nonfungsional serta perancangan-perancangan dalam pembuatan perangkat lunak.

3.1.1 Analisis Masalah

Game yang bertipe *Real Time Strategy* (RTS) merupakan *game* yang mengharuskan pemain mengatur unit-unit, mengatur *resources* dan mengalahkan lawan. Pada game RTS yang dibuat memiliki konsep bagaimana mengatur perilaku menyerang dan *route finding* untuk mendapatkan prioritas serangan yang dilakukan dengan mendapatkan posisi dan jarak yang tepat untuk menyerang. Logika fuzzy merupakan solusi yang bisa diterapkan pada *game* bertipe *Real Time Strategy* (RTS) untuk mengatur perilaku menyerang dengan pola *auto-attack* dan *route finding*. Pola menyerang pada *game* RTS pada saat ini pada umumnya mengacu pada posisi dan jangkauan tembak atau serangan, sementara itu pada *game* RTS juga ada pola yang dinamakan *route finding*, dimana *route finding* pada saat ini juga digunakan untuk menggerakan unit ke tempat yang diinginkan *player*.

3.1.2 Analisis *Game*

Misi utama dari *game* RTS yang dibuat adalah menyerang musuh, yang menjadi objek atau target sasaran serangannya adalah pasukan musuh dan bangunan musuh. Penggunaan logika fuzzy digunakan untuk mengatur perilaku menyerang dengan pola *auto-attack* dan *route finding* kemudian mendapat posisi dan jarak yang tepat untuk menyerang. Peran *player* pada *game* yang dibuat adalah menggerakan

unit pasukan ke tempat yang diinginkan *player*, kemudian menyerang satu pasukan musuh untuk serangan selanjutnya logika fuzzy yang berperan untuk mengatur perilaku menyerang *auto-attack* dan *route finding* dimana *player* tidak perlu memerintahkan unit pasukan untuk menyerang tetapi unit pasukan akan bergerak dan menyerang secara otomatis apabila musuh berada pada jarak dan jangkauan serangan unit pasukan, maka unit pasukan akan menyerang musuh.

Parameter-parameter yang dimiliki musuh menjadi acuan serangan sebab pasukan akan mendapat prioritas serangan berdasarkan *life* dan *power* yang dimiliki musuh, peran logika fuzzy yaitu untuk menghasilkan pengambilan keputusan dan prioritas serangan untuk pasukan *player*, yang sebelumnya melalui tahapan dari logika fuzzy mulai dari tahap fuzzifikasi sampai tahap defuzzikasi. Sehingga keputusan yang dihasilkan relevan dan efektif untuk menyerang musuh.

3.1.3 Analisis Komponen Pembuatan *Game*

Berikut merupakan komponen-komponen yang digunakan dalam pembuatan aplikasi yang dibuat pada penelitian untuk *game* bertipe *Real Time Strategy* (RTS).

Tabel 3.1 Komponen Pembuatan *Game*

No	Komponen	Keterangan
1	Kecerdasan Buatan (AI)	Menggunakan logika fuzzy untuk mengatur perilaku menyerang dengan menggunakan pola *auto-attack* dan *route finding*.
2	Karakter Pemain dan Musuh	Unit yang dimodelkan dan objek yang digunakan sebagai sasaran serangan yaitu : 1. Pasukan (*Player*), dan 2. Musuh (Pasukan Musuh dan Bangunan Musuh)
3	Skenario *Game*	Mengubah parameter dan posisi musuh dengan

		tujuan menerapkan algoritma logika fuzzy untuk mengatur perilaku menyerang dan *route finding* pasukan *player* untuk mendapat prioritas serangan yang dilakukan dengan mendapat posisi dan jarak yang tepat untuk menyerang berdasarkan *life* dan *power* yang dimiliki musuh.

3.1.4 Analisis Masukan

Analisis masukan yang dilakukan dalam penelitian ini adalah menganalisis parameter dalam algoritma logika *fuzzy* pada *Game* yang bertipe *Real Time Strategy* (RTS), parameter tersebut antara lain jarak, nyawa dan kekuatan yang dimiliki pasukan *player*, yang nantinya parameter- parameter tersebut berpengaruh pada prioritas serangan yang dilakukan pasukan *player*. Berikut rentang nilai masukan yang diperbolehkan untuk dimasukan ke dalam parameter jaak, nyawa dan kekuatan musuh pada tabel 3.2.

Tabel 3.2 Parameter Masukan Musuh dan Rentang Nilai Masukannya

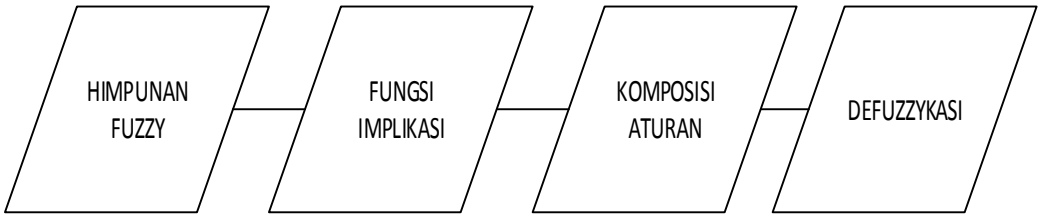
No	Parameter	Rentang Nilai
1	*Range* (Jarak)	10 – 80
2	*Life* (Nyawa)	1 – 100
3	*Power* (Kekuatan)	0 – 100

3.1.5 Analisis Algoritma

Analisis algoritma yang dilakukan dalam penelitian ini adalah menerapkan logika fuzzy pada perilaku menyerang dengan menggunakan pola *auto-attack* dan *route finding* pada unit pasukan *player*. Logika fuzzy yang digunakan adalah logika fuzzy Mamdani.

3.1.5.1 Analisis Logika Fuzzy

Logika fuzzy pada *game* yang dibuat diterapkan pada unit pasukan, untuk mengatur perilaku menyerang dan *route finding*, dengan tujuan digunakannya logika fuzzy agar keputusan yang unit lakukan menjadi relevan dan efektif. Pada *game* yang dibuat ada beberapa tahapan yang digunakan dalam penerapan logika fuzzy untuk mengatur perilaku menyerang dan *route finding* unit pasukan *player* menggunakan logika fuzzy Mamdani. Berikut adalah gambar proses dari logika fuzzy Mamdani.



Gambar 3.1 Proses Logika Fuzzy Mamdani

Adapun tahapan-tahapan dari proses logika fuzzy Mamdani antara lain :

1. Pembentukan himpunan fuzzy

   Terdapat empat parameter yang digunakan dalam pembentukan himpunan fuzzy antara lain :

   a. Parameter *Range* (Jarak) sebagai *input*.
   b. Parameter *Life* (Nyawa) sebagai *input*.
   c. Parameter *Power* (Kekuatan) sebagai *input*.
   d. Parameter Target sebagai *output*.

2. Aplikasi Fungsi Implikasi

   Terdapat tiga parameter implikasi yang digunakan antara lain :

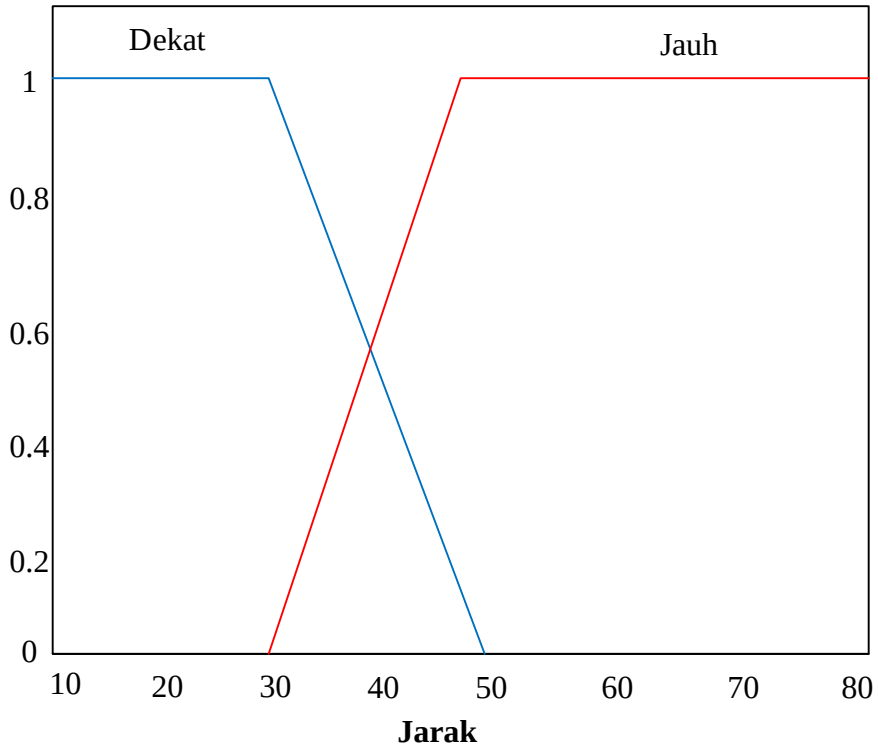
   a. *Range* (Jarak) : Dekat dan Jauh.
   b. *Life* (Nyawa) : Lemah, dan Kuat.
   c. *Power* (Kekuatan) : Lemah, dan Kuat

3. Komposisi Aturan

Pada kompisisi aturan yang digunakan menggunakan inferensi *min*. Berikut adalah komposisi aturan untuk logika fuzzy yang digunakan.

a. Parameter *Range* (Jarak)

Parameter *Range* (Jarak) dibagi menjadi tiga bagian yaitu Dekat dan Jauh. Rentang nilai dari parameter jarak yaitu antara 10–80. Berikut adalah derajat keanggotaan dari parameter *Range* (Jarak) pada gambar 3.2.



Gambar 3.2 Derajat Keanggotaan *Range* (Jarak)

Pada parameter jarak (Dekat) fuzzifikasinya dihitung berdasarkan grafik yang berbentuk trapesium sebelah kiri. Berikut adalah persamaan untuk parameter jarak (Dekat) pada persamaan 3.1.

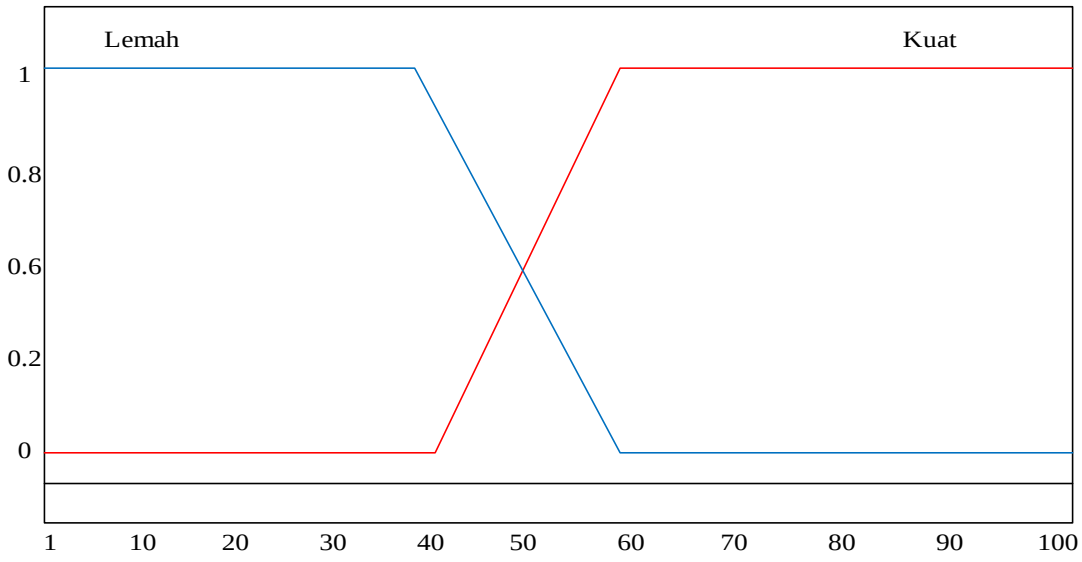
$$\mu\text{Dekat}[x] = \begin{cases} 1; x \leq 30 \\ \frac{x-40}{20}\ ;\ x;\ 30 \leq x \leq 50 \\ 0;\ x \geq 50 \end{cases} \qquad (3.1)$$

Pada parameter jarak (Jauh) fuzzifikasinya dihitung berdasarkan grafik yang berbentuk trapesium sebelah kanan. Berikut adalah persamaan untuk parameter jarak (Jauh) pada persamaan 3.2.

$$\mu\text{Jauh}[x] = \begin{cases} 0; x \leq 50 \\ \frac{x-30}{20}\ ;\ x;\ 50 \leq x \leq 80 \\ 1;\ x \geq 80 \end{cases} \qquad (3.2)$$

b. Parameter *Life* (Nyawa)

Parameter *Life* (Nyawa) dibagi menjadi dua bagian yaitu : Lemah dan Kuat. Rentang nilai dari parameter *Life* (Nyawa) yaitu antara 1-100. Berikut adalah derajat keanggotaan *Life* (Nyawa) pada gambar 3.3



Gambar 3.3 Derajat Keanggotaan *Life* (Nyawa)

Pada parameter nyawa (Lemah) fuzzifikasinya dihitung berdasarkan grafik yang berbentuk trapesium sebelah kiri. Berikut adalah persamaan untuk parameter nyawa (Lemah) pada persamaan 3.3.

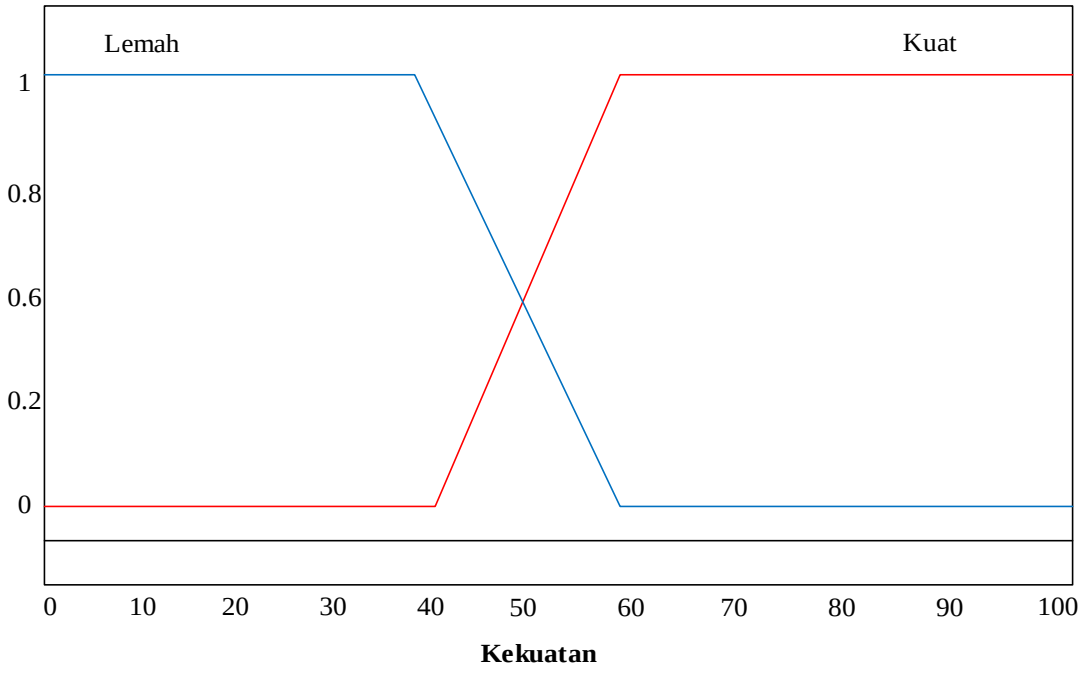
$$\mu\text{Lemah}[x] = \begin{cases} 1; x \leq 40 \\ \frac{x-60}{20}; x; 40 \leq x \leq 60 \\ 0; x \geq 60 \end{cases} \qquad (3.3)$$

Pada parameter nyawa (Kuat) fuzzifikasinya dihitung berdasarkan grafik yang berbentuk trapesium sebelah kanan. Berikut adalah persamaan untuk parameter nyawa (Kuat) pada persamaan 3.4.

$$\mu\text{Kuat}[x] = \begin{cases} 0; x \leq 60 \\ \frac{x-40}{20}; x; 60 \leq x \leq 100 \\ 1; x \geq 100 \end{cases} \qquad (3.4)$$

c. Parameter *Power* (Kekuatan)

Parameter *Power* (Kekuatan) dibagi menjadi dua bagian yaitu : Lemah dan Kuat. Rentang nilai parameter *Power* (Kekuatan) yaitu antara 0-100. Berikut adalah derajat keanggotaan parameter *Power* (Kekuatan) pada gambar 3.4.



Gambar 3.4 Derajat Keanggotaan *Power* (Kekuatan)

Pada parameter kekuatan (Lemah) fuzzifikasinya dihitung berdasarkan grafik yang berbentuk trapesium sebelah kiri. Berikut adalah persamaan untuk parameter kekuatan (Lemah) pada persamaan 3.5.

$$\mu\text{Lemah}[x] = \begin{cases} 1; x \leq 40 \\ \frac{x-60}{20}; x; 40 \leq x \leq 60 \\ 0; x \geq 60 \end{cases} \qquad (3.5)$$

Pada parameter kekuatan (Kuat) fuzzifikasinya dihitung berdasarkan grafik yang berbentuk trapesium sebelah kanan. Berikut adalah persamaan untuk parameter kekuatan (Lemah) pada persamaan 3.6.

$$\mu\text{Kuat}[x] = \begin{cases} 0; x \leq 60 \\ \frac{x-40}{20}; x; 60 \leq x \leq 100 \\ 1; x \geq 100 \end{cases} \qquad (3.6)$$

d. Kaidah / *Rules* Fuzzy

Berikut adalah aturan perilaku yang berlaku berdasarkan kaidah atau *rules* yang diterapkan dalam penentuan perilaku menyerang dari unit pasukan *player* untuk mendapat prioritas serangan diantaranya adalah :

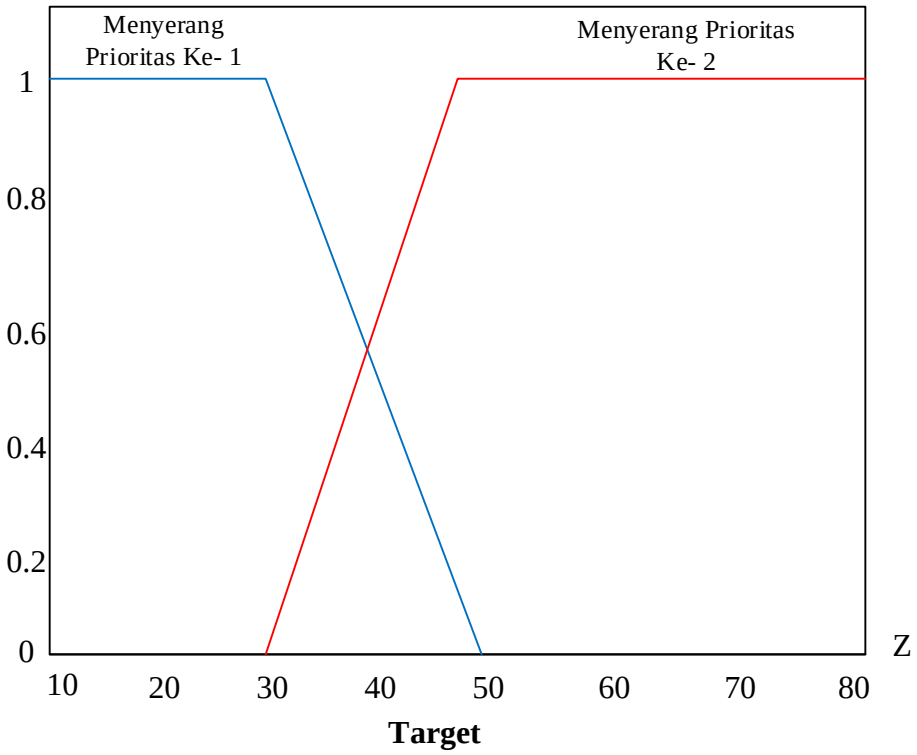
1) *If* (*Range* is Dekat) and (*Life* is Lemah) and (*Power* is Lemah) then (Menyerang Prioritas Ke-1).
2) *If* (*Range* is Dekat) and (*Life* is Lemah) and (*Power* is Kuat) then (Menyerang Prioritas Ke-2).
3) *If* (*Range* is Dekat) and (*Life* is Kuat) and (*Power* is Lemah) then (Menyerang Prioritas Ke-1)
4) *If* (*Range* is Dekat) and (*Life* is Kuat) and (*Power* is Kuat) then (Menyerang Prioritas Ke-2).
5) *If* (*Range* is Jauh) and (*Life* is Lemah) and (*Power* is Lemah) then (Menyerang Prioritas Ke-1).
6) *If* (*Range* is Jauh) and (*Life* is Lemah) and (*Power* is Kuat) then (Menyerang Prioritas Ke-2).
7) *If* (*Range* is Jauh) and (*Life* is Kuat) and (*Power* is Lemah) then (Menyerang Prioritas Ke-1).
8) *If* (*Range* is Jauh) and (*Life* is Kuat) and (*Power* is Kuat) then (Menyerang Prioritas Ke-2).

e. Defuzzikasi

Merupakan suatu proses untuk memperoleh nilai *crisp* sebagai solusi dari sistem fuzzy menggunakan metode centroid. Metode centroid akan mencari nilai z yang nantinya menjadi pengambilan keputusan pasukan *player* untuk menyerang prioritas utama atau prioritas kedua berdasarkan rumus 2.1.

Pada parameter target yaitu menyerang prioritas ke-1 dan menyerang prioritas ke-2 mempunyai rentang nilai antara 10 – 80 dijelaskan sebagai berikut :

1. Menyerang Prioritas Ke-1 = 10 – 30
2. Menyerang Prioritas Ke-2 = 50 – 80



Gambar 3.5 Derajat Keanggotan Target

Untuk menghitung target menyerang prioritas ke-1 menggunakan perhitungan dengan rumus 2.1. Berikut perhitungan untuk target menyerang prioritas ke-1.

$$30 – (\min[i] * 8)$$

Sedangkan untuk menghitung target menyerang prioritas ke-2 menggunakan perhitungan dengan rumus 2.1, Berikut perhitungan untuk target menyerang prioritas ke-2

$$50 - (\min[i] * 8)$$

Contoh Kasus Pada *Game*

Misal apabila *Range* (Jarak) antara pasukan *player* dan pasukan musuh memiliki nilai [31], *Life* (Nyawa) yang dimiliki pasukan *player* memiliki nilai [45], dan *Power* (Kekuatan) yang dimiliki pasukan *player* memiliki nilai [25], maka tahapan-tahapan yang menentukan *output* sesuai dengan aturan yang sudah dibuat akan dijelaskan sebagai berikut :

1. Pembentukan Himpunan Fuzzy

a. Pembentukan himpunan fuzzy dengan menghitung *Range* (Jarak) setiap parameter yaitu Dekat dan Jauh. Apabila nilai *Range* (Jarak) antara pasukan *player* dan pasukan musuh memiliki nilai [31] untuk fungsi keanggotaan Dekat dihitung berdasarkan persamaan 3.1, sedangkan untuk fungsi keanggotaan Jauh dihitung berdasarkan persamaan 3.2.

$$\mu \text{Dekat } [31] = \frac{31-50}{20} = 0.95$$

$$\mu \text{Jauh } [31] = 31 \leq 50 = 0$$

Maka diperoleh nilai untuk fungsi keanggotaan untuk parameter *Range* (Jarak) yaitu :

1. Fungsi Keanggotaan Dekat = 0.95
2. Fungsi Keanggotaan Jauh = 0

b. Pembentukan himpunan fuzzy dengan menghitung *Life* (Nyawa) setiap parameter yaitu Lemah dan Kuat. Apabila nilai *Life* (Nyawa) yang dimiliki musuh memiliki nilai [45] untuk fungsi keanggotaan Lemah dihitung berdasarkan persamaan 3.3, sedangkan untuk fungsi keanggotaan Kuat dihitung berdasarkan persamaan 3.4.

$$\mu \text{Lemah } [45] = \frac{45-60}{20} = 0.75$$

$$\mu \text{Kuat } [45] = 45 \leq 60 = 0$$

Maka diperoleh nilai untuk fungsi keanggotaan untuk parameter *Life* (Nyawa) yaitu :

1. Fungsi Keanggotaan Lemah = 0.75
2. Fungsi Keanggotaan Kuat = 0

c. Pembentukan himpunan fuzzy dengan menghitung *Power* (Kekuatan) setiap parameter yaitu Lemah dan Kuat. Apabila nilai *Power* (Kekuatan) yang musuh memiliki nilai [25] untuk fungsi keanggotaan Lemah dihitung berdasarkan persamaan 3.5, sedangkan untuk fungsi keanggotaan Kuat dihitung berdasarkan persamaan 3.6.

$$\mu \text{Lemah } [25] = 25 \leq 40 = 1$$
$$\mu \text{Kuat } [25] = 25 \leq 60 = 0$$

Maka diperoleh nilai untuk fungsi keanggotaan untuk parameter *Power* (Kekuatan) yaitu :

1. Fungsi Keanggotaan Lemah = 1
2. Fungsi Keanggotaan Kuat = 0

2. Aplikasi Fungsi Implikasi dan Komposisi Aturan

Setelah fungsi keanggotaan didapat, maka nilai-nilai tersebut harus dibandingkan sesuai dengan *rule* yang ada, tujuannya untuk mencari nilai minimum untuk setiap *rule* yang dihitung. Berikut adalah fungsi implikasi yang dibandingkan dengan *rule* yang telah ditentukan antara lain :

1. *If* (*Range* is Dekat) and (*Life* is Lemah) and (*Power* is Lemah) then (Menyerang Prioritas Ke-1)
   Range (Dekat) = 0.95
   Life (Lemah) = 0.75

Power (Lemah) = 1

Min (0.95, 0.75, 1) = 0.75

2. *If* (*Range* is Dekat) and (*Life* is Lemah) and (*Power* is Kuat) then (Menyerang Prioritas Ke-2).

   Range (Dekat) = 0.95

   Life (Lemah) = 0.75

   Power (Kuat) = 0

   Min (0.95, 0.75, 0) = 0

3. *If* (*Range* is Dekat) and (*Life* is Kuat) and (*Power* is Lemah) then (Menyerang Prioritas Ke-1).

   Range (Dekat) = 0.95

   Life (Kuat) = 0

   Power (Lemah) = 1

   Min (0,95, 0, 1) = 0

4. *If* (*Range* is Dekat) and (*Life* is Kuat) and (*Power* is Kuat) then (Menyerang Prioritas Ke-2)

   Range (Dekat) = 0.95

   Life (Kuat) = 0

   Power (Kuat) = 0

   Min (0,95, 0, 0) = 0

5. *If* (*Range* is Jauh) and (*Life* is Lemah) and (*Power* is Lemah) then (Menyerang Prioritas Ke-1).

   Range (Jauh) = 0

   Life (Lemah) = 0.75

   Power (Lemah) = 1

   Min (0, 0,75, 1) = 0

6. *If* (*Range* is Jauh) and (*Life* is Lemah) and (*Power* is Kuat) then (Menyerang Prioritas Ke-2).

   Range (Jauh) = 0

Life (Lemah) = 0.75

Power (Kuat) = 0

Min (0, 0,75, 0) = 0

7. *If* (*Range* is Jauh) and (*Life* is Kuat) and (*Power* is Lemah) then (Menyerang Prioritas Ke-1).

   Range (Jauh) = 0

   Life (Kuat) = 0

   Power (Lemah) = 1

   Min (0, 0, 1) = 0

8. *If* (*Range* is Jauh) and (*Life* is Kuat) and (*Power* is Kuat) then (Menyerang Prioritas Ke-2).

   Range (Jauh) = 0

   Life (Kuat) = 0

   Power (Kuat) = 0

   Min (0, 0, 0) = 0

3. Defuzzikasi

Setelah nilai keanggotaan didapat melalui proses fungsi implikasi maka langkah berikutnya adalah menentukan himpunan nilai z dari setiap *rule*. Berikut adalah proses menentukan nilai z menggunakan rumus perhitungan 2.1 pada tabel 3.3.

Tabel 3.3 Proses Menentukan Nilai Z

Rule	**Perhitungan**	**Target Menyerang**
1	$Z = 30 - (\min[t] * 8)$ Z = 30 – (0,75 * 8) = 23.25	Prioritas Ke-1
2	$Z = 30 - (\min[t] * 8)$ Z = 30 – (0,95 * 8) = 22.8	Prioritas Ke-2
3	$Z = 30 - (\min[t] * 8)$ Z = 30 – (0 * 8) = 30	Prioritas Ke-1

4	$Z = 50 - (\min[t] * 8)$ Z = 50 – (0 * 8) = 50	Prioritas Ke-2
5	$Z = 30 - (\min[t] * 8)$ Z = 30 – (0 * 8) = 30	Prioritas Ke-1
6	$Z = 30 - (\min[t] * 8)$ Z = 30 – (0 * 8) = 30	Prioritas Ke-2
7	$Z = 30 - (\min[t] * 8)$ Z = 30 – (0 * 8) = 30	Prioritas Ke-1
8	$Z = 50 - (\min[t] * 8)$ Z = 50 – (0 * 8) = 50	Prioritas Ke-2

Setelah mendapatkan nilai dari setiap *rules* maka langkah selanjutnya adalah menentukan nilai defuzzifikasi dengan menggunakan rumus metode centroid pada rumus 2.1.

$$Z = \frac{0.75\,x\,23.25 + 0.95\,x\,22.8 + 0\,x\,30 + 0\,x\,50 + 0\,x\,30 + 0x30 + 0\,x\,30 + 0\,x\,50}{0.75 + 0.95 + 0 + 0 + 0 + 0 + 0 + 0}$$

$$Z = \frac{55.1475}{1.65} = 33.42$$

Jadi nilai yang dihasilkan berdasarkan *rule*/aturan yang sudah dibentuk sebelumnya. Sesuai dengan contoh dari perhitungan yang dihasilkan dari nilai *Range* (Jarak) [31], nyawa [45] dan Kekuatan [25] yang dimiliki pasukan *player* dengan nilai Z yang didapatkan yaitu 33.42 maka keputusannya adalah target menyerang prioritas Ke-1.

3.2 Analisis Kebutuhan Perangkat Lunak

Pada tahap ini meliputi elemen-elemen yang diperlukan dalam mengimplementasikan logika fuzzy yaitu terdiri dari kebutuhan non-fungsional dan

kebutuhan fungsional. Spesifikasi kebutuhan perangkat lunak akan dibagi kedalam dua bagian yaitu Kebutuhan Fungsional dan Spesifikasi Kebutuhan Non-Fungsional.

3.2.1 Analisis Kebutuhan Non-Fungsional

Analisis kebutuhan non-fungsional adalah langkah pembangunan perangkat lunak dengan menganalisis sumber daya yang akan dibutuh dan digunakan serta menggunakan perangkat lunak yang digunakan untuk membangun sistem. Perangkat keras dan perangkat lunak yang dimiliki harus sesuai dengan kebutuhan, sehingga dapat ditentukan kompabilitas aplikasi yang dibangun terhadap sumber daya yang ada.

3.2.1.1 Analisis Perangkat Keras

Perangkat keras atau *hardware* merupakan salah satu hal yang penting karena tanpa *hardware* yang memenuhi syarat, sistem yang akan dibuat tidak akan dapat berjalan. Perangkat keras minimum yang digunakan :

1. Processor Dual Core 1,80 Ghz
2. RAM 1 GB
3. VGA 512 MB
4. Hard disk 80 GB
5. Monitor
6. *Keyboard*

3.2.1.2 Analisis Perangkat Lunak

Perangkat lunak atau *software* merupakan hal yang terpenting dalam mendukung kinerja sebuah sistem. Perangkat lunak digunakan dalam sebuah sistem merupakan perintah-perintah yang diberikan kepada perangkat keras agar dapat saling berinteraksi diantara keduanya. Perangkat lunak yang dibutuhkan untuk membangun aplikasi *game* ini adalah sebagai berikut :

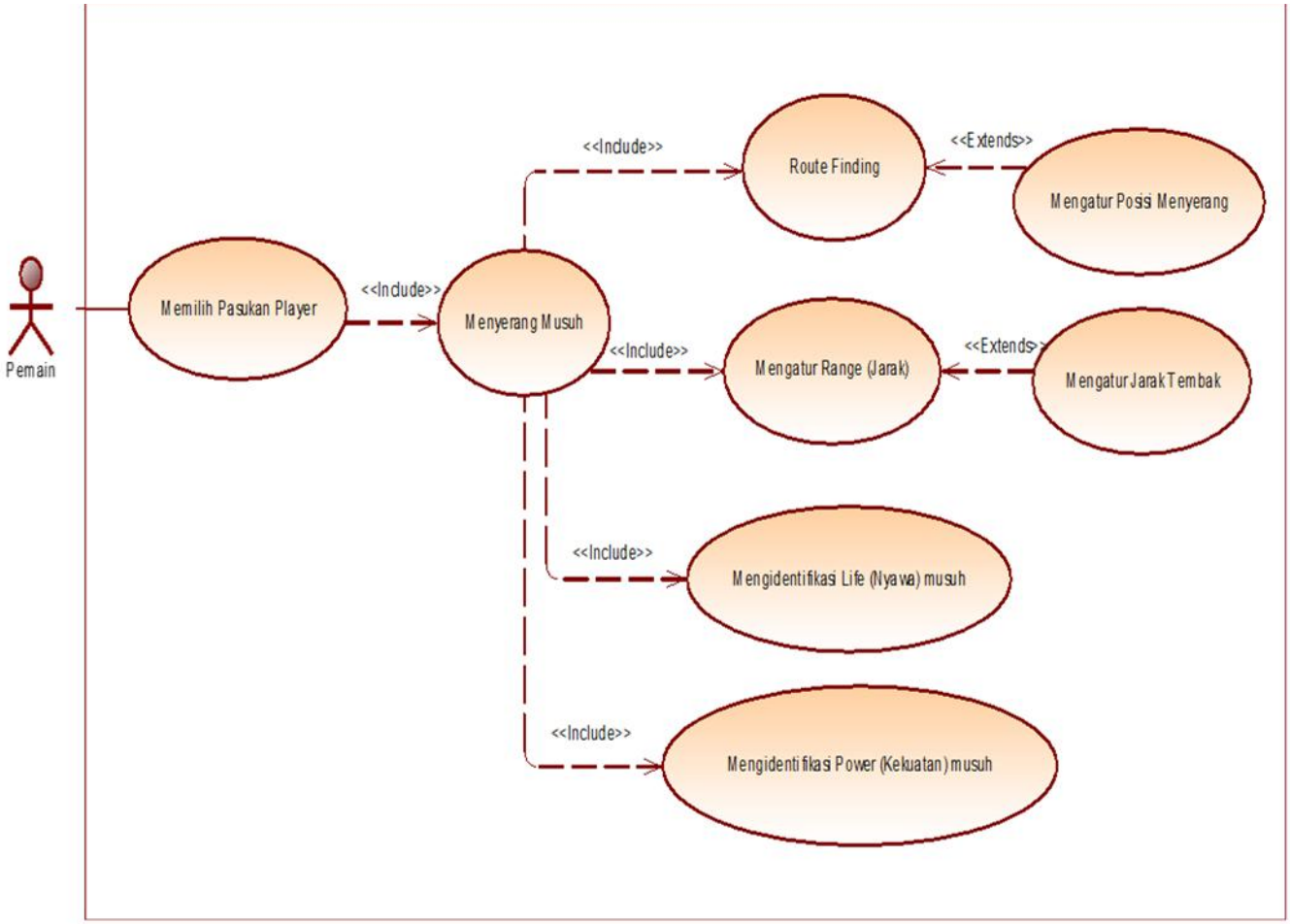
1. Sistem operasi Microsoft Windows 8.
2. Unity sebagai editor.

3.2.2 Analisis Kebutuhan Fungsional

Tahap pertama yang dilakukan dalam melakukan analisis berorientasi objek adalah salah satunya menggunakan *Unified Modelling Language* (UML) dengan menentukan aktor yang ada pada sistem. Aktor pada UML yaitu dengan menampilkan peran atau *roles* dari sistem yang dapat dikembangkan pada aspek perangkat lunak, perangkat keras dan lain sebagainya.

3.2.2.1 *Use Case Diagram*

Diagram *use case* akan sangat membantu analisis sistem dan juga program untuk merumuskan spesifikasi kebutuhan untuk sistem atau perangkat lunak yang akan dibangun maupun untuk dikembangkan. Dari hasil analisis *game* yang dibangun maka diagran *use case* seperti pada gambar 3.6.



Gambar 3.6 *Use Case Diagram*

3.2.2.2 Identifikasi Aktor

Mengidentifikasi aktor adalah sangat penting, sepenting mengidentifikasi kelas-kelas, struktur, relasi, atribut dan perilaku. Kata aktor dalam konteks UML adalah menampilkan peran pengguna sistem. Identifikasi aktor diterangkan pada tabel 3.4.

Tabel 3.4 Identifikasi Aktor

Aktor	Deskripsi
Pemain	Pemain atau Orang yang memainkan *game*,

3.2.2.3 Identifikasi *Use Case*

Setelah mengidentifikasi aktor kemudian mengidentifikasi dari *use case* dengan melihat pekerjaan-pekerjaan dan fungsi-fungsi yang aktor lakukan, secara representatif yang mencerminkan layanan spesifik atau fungsionalitas yang diharapkan dari sistem atau perangkat lunak yang dibangun serta mendeskripsikan masing-masing kelas dalam konteks *use case* agar tidak muncul pemahaman ganda (ambigu). Berikut identifikasi *use case* pada tabel 3.5.

Tabel 3.5 Identifikasi *Use Case*

No	*Use Case*	Deskripsi
1	Memilih Pasukan Player	Pemain memilih pasukan player
2	Menyerang Musuh	Pasukan player menyerang musuh pertama yang dipilih pemain, selanjutnya menyerang musuh secara otomatis dengan menyerang prioritas ke-1 dan prioritas ke-2.
3	*Route Finding*	*Route finding* digunakan pasukan untuk bergerak ke tempat yang diinginkan pemain kemudian mendapat posisi untuk menyerang
4	Mengatur *Range* (Jarak)	Pasukan mengatur jarak dan jangkauan tembak
5	Mengidentifikasi *Life* (Nyawa) musuh	Pasukan mengidentifikasi *Life* (Nyawa) yang dimiliki oleh musuh
6	Mengidentifikasi *Power* (Kekuatan) musuh	Pasukan mengidentifikasi *Power* (Kekuatan) yang dimiliki oleh musuh
7	Mengatur posisi menyerang	Pasukan mengatur posisi untuk menyerang musuh
8	Mengatur jarak tembak	Pasukan mengatur jarak tembak

3.2.2.4 Skenario *Use Case*

Skenario use case berisi tentang *flow of event* atau alur dari *use case* utama, yang menggambarkan urutan interaksi Aktor dengan *use case* dari awal sampai akhir. Skenario *use case* pada *game* yang dibuat adalah :

3.2.2.5 Skenario *Use Case* Memilih Pasukan Player

Skenario *use case* memilih pasukan player, digunakan pemain untuk memilih pasukan player yang digunakan. Skenario *use case* memilih pasukan seperti pada tabel 3.6.

Tabel 3.6 Skenario *Use Case* Memilih Pasukan Player

Identifikasi	
Nama *Use Case*	Memilih Pasukan Player
Aktor	Pemain
Tujuan	Setfocus ke pasukan player
Kondisi Awal	Pasukan pada posisi *default* atau awal
Kondisi Akhir	Pemain sudah memilih pasukan player kemudian *setfocus* ke pasukan player
Skenario	
Aksi Aktor	**Sistem**
a. Memilih pasukan player	
	b. *Setfocus* ke pasukan player

3.2.2.6 Skenario *Use Case* Menyerang Musuh

Skenario *use case* menyerang musuh, bertujuan agar pasukan player menyerang musuh tanpa perintah dari pemain dan menyerang musuh secara *auto-attack* kemudian mendapat prioritas serangan dengan menyerang prioritas ke-1 dan prioritas ke-2. Skenario *use case* menyerang musuh seperti pada tabel 3.7.

Tabel 3.7 Skenario *Use Case* Menyerang Musuh

Identifikasi	
Nama *Use Case*	Menyerang Musuh
Aktor	Pemain
Tujuan	Menyerang musuh Prioritas ke-1 dan prioritas ke-2
Kondisi Awal	Pasukan pada posisi *default* atau awal
Kondisi Akhir	Pasukan menyerang musuh dengan prioritas ke-1 dan prioritas ke-2
Skenario	
Aksi Aktor	**Sistem**
a. Memilih pasukan player	
	b. *Setfocus* ke pasukan player
c. Memilih musuh yang diserang pertama	
	d. Pasukan player menyerang musuh yang dipilih pemain
	e. Pasukan player mengidentifikasi *life*, *power* serta *range* musuh yang berada pada jangkauan tembak pasukan
	f. Menyerang Prioritas Ke-1 dan Menyerang Prioritas Ke-2

3.2.2.7 Skenario *Use Case Route Finding*

Skenario *use case route finding* digunakan agar pasukan player mendapatkan posisi yang tepat untuk menyerang. Skenario *use case route finding* seperti pada tabel 3.8.

Tabel 3.8 Skenario *Use Case Route Finding*

Identifikasi	
Nama *Use Case*	*Route Finding*
Aktor	Pemain
Tujuan	Pasukan mendapat posisi menyerang
Kondisi Awal	Pasukan pada posisi *default* atau awal
Kondisi Akhir	Pasukan mendapatkan posisi yang tepat untuk menyerang
Skenario	
Aksi Aktor	**Sistem**
a. Memilih pasukan player	
	b. *Setfocus* ke pasukan player
c. Memilih musuh yang diserang pertama	
	d. Pasukan menyerang musuh yang dipilih pemain
	e. Pasukan player mengidentifikasi *life*, *power* serta *range* musuh yang berada pada jangkauan tembak pasukan
	f. Pasukan player mendapatkan posisi yang tepat untuk menyerang

3.2.2.8 Skenario *Use Case* Mengatur *Range* (Jarak)

Skenario *use case* mengatur *Range* (Jarak) digunakan pasukan player untuk mengatur jarak untuk menyerang musuh. Skenario *use case* mengatur *Range* (Jarak) seperti pada tabel 3.9.

Tabel 3.9 Skenario *Use Case* Mengatur *Range* (Jarak)

Identifikasi	
Nama *Use Case*	Mengatur *Range* (Jarak)
Aktor	Pemain
Tujuan	Mengatur jarak untuk menyerang
Kondisi Awal	Pasukan pada posisi *default* atau awal
Kondisi Akhir	Pasukan mengatur jarak untuk menyerang
Skenario	
Aksi Aktor	**Sistem**
a. Memilih pasukan player	
	b. *Setfocus* ke pasukan player
c. Memilih musuh yang diserang pertama	
	d. Pasukan player menyerang musuh yang dipilih pemain
	e. Pasukan player mengidentifikasi *range* musuh yang berada pada jangkauan tembak pasukan
	f. Pasukan player mengatur jarak untuk menyerang musuh

3.2.2.9 Skenario *Use Case* Mengidentifikasi *Life* (Nyawa) Musuh

Skenario *use case* mengidentifikasi *Life* (Nyawa) musuh digunakan pasukan player untuk mendapatkan prioritas serangan dengan mengetahui nyawa yang dimiliki musuh. Skenario *use case* mengidentifikasi *Life* (Nyawa) musuh seperti pada tabel 3.10.

Tabel 3.10 Skenario *Use Case* Mengidentifikasi *Life* (Nyawa) Musuh

Identifikasi	
Nama *Use Case*	Mengidentifikasi *Life* (Nyawa) Musuh
Aktor	Pemain
Tujuan	Pasukan dapat mengetahui *life* yang dimiliki musuh
Kondisi Awal	Pasukan pada posisi *default* atau awal
Kondisi Akhir	Pasukan mengetahui *life* yang dimiliki musuh
Skenario	
Aksi Aktor	**Sistem**
a. Memilih pasukan player	
	b. *Setfocus* ke pasukan player
c. Memilih musuh yang diserang pertama	
	d. Pasukan player menyerang musuh yang dipilih pemain
	e. Pasukan player mengidentifikasi *life* musuh yang berada pada jangkauan tembak pasukan
	f. Pasukan player mengetahui *life* yang dimiliki musuh

3.2.2.10 Skenario *Use Case* Mengidentifikasi *Power* (Kekuatan) Musuh

Skenario *use case Power* (Kekuatan) musuh digunakan pasukan player untuk mendapatkan prioritas serangan dengan mengetahui kekuatan yang dimiliki musuh. Skenario *use case* mengidentifikasi *Power* (Kekuatan) musuh seperti pada tabel 3.11.

Tabel 3.11 Skenario *Use Case* Mengidentifikasi *Power* (Kekuatan) Musuh

Identifikasi	
Nama *Use Case*	Mengidentifikasi *Power* (Kekuatan) Musuh
Aktor	Pemain
Tujuan	Pasukan player dapat mengetahui *power* yang dimiliki musuh
Kondisi Awal	Pasukan player pada posisi *default* atau awal
Kondisi Akhir	Pasukan player mengetahui *power* musuh
Skenario	
Aksi Aktor	**Sistem**
a. Memilih pasukan player	
	b. *Setfocus* ke pasukan player
c. Memilih musuh yang diserang pertama	
	d. Pasukan player menyerang musuh yang dipilih pemain
	e. Pasukan player mengidentifikasi *power* musuh yang berada pada jangkauan tembak pasukan
	f. Pasukan player mengetahui *power* yang dimiliki musuh

3.2.2.11 Skenario *Use Case* Mengatur Posisi Menyerang

Skenario *use case* mengatur posisi menyerang digunakan pasukan player untuk mendapatkan posisi yang tepat untuk menyerang musuh. Skenario *use case* mengatur posisi menyerang seperti pada tabel 3.12.

Tabel 3.12 Skenario *Use Case* Mengatur Posisi Menyerang

Identifikasi	
Nama *Use Case*	Mengatur Posisi Menyerang
Aktor	Pemain
Tujuan	Pasukan player mendapatkan posisi yang tepat untuk menyerang
Kondisi Awal	Pasukan player pada posisi *default* atau awal
Kondisi Akhir	Pasukan player dapat mengatur posisi menyerang
Skenario	
Aksi Aktor	**Sistem**
a. Memilih pasukan player	
	b. *Setfocus* ke pasukan player
c. Memilih musuh yang diserang pertama	
	d. Pasukan player menyerang musuh yang dipilih pemain
	e. Pasukan player mengidentifikasi *life*, *power* serta *range* musuh yang berada pada jangkauan tembak pasukan
	f. Pasukan player mengatur posisi untuk menyerang

3.2.2.12 Skenario *Use Case* Mengatur Jarak Tembak

Skenario *use case* mengatur jarak tembak digunakan pasukan player untuk mengatur jarak tembak untuk menyerang. Skenario *use case* mengatur jarak tembak seperti pada tabel 3.13.

Tabel 3.13 Skenario *Use Case* Mengatur Jarak Tembak

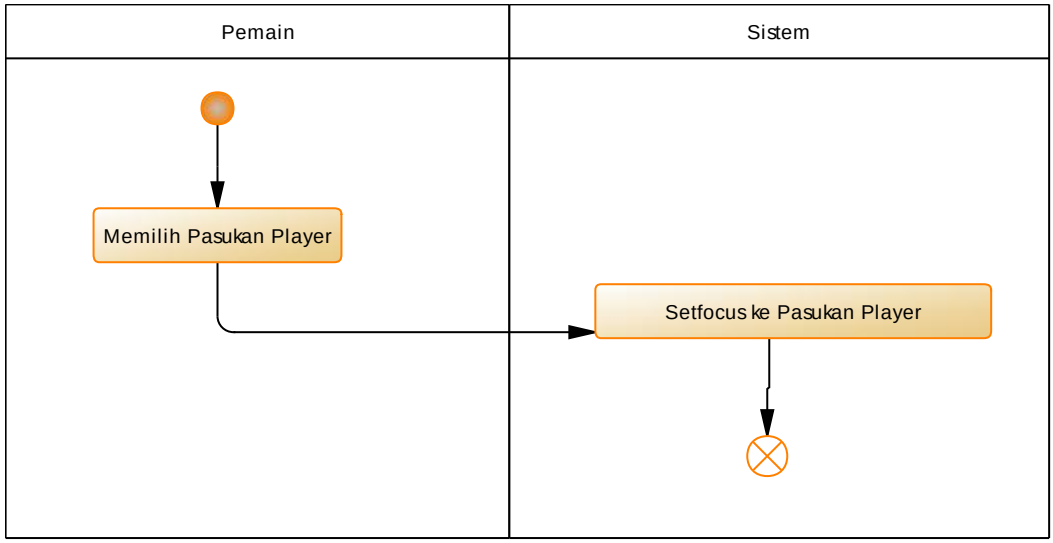
Identifikasi	
Nama *Use Case*	Mengatur Jarak Tembak
Aktor	Pemain
Tujuan	Pasukan player dapat mengatur jarak tembak untuk menyerang
Kondisi Awal	Pasukan player pada posisi *default* atau awal
Kondisi Akhir	Pasukan player mengatur jarak tembak
Skenario	
Aksi Aktor	**Sistem**
a. Memilih pasukan player	
	b. *Setfocus* ke pasukan player
c. Memilih musuh yang diserang pertama	
	d. Pasukan player menyerang musuh yang dipilih pemain
	e. Pasukan player mengidentifikasi *life*, *power* serta *range* musuh yang berada pada jangkauan tembak pasukan
	f. Pasukan player mengatur jarak tembak

3.3 *Activity Diagram*

Activity diagram menggambarkan berbagai alir aktifitas dalam sistem yang dirancang dan bagaimana alir-alir berawal. *Activity diagram* juga dapat menggambarkan proses pararel yang mungkin terjadi pada beberapa eksekusi. *Activity diagram* merupakan *state* program khusus, dimana sebagian *state* adalah *action*.

3.3.1 *Activity Diagram* Memilih Pasukan Player

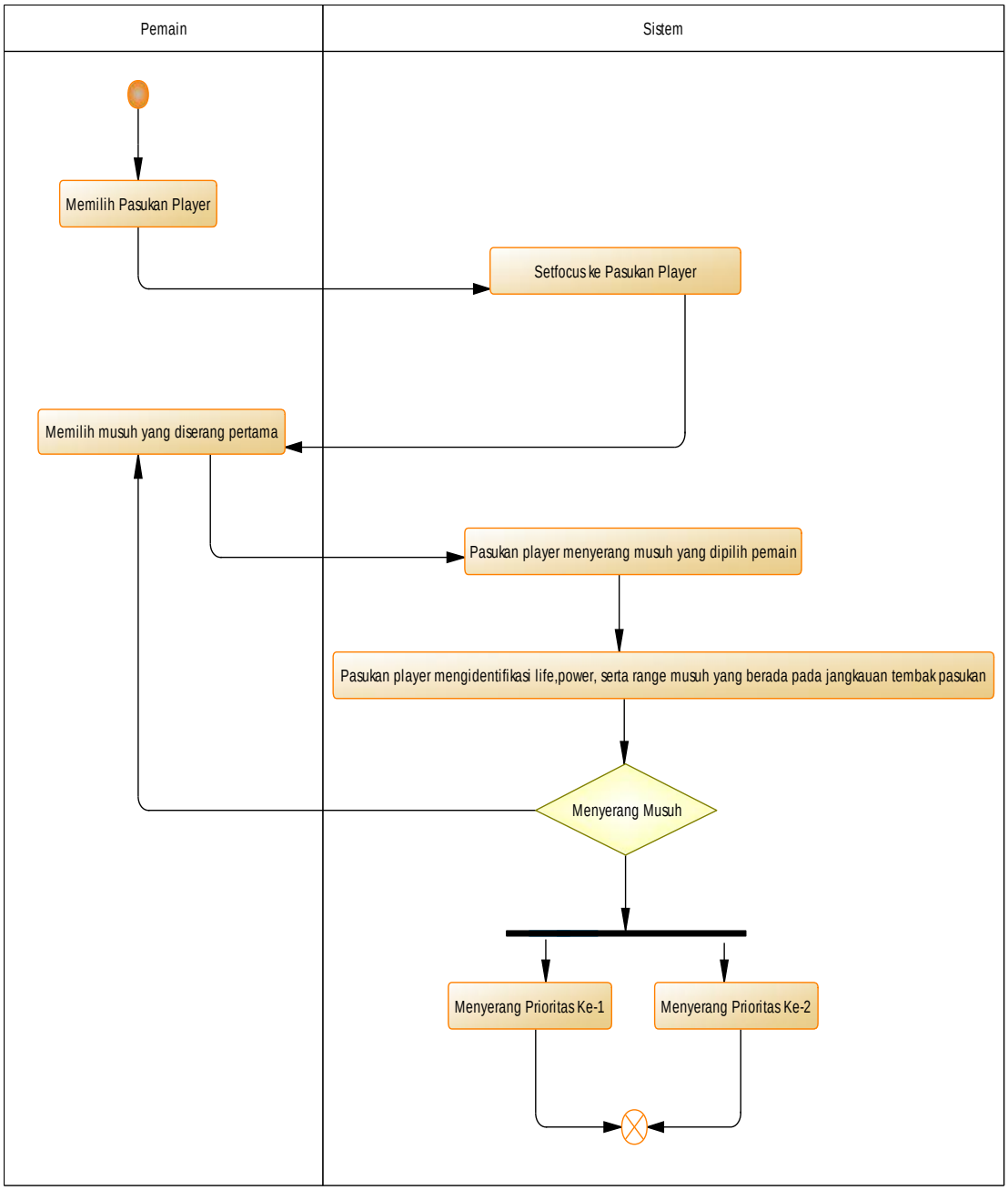
Activity diagram memilih pasukan player, dimana pemain memilih pasukan player. Berikut adalah *activity diagram* pasukan pada gambar 3.7.



Gambar 3.7 *Activity Diagram* Memilih Pasukan

3.3.2 *Activity Diagram* Menyerang Musuh

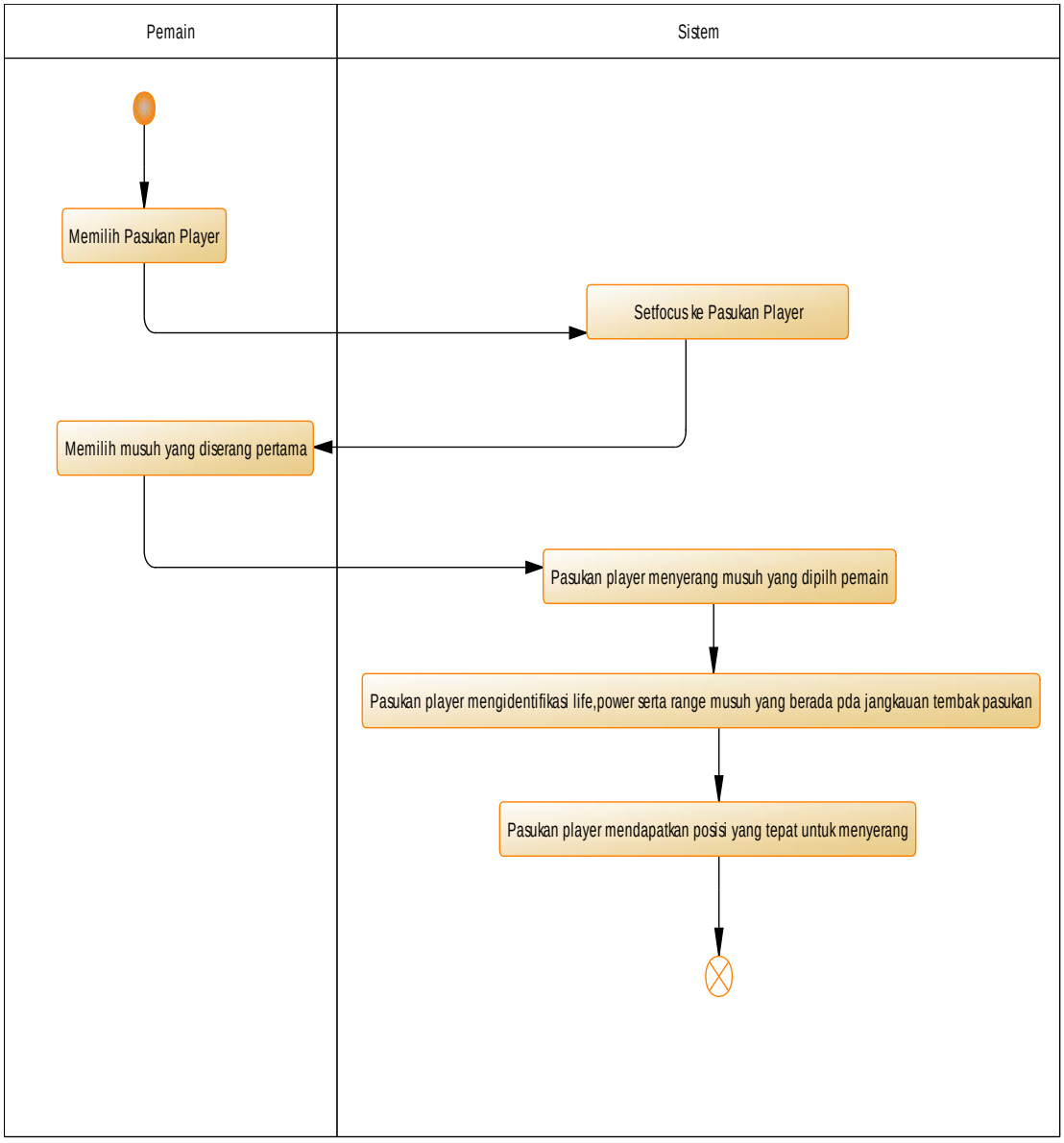
Activity diagram menyerang musuh, pemain memilih pasukan player untuk menyerang musuh pertama yang dipilih pemain untuk serangan selanjutnya pasukan mengidentifikasi *life*, *power*, *range* dan jangkauan tembak kemudian menyerang musuh yang menjadi target serangan pasukan *player*. Berikut adalah *activity diagram* untuk menyerang pada gambar 3.8.



Gambar 3.8 *Activity Diagram* Menyerang Musuh

3.3.3 *Activity Diagram Route Finding*

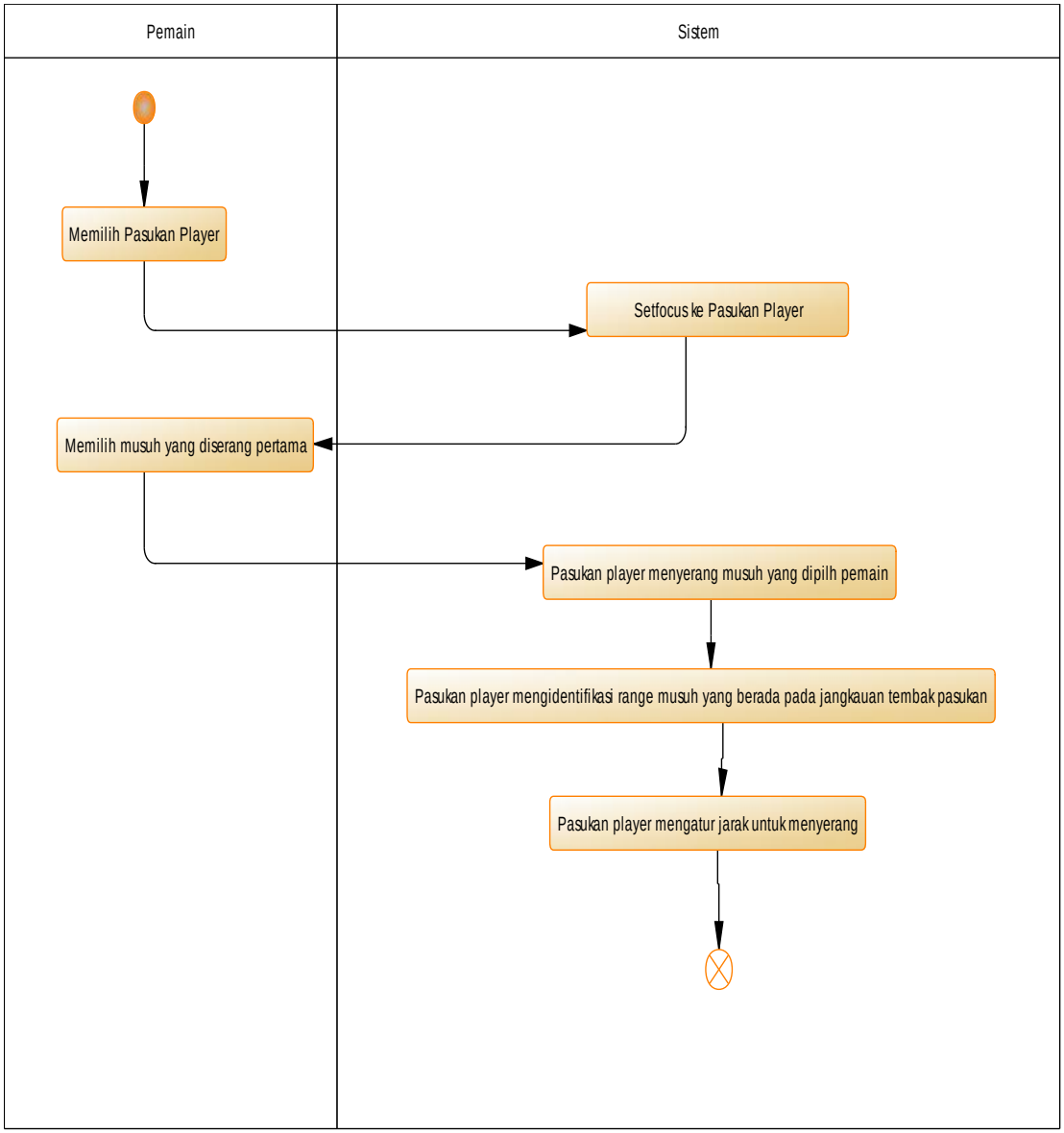
Activity diagram route finding dimana digunakan pasukan player untuk mendapat posisi yang tepat untuk menyerang. Berikut adalah gambar *activity diagram route finding* pada gambar 3.9.



Gambar 3.9 *Activity Diagram Route Finding*

3.3.4 *Activity Diagram* Mengatur *Range* (Jarak)

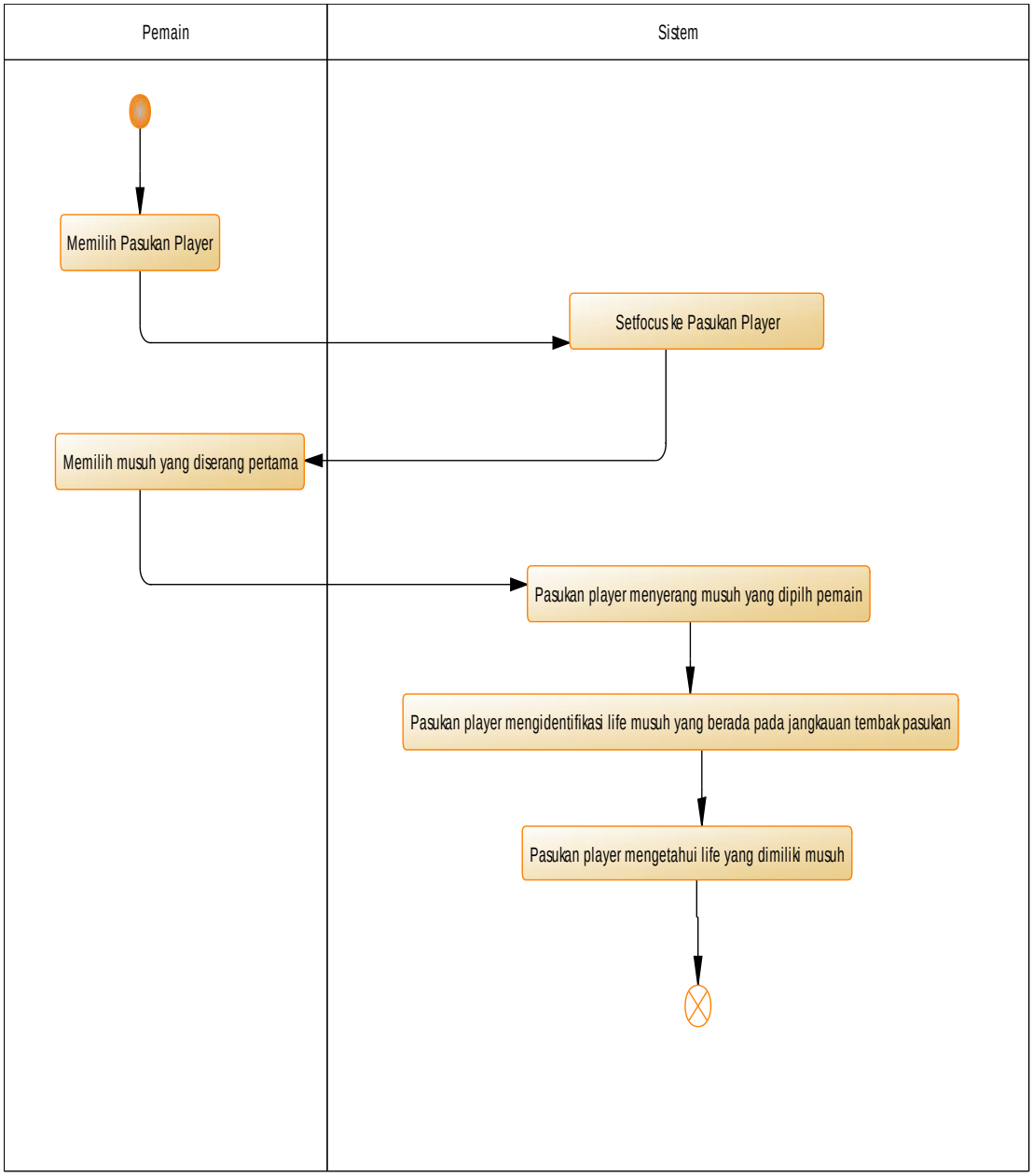
Activity diagram mengatur *Range* (Jarak) digunakan pasukan player untuk mengetahui jarak musuh dan jangkauan menembak dari pasukan player. Berikut adalah gambar *activity diagram* Mengatur *Range* (Jarak) pada gambar 3.10.



Gambar 3.10 *Acivity Diagram* Mengatur *Range* (Jarak)

3.3.5 *Activity Diagram* Mengidentifikasi *Life* (Nyawa) Musuh

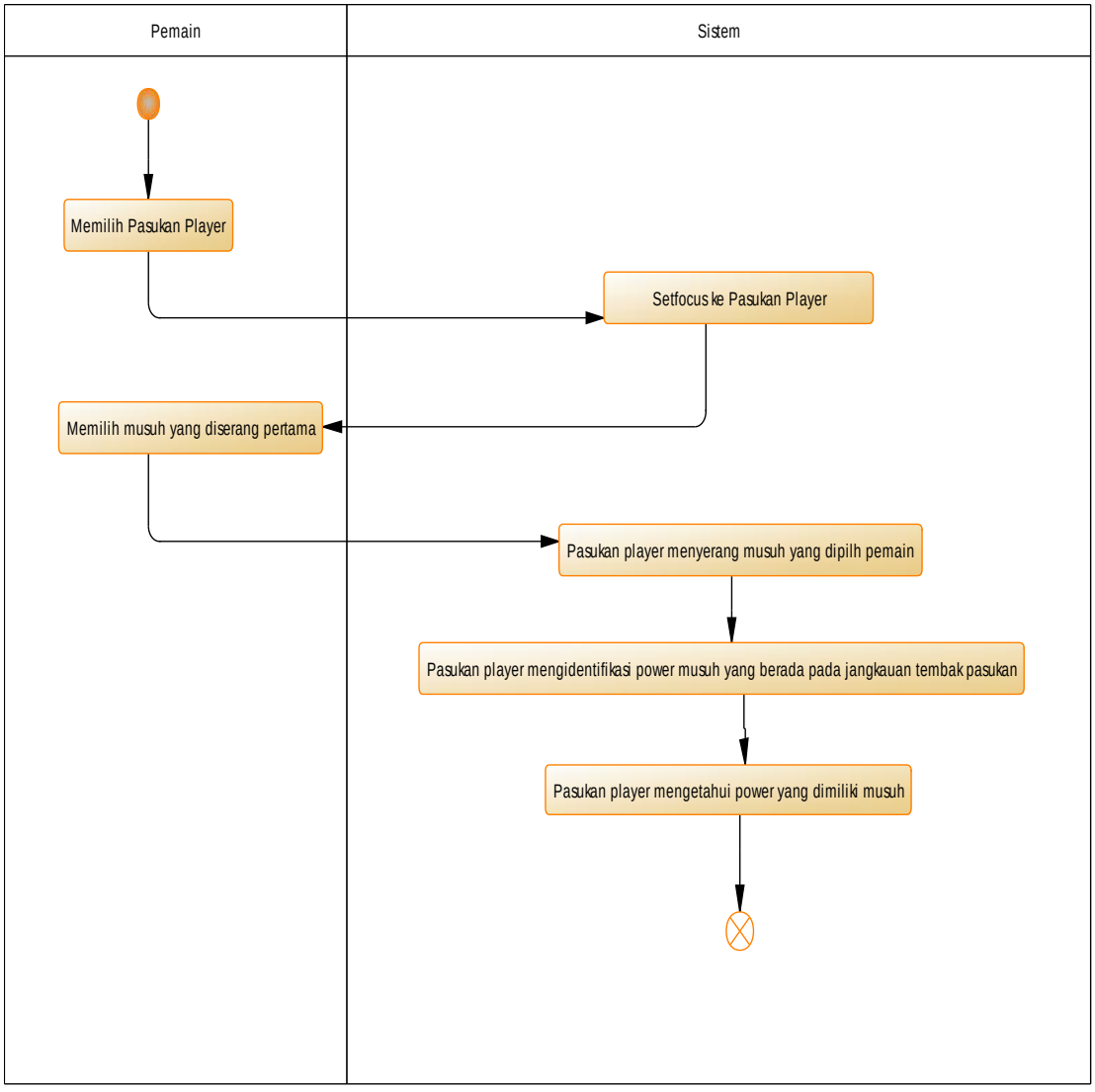
Activity diagram mengidentifikasi *Life* (Nyawa) musuh merupakan parameter yang dimiliki musuh yang akan diidentifikasi oleh pasukan player. Berikut adalah gambar *activity diagram* mengidentifikasi *Life* (Nyawa) musuh pada gambar 3.11.



Gambar 3.11 *Activity Diagram* Mengidentifikasi *Life* (Nyawa) Musuh

3.3.6 *Activity Diagram* Mengidentifikasi *Power* (Kekuatan) Musuh

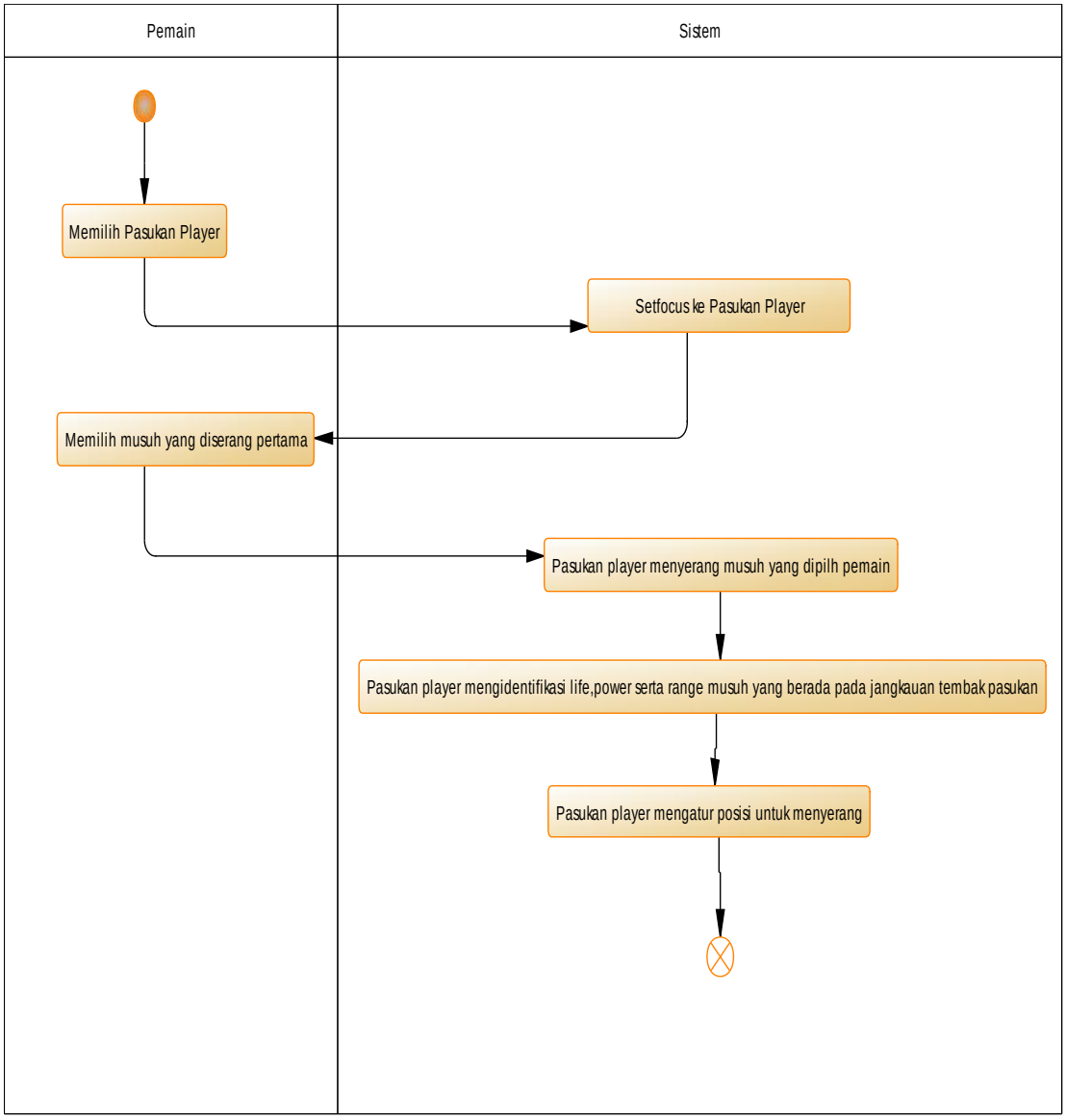
Activity diagram mengidentifikasi *Power* (Kekuatan) musuh merupakan parameter yang dimiliki musuh yang akan diidentifikasi oleh pasukan player. Berikut adalah gambar *activity diagram* mengidentifikasi *Power* (Kekuatan) musuh pada gambar 3.12.



Gambar 3.12 *Activity Diagram* Mengidentifikasi *Power* (Kekuatan) Musuh

3.3.7 *Activity Diagram* Mengatur Posisi Menyerang

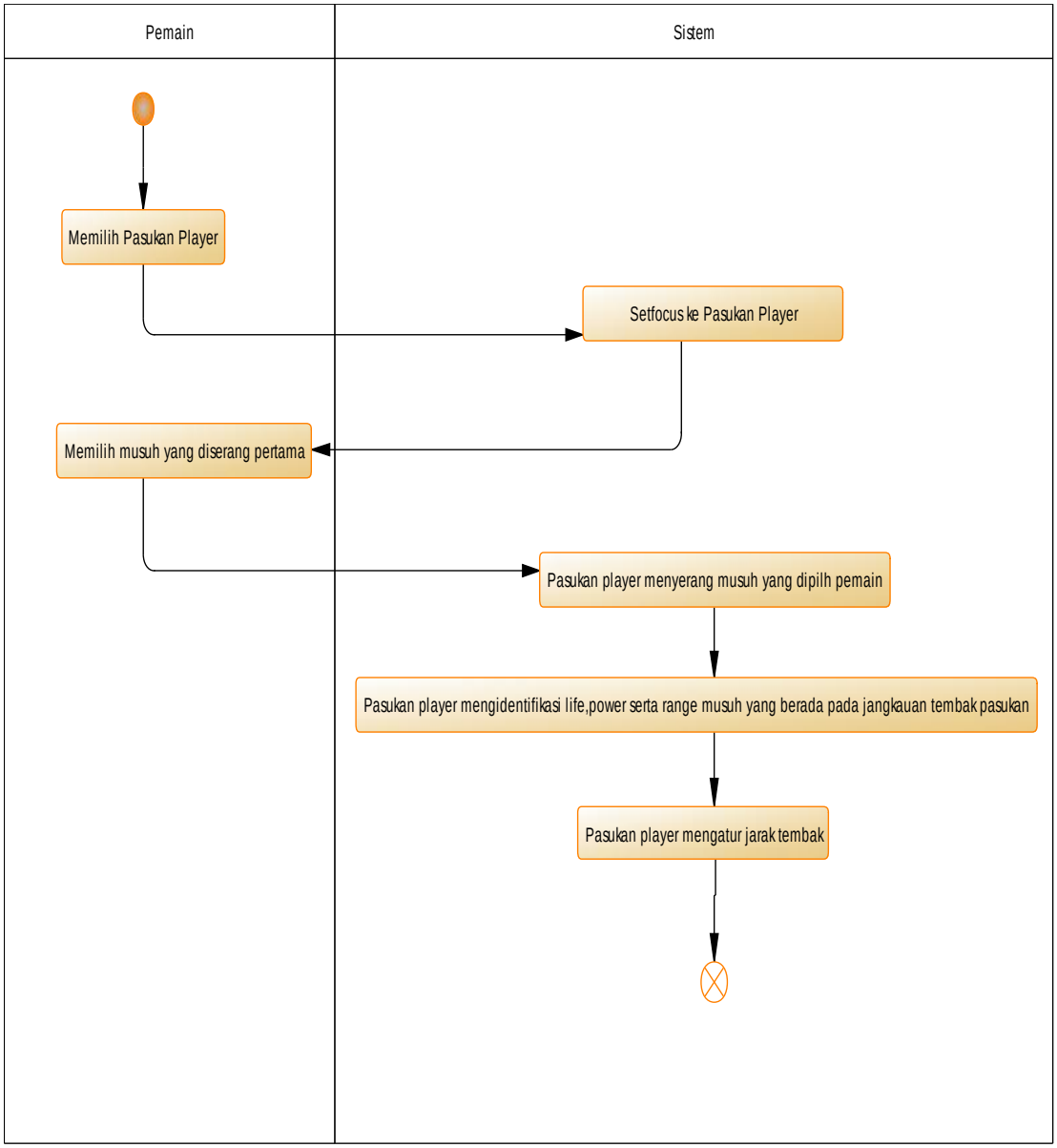
Activity diagram mengatur posisi menyerang digunakan pasukan player untuk mendapat posisi yang tepat untuk menyerang. Berikut adalah gambar mengatur posisi menyerang pada gambar 3.13.



Gambar 3.13 *Activity Diagram* Mengatur Posisi Menyerang

3.3.8 *Activity Diagram* Mengatur Jarak Tembak

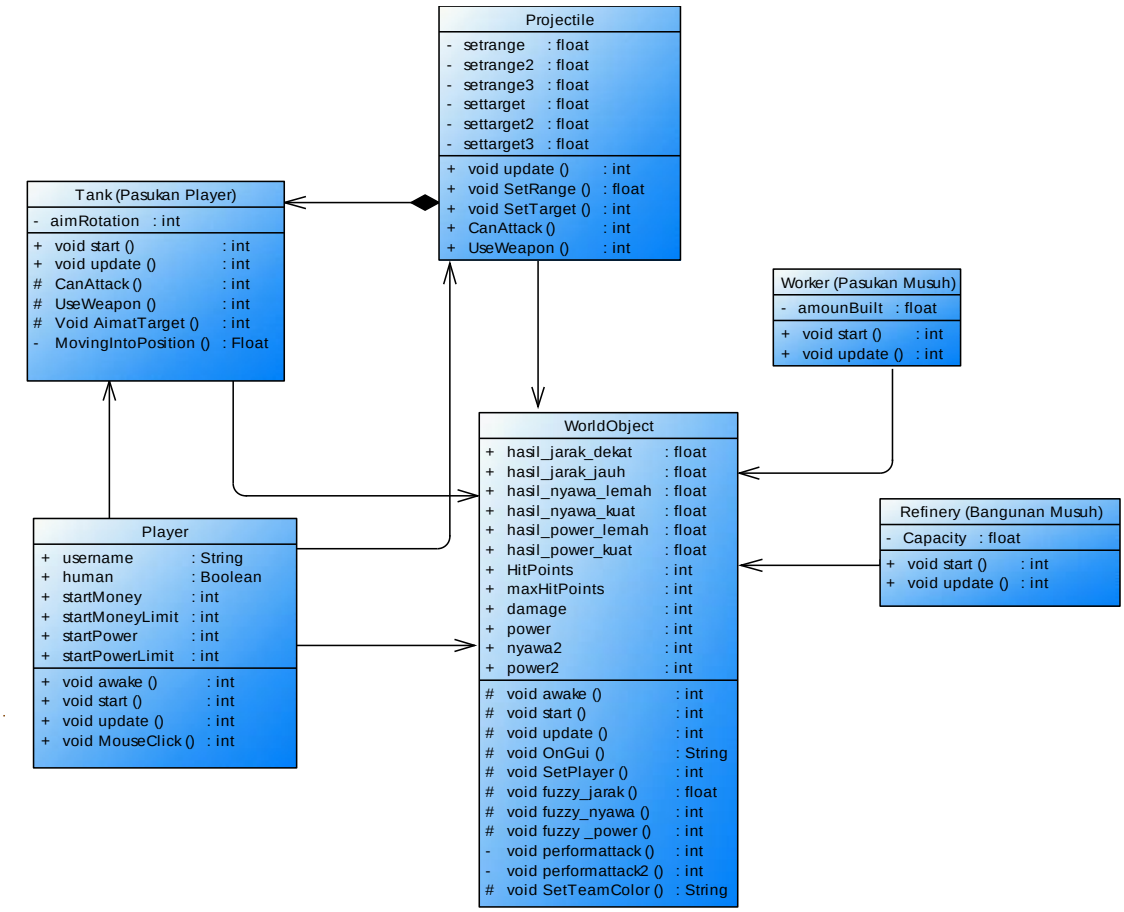
Activity diagram mengatur jarak tembak digunakan pasukan player untuk mengatur jarak tembak untuk menyerang. Berikut adalah gambar *activity diagram* mengatur jarak tembak pada gambar 3.14.



Gambar 3.14 *Activity Diagram* Mengatur Jarak Tembak

3.4 Class Diagram

Class diagram merupakan diagram menggambarkan struktur sistem dari aspek pendefinisian kelas-kelas yang dibuat pada sistem. *Class diagram* pada aplikasi yang dibuat dapat dilihat pada gambar 3.15.



Gambar 3.15 *Class Diagram*

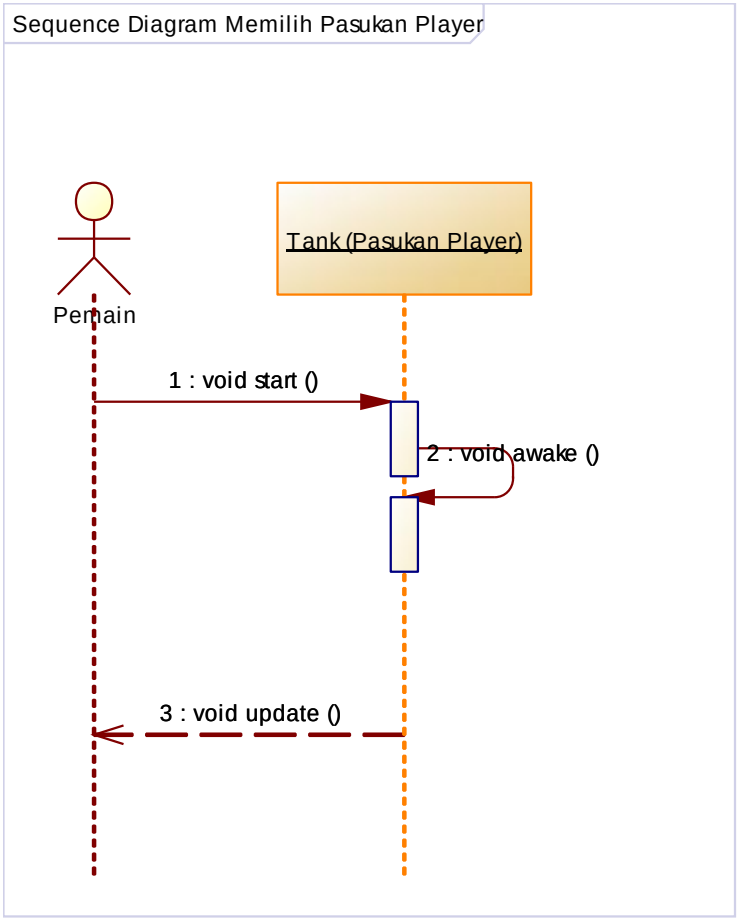
3.5 *Sequence Diagram*

Sequence Diagram menggambarkan interaksi antar objek baik yang ada di dalam sistem maupun yang berada di sekitar sistem, berupa pesan atau *message* yang digambarkan terhadap waktu. *Sequence Diagram* terdiri dari dimensi vertikal (waktu) dan dimensi horisontal (objek-objek terkait).

Sequence Diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respon dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali aktifitas, proses dan perubahan apa saja yang terjadi dan *output* yang dihasilkan.

3.5.1 *Sequence Diagram* Memilih Pasukan Player

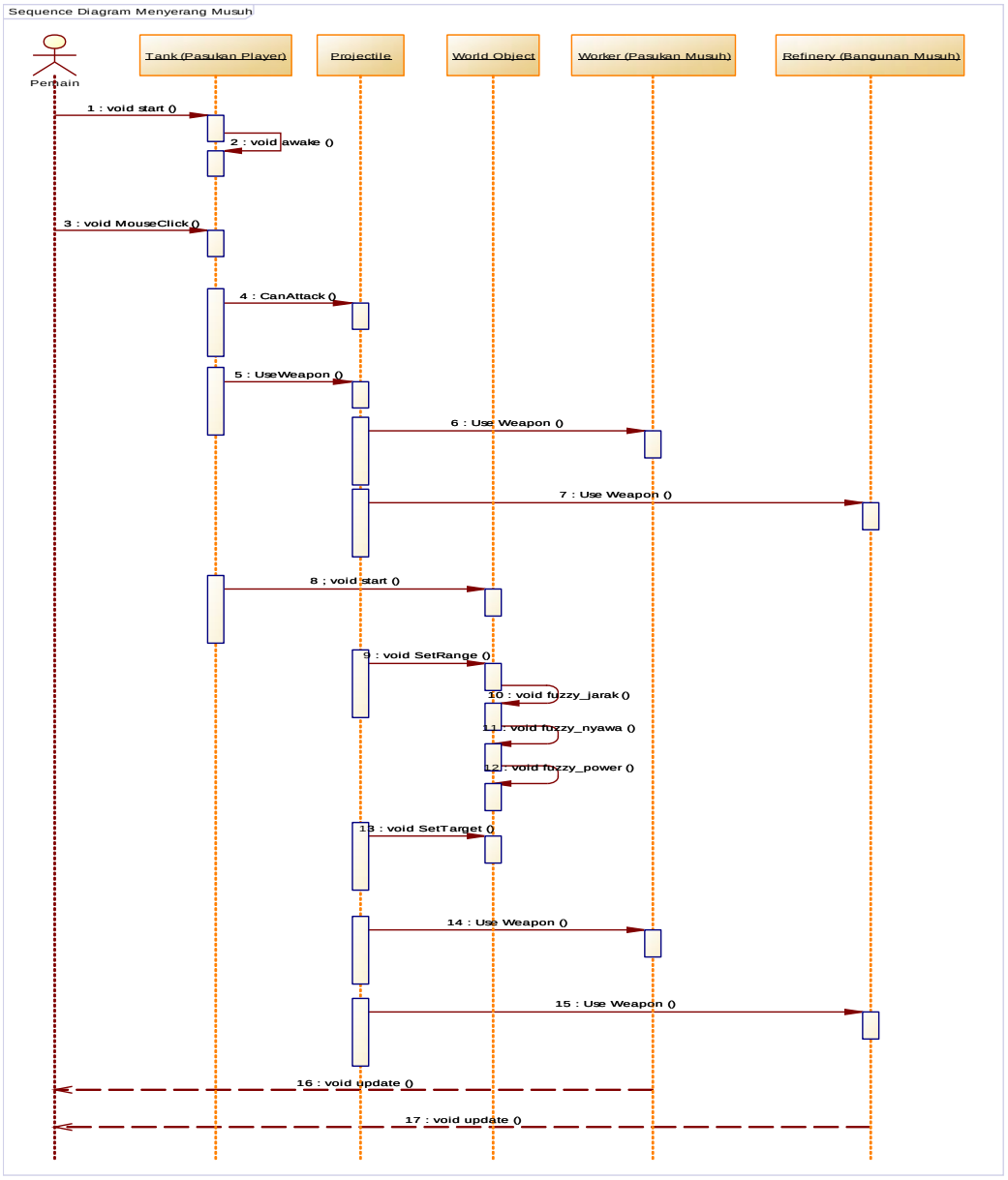
Sequence diagram pasukan dimana pemain memerintahkan pasukan player. Berikut dalah gambar *sequence diagram* memilih pasukan pada gambar 3.16.



Gambar 3.16 *Sequence Diagram* Memilih Pasukan Player

3.5.2 *Sequence Diagram* Menyerang Musuh

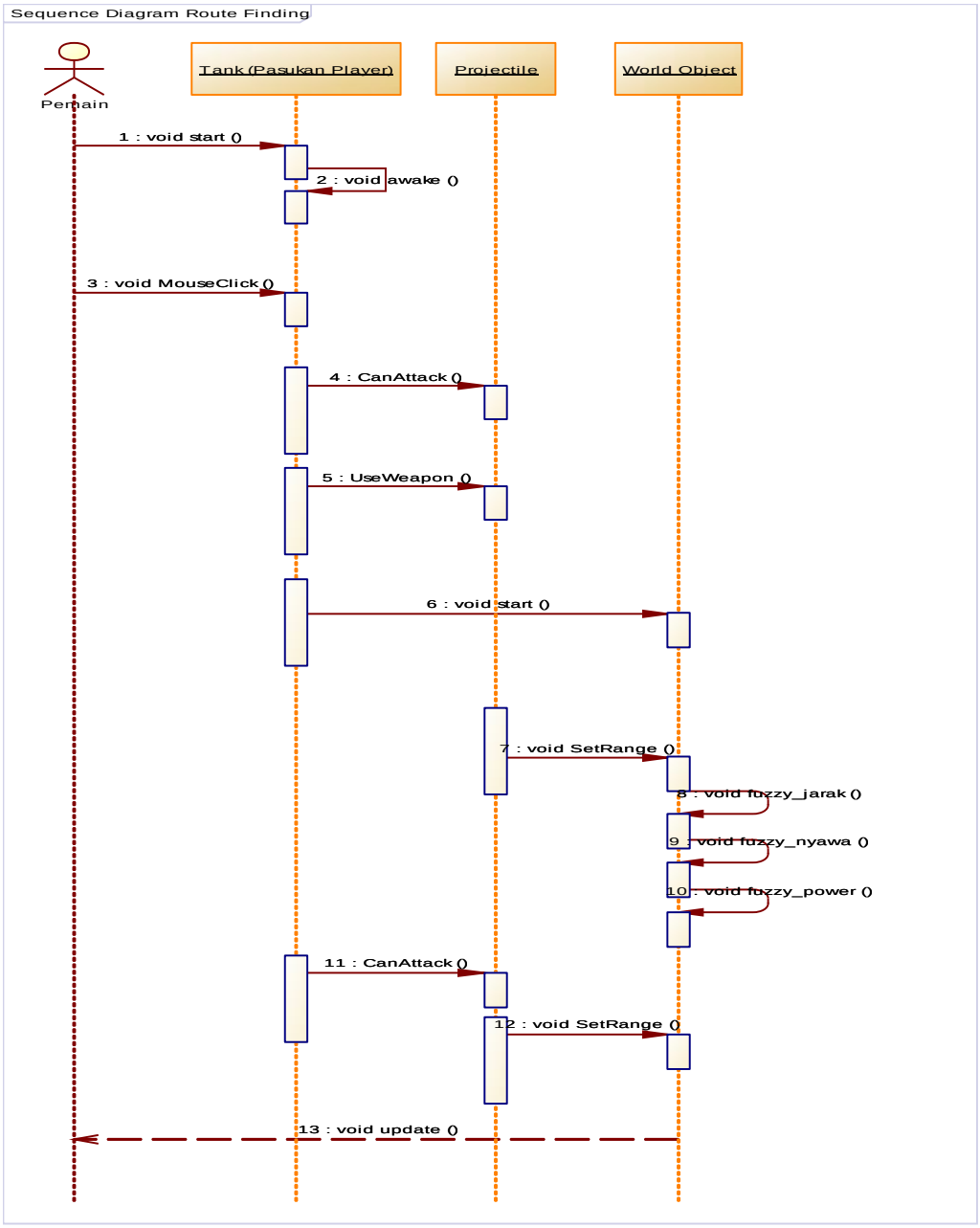
Sequence diagram menyerang digunakan untuk mengetahui langkah-langkah menyerang dari unit pasukan. Berikut adalah gambar *sequence diagram* menyerang pada gambar 3.17.



Gambar 3.17 *Sequence Diagram* Menyerang Musuh

3.5.3 *Sequence Diagram Route Finding*

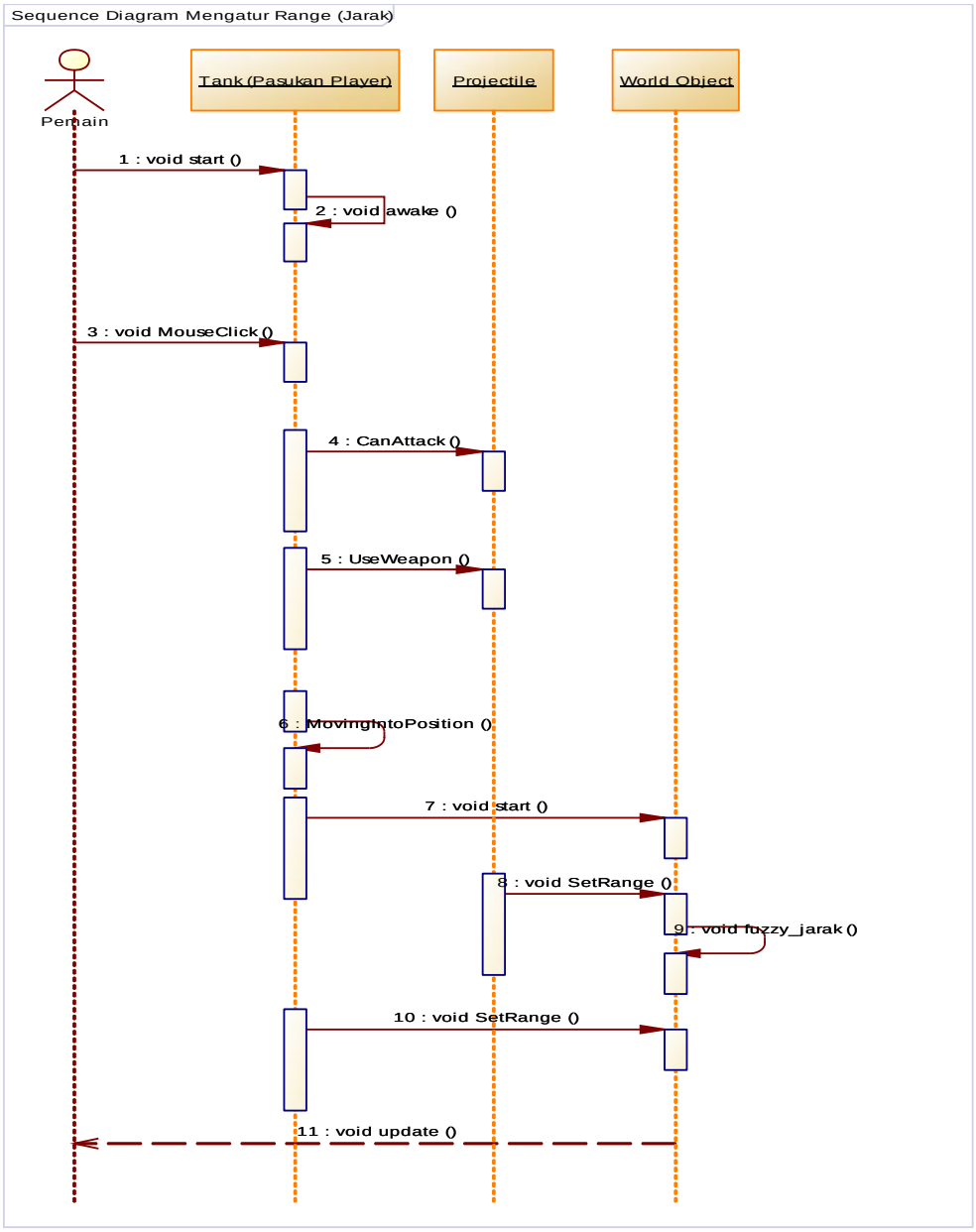
Sequence diagram route finding digunakan pasukan player untuk mendapat posisi yang tepat untuk menyerang. Berikut adalah gambar *sequence diagram route finding* pada gambar 3.18.



Gambar 3.18 *Sequence Diagram Route Finding*

3.5.4 *Sequence Diagram* Mengatur *Range* (Jarak)

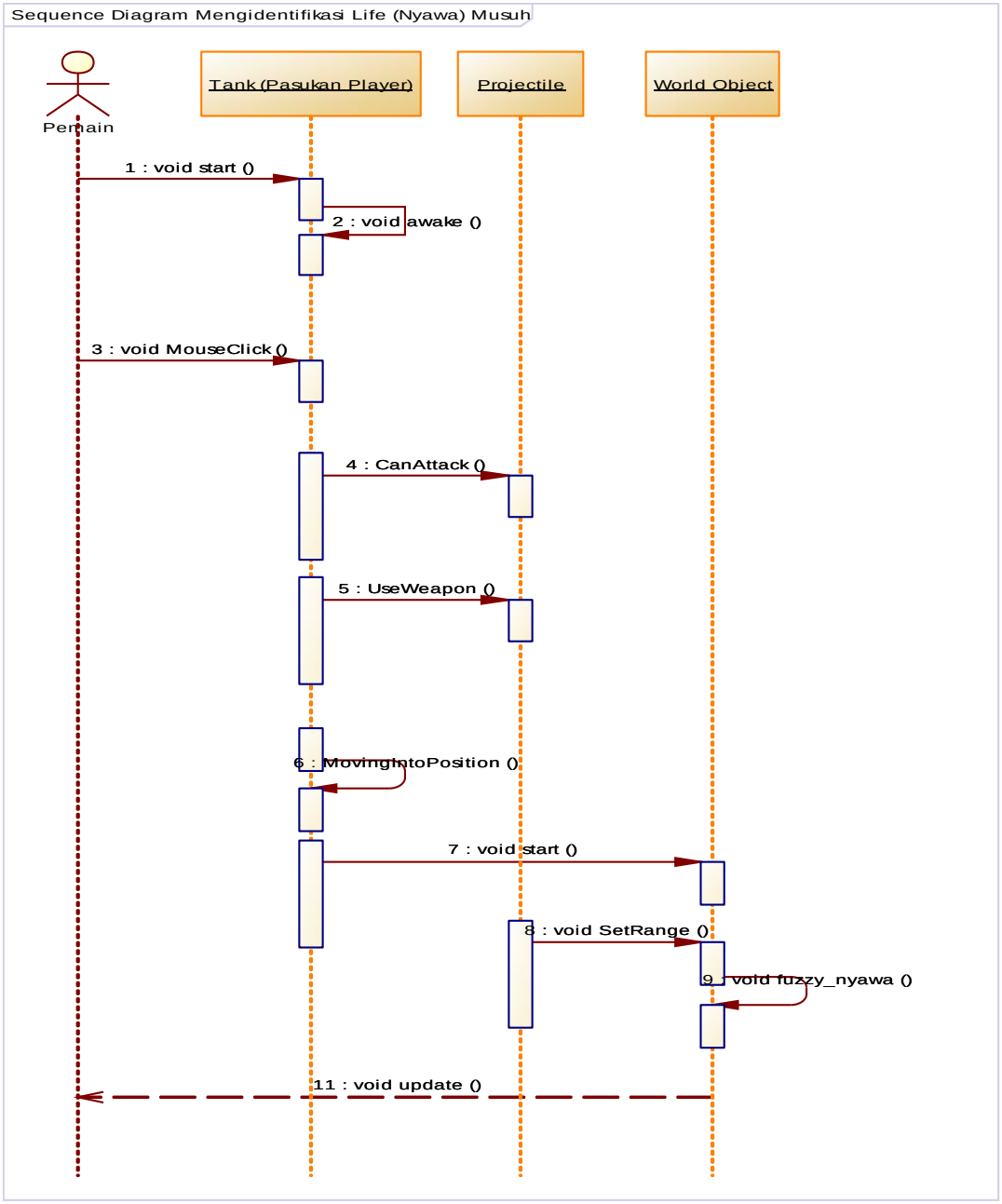
Sequence diagram mengatur *Range* (Jarak) digunakan pasukan player untuk mengatur jarak menyerang musuh. Berikut adalah gambar *sequence diagram* mengatur *Range* (Jarak) pada gambar 3.19.



Gambar 3.19 *Sequence Diagram* Mengatur *Range* (Jarak)

3.5.5 *Sequence Diagram* Mengidentifikasi *Life* (Nyawa) Musuh

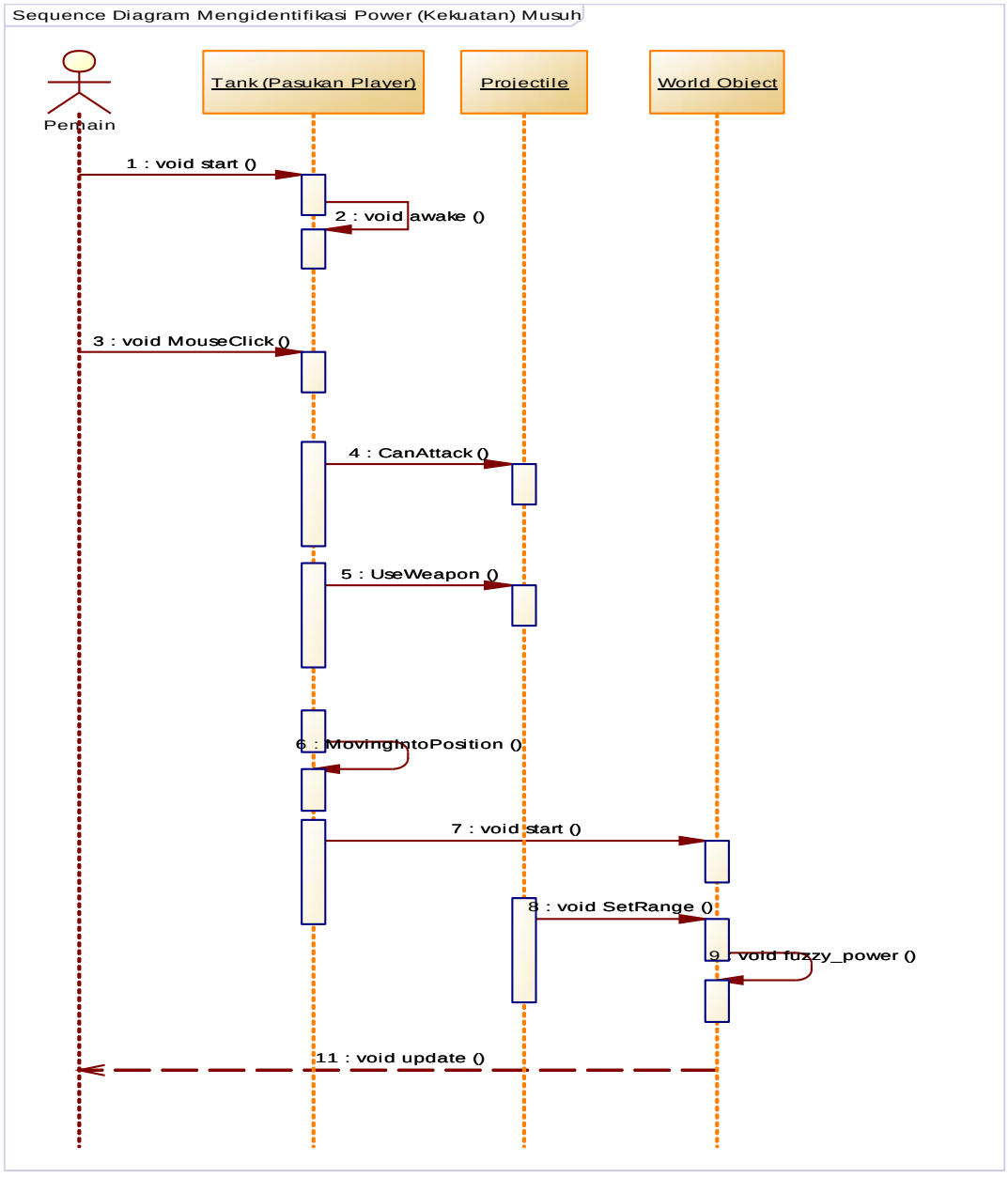
Sequence diagram mengidentifikasi *Life* (Nyawa) musuh digunakan pasukan player untuk mengetahui *life* yang dimiliki musuh. Berikut adalah gambar *sequence diagram* mengidentifikasi *Life* (Nyawa) musuh pada gambar 3.20.



Gambar 3.20 *Sequence Diagram* Mengidentifikasi *Life* (Nyawa) Musuh

3.5.6 *Sequence Diagram* Mengidentifikasi *Power* (Kekuatan) Musuh

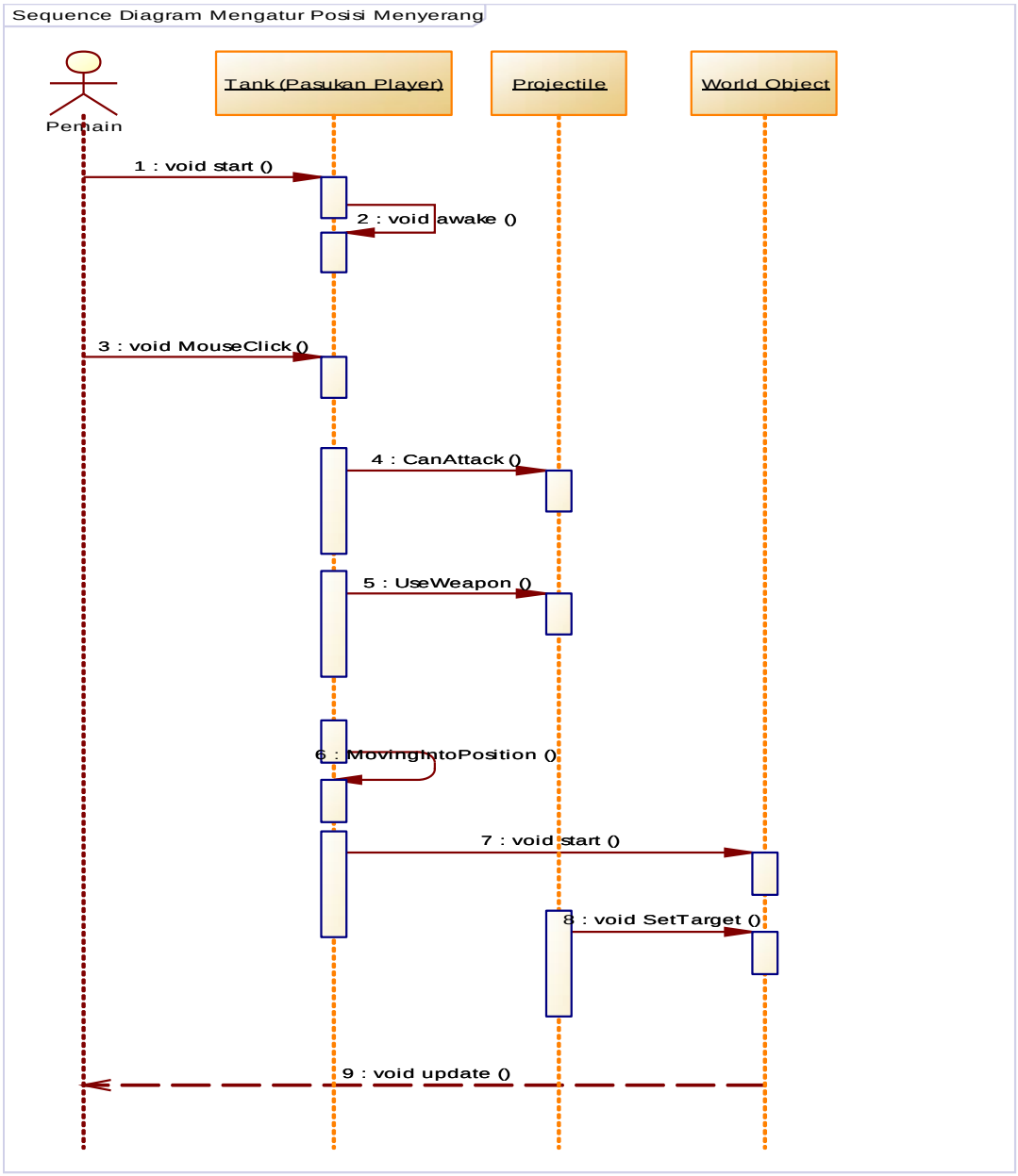
Sequence diagram Power (Kekuatan) digunakan pasukan player untuk mengetahui *power* yang dimiliki musuh. Berikut adalah gambar *sequence diagram* mengidentifikasi *Power* (Kekuatan) musuh pada gambar 3.21.



Gambar 3.21 *Sequence Diagram* Mengidentifikasi *Power* (Kekuatan) Musuh

3.5.7 *Sequuence Diagram* Mengatur Posisi Menyerang

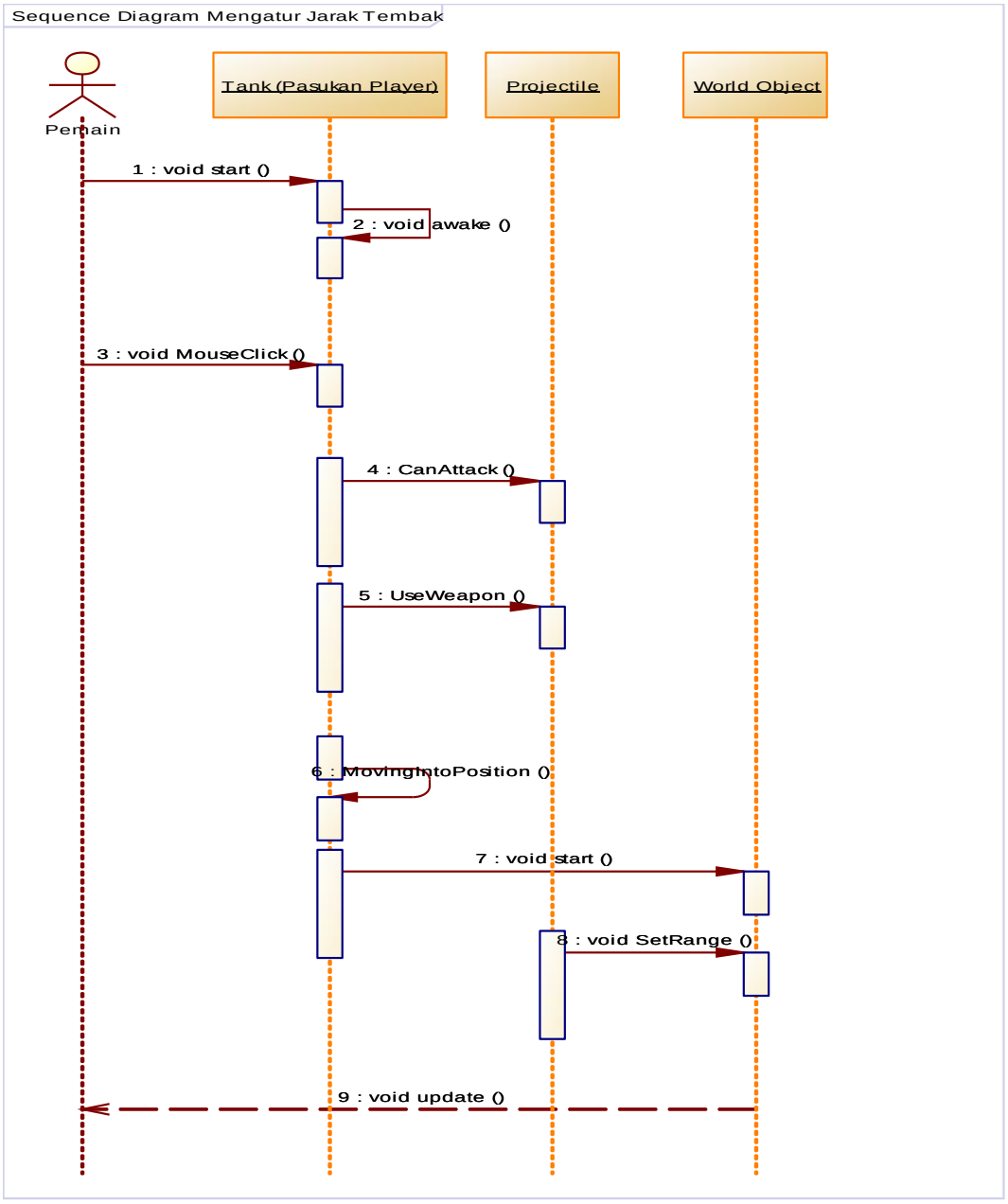
Sequence diagram mengatur posisi menyerang digunakan pasukan player untuk mendapatkan posisi yang tepat untuk menyerang. Berikut adalah gambar mengatur posisi menyerang pada gambar 3.22.



Gambar 3.22 *Sequence Diagram* Mengatur Posisi Menyerang

3.5.8 *Sequuence Diagram* Mengatur Jarak Tembak

Sequence diagram mengatur jarak tembak digunakan pasukan player untuk mengatur jarak tembak untuk menyerang. Berikut adalah gambar mengatur posisi menyerang pada gambar 3.23.



Gambar 3.23 *Sequence Diagram* Mengatur Jarak Tembak

3.6 Perancangan

Perancangan adalah suatu bagian metodologi pengembangam dari suatu perangkat lunak yang dilakukan setelah tahapan untuk memberikan gambaran secara terperinci. Dalam tahap ini digambarkan rancangan perangkat lunak yang akan dibangun sebelum dilanjutkan ke proses pengkodean (*coding*) ke dalam bahasa pemrograman.

3.6.1 Perancangan Perangkat Lunak

Perancangan perangkat lunak adalah kegiatan untuk merancang aplikasi yang dibangun, tahapannya dimulai dari perancangan antar muka, jaringan semantik dan perancangan *method*.

3.6.1.1 Perancangan Antar Muka

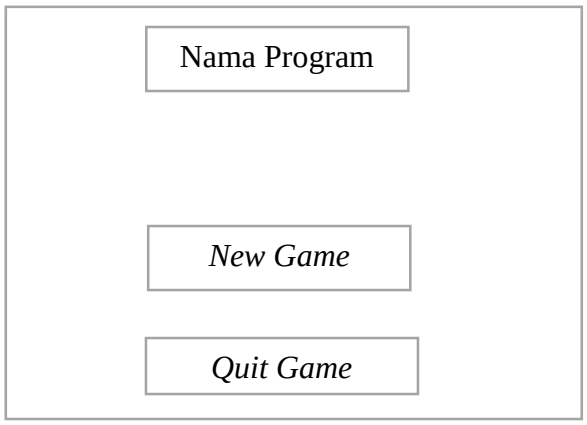
Perancangan antar muka dilakukan untuk mempermudah dalam megimplementasikan sistem yang dibangun. Berikut adalah perancangan antar muka dari aplikasi yang dibangun.

1. Tampilan Menu Utama

T01	
	- Klik *Select* menuju ke T02 - Klik *Pause* untuk keluar

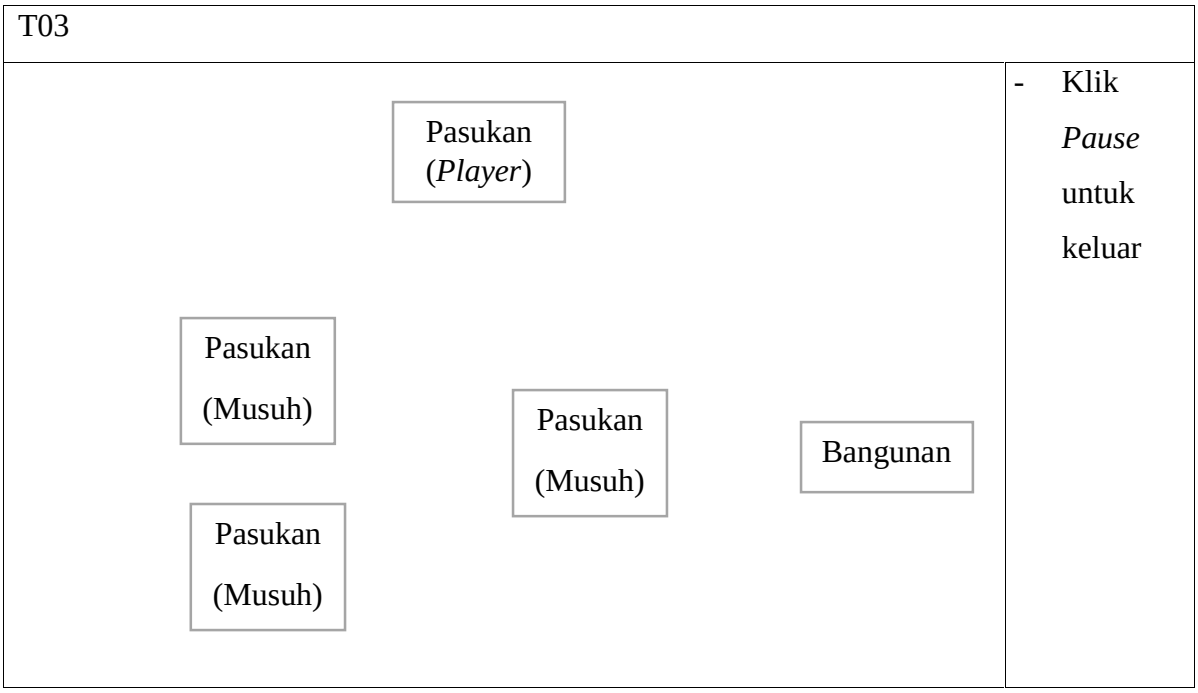
Gambar 3.24 Antar Muka Tampilan Menu Utama

2. Antar muka untuk mulai permainan baru

T02	
	- Klik *New Game* menuju ke T03 - Klik *Quit Game* untuk keluar - Klik *Pause* untuk keluar

Gambar 3.25 Antar Muka Untuk Mulai Permainan Baru

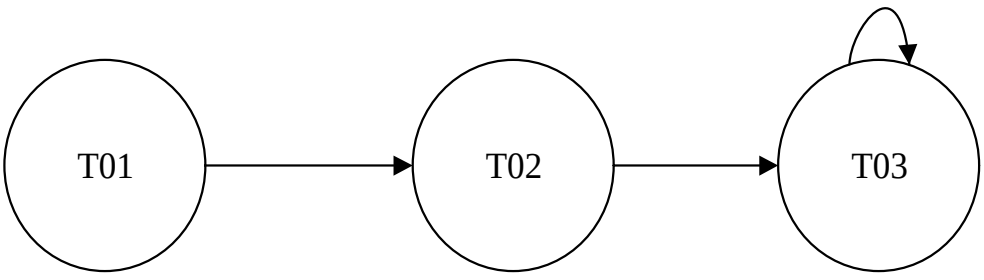
3. Antar muka posisi awal



Gambar 3.26 Antar Muka Posisi Awal

3.6.1.2 Jaringan Semantik

Struktur aplikasi dibuat secara moduler, dimana program dipecah menjadi modul-modul kecil yang mudah dibuat, mudah dites dan mudah untuk dimodifikasi. Dalam pembuatan *game* ini menggunakan alat bantu bagan struktural yang disebut jaringan semantik. Berikut adalah jaringan semantik dari *game* yang dibuat pada gambar 3.27.



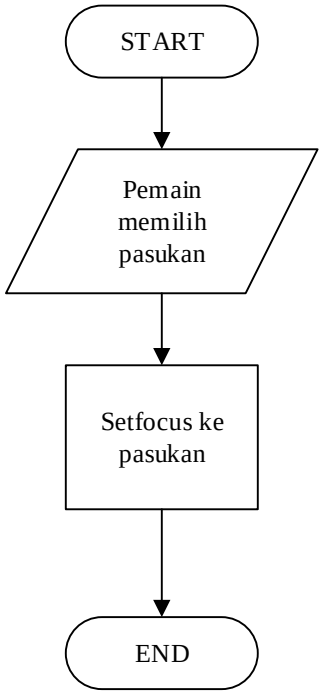
Gambar 3.27 Jaringan Semantik

3.6.2 Perancangan *Method*

Perancangan metode merupakan perancangan yang dibuat setelah perancangan antar muka dan jaringan semantik. Perancangan metode berfungsi mendeskripsikan *method-method* yang berada di dalam aplikasi. Adapun *method-method* yang terdapat pada *game* yang dibuat sebagai berikut.

a. *Method* void awake ()

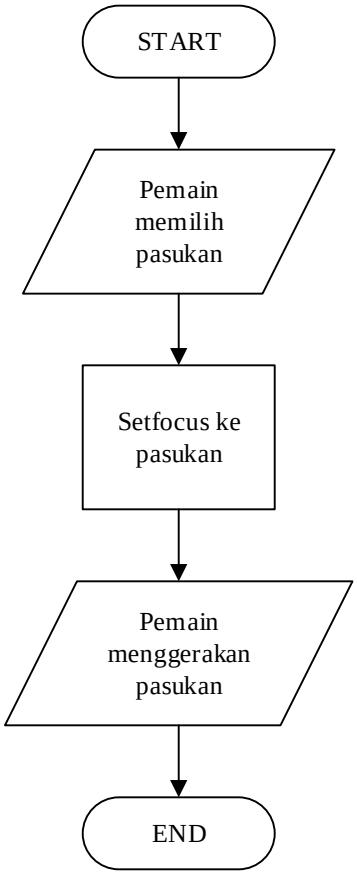
Method void awake () digunakan pemain untuk memilih pasukan kemudian *setfocus* ke pasukan.



Gambar 3.28 *Flowchart Method* void awake ()

b. *Method* void start ()

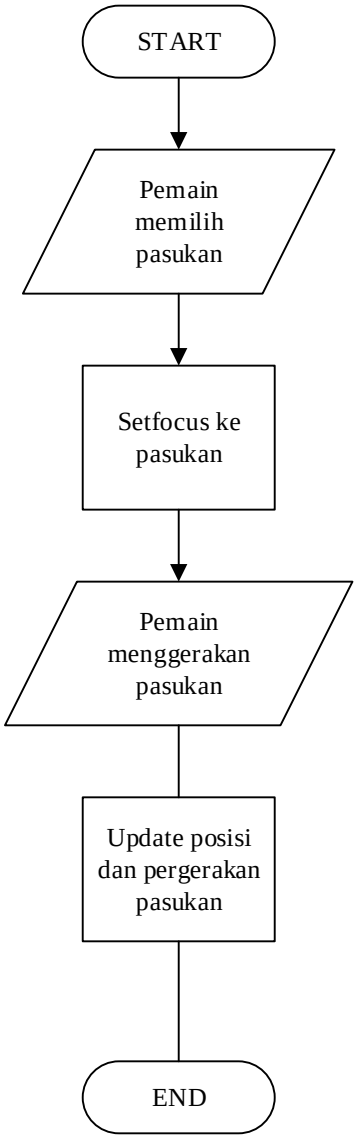
Method void start () digunakan sebagai *method* untuk pergerakan pasukan.



Gambar 3.29 *Flowchart Method* void start ()

c. *Method* void update ()

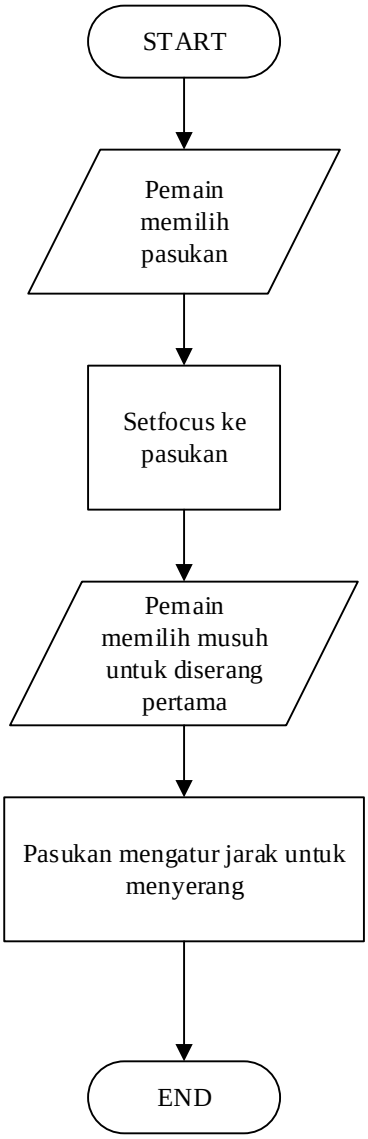
Method void update () digunakan untuk *update* keadaan setelah pasukan melakukan perintah pemain untuk bergerak atau telah melakukan serangan.



Gambar 3.30 *Flowchart Method* void update ()

d. *Method* void fuzzy_jarak ()

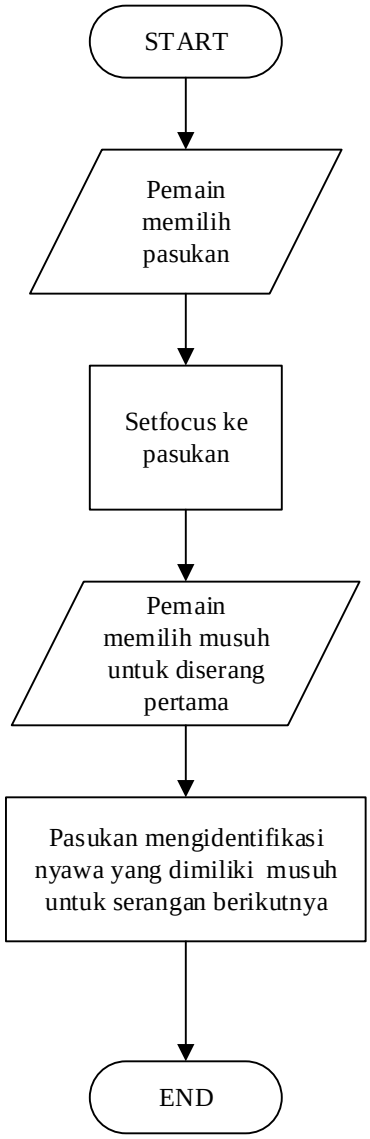
Method void fuzzy_jarak () digunakan pasukan untuk mengatur jarak untuk menyerang musuh.



Gambar 3.31 *Flowchart Method* void fuzzy_jarak

e. *Method* void fuzzy_nyawa ()

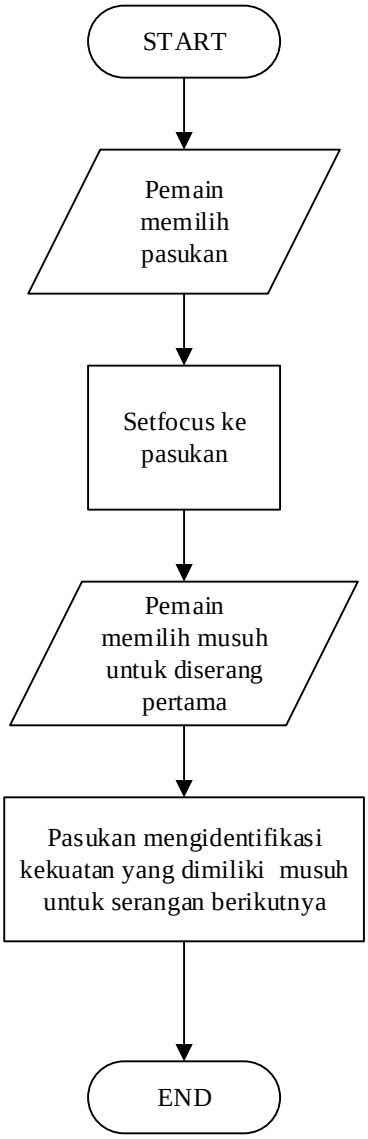
Method void fuzzy_nyawa digunakan pasukan untuk mengetahui nyawa yang dimiliki musuh.



Gambar 3.32 *Flowchart Method* void fuzzy_nyawa ()

f. *Method* void fuzzy_power ()

Method void fuzzy_nyawa digunakan pasukan untuk mengetahui kekuatan yang dimiliki musuh.



Gambar 3.33 *Flowchart Method* void fuzzy_power ()