BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1. Analisis Masalah

Berdasarkan latar belakang masalah diatas, maka dapat dirumuskan masalah adalah bagaimana menyelesaikan masalah tentang perilaku tower dalam mengambil keputusan serta pencarian jalur terbaik bagi *enemy* dalam mencapai base *player*. Selanjutnya , perumusan masalah dari permasalahan di atas adalah sebagai berikut :

1. Bagaimana membuat pergerakan *enemy* lebih dinamis dalam melakukan pemilihan jalur.
2. Bagaimana membuat tower berperilaku dalam mengambil keputusan .

3.2. Analisis *Game*

Game yang akan dibangun berjudul "Benteng Bertahan" yang ber-genre *Strategy,* dimana *player* harus berusaha mempertahankan kota yang menjadi benteng terakhir bagi manusia.

3.2.1. *Storyline*

Tower defense "Benteng Bertahan" menceritakan tentang penyerbuan *enemy* terhadap manusia. Kekuatan *enemy* yang jauh diluar perkiraan manusia membuat semua benteng yang ada di dunia berhasil di hancurkan oleh *enemy* dengan sangat mudah. *Enemy* juga membunuh semua manusia yang hidup sehingga ras manusia terancam punah dan musnah.

Sisa – sisa manusia yang hidup membangun sebuah senjata yang dapat menghentikan langkah *enemy* dalam menghancurkan ras manusia. Senjata tersebut adalah tower dengan kekuatan yang dasyat yang dapat menghancurkan *enemy* dengan mudah dan menjadi harapan terakhir manusia yang bertahan di dalam benteng. Tower – tower tersebut akhirnya di pasang di setiap benteng yang ditinggali oleh manusia untuk mencegah *enemy* menyerang ras manusia .

Di suatu tempat di sebuah kota bernama Rain City , hiduplah seorang pemuda . Pemuda tersebut adalah seorang ahli strategi yang sangat jenius. Dia berhasil mempertahankan banyak benteng tempat tinggalnya umat manusia dari serangan para *enemy* yang mencoba untuk menghancurkan ras manusia. Dengan taktik dan strategi dalam penempatan tower sehingga tidak seekor *enemy* pun bisa memasuki benteng – benteng yang telah dia jaga. Tetapi kedamaian tidak selamanya terjadi , tiba - tiba *enemy* datang !!! kali ini enemy berusaha untuk menghancurkan Benteng Takesi . Penduduk benteng tersebut memanggil sang pemuda jenius untuk menyelamatkan mereka dari serangan *enemy* .

Player adalah seorang ahli strategi yang sangat jenius. Ia harus menghentikan *enemy* yang akan mencoba untuk menghancurkan Benteng Takesi yang menjadi benteng pertahanan umat manusia dengan berbekal kemampuan strategi dan tower – tower yang akan membantu *player* dalam mempertahankan dan menghentikan usaha *enemy* untuk menghancurkan Benteng Takesih !!!

3.2.2. *Gameplay*

Pada aplikasi game yang akan dibangun ini akan menggunakan algoritma genetika yang akan diimplementasikan terhadap *enemy* dalam mencari jalur menuju *base player*. Hal yang pertama kali dilakukan adalah proses inisialisasi pembangkitan populasi secara random , kemudian dilakukan evaluasi kromosom dilanjutkan dengan seleksi kromosom. Pada proses seleksi akan dilakukan dengan metode *roulette – wheel* dalam prosesnya. Kemudian dilanjutkan dengan operasi *crossover* dengan menggunakan metode *Partial –Mapped Crossover (PMX)* lalu kemudian dilakukan proses mutasi dengan metode *swapping mutation.* Pada proses pencarian nilai *fitness* awal yang menjadi nilai *fitness* adalah penjumlahan jarak semua node dalam jalur yang muncul . Jarak disini berbeda dengan jarak yang menjadi nilai input dalam *fuzzy logic* . Apabila pada jalur terdapat tower yang di pasang oleh *player* maka pada tiap tower yang dipasang akan ditambah nilai sebesar 20 pada jarak antar node . Misalkan

pada node B – C mempunyai nilai sebesar 32 , apabila *player* meletakan tower diantara node B – C maka nilai jarak antara node B – C akan menjadi sebesar 52 . Dalam kasus pemasangan tower yang melalui 2 jalur node maka tiap jalur node akan mendapatkan tambahan nilai sebesar 20. Misalkan tower di pasang melalui jalur node B – C dan C – D yang terletak pada persimpangan . Apabila node B – C mempunyai nilai sebesar 32 dan node C – D mempunyai nilai sebesar 7 , maka nilai node B – C yang baru akan menjadi 52 dan node C – D akan menjadi 27 .

Dalam game ini juga digunakan *fuzzy logic* sebagai kecerdasan buatan bagi tower dalam mengambil keputusan dalam menyerang berdasarkan tingkat prioritas *enemy.* Dalam *fuzzy logic* parameter yang akan digunakan adalah *Health Point Enemy* (HPE) , *Damage Point Enemy* (DPE) dan Jarak *Enemy* (JE) . Jarak *Enemy* dalam *fuzzy logic* berbeda dengan jarak yang digunakan dalam algoritma genetika . Pada jarak dalam kasus perhitungan *fuzzy logic* akan digunakan sebagai parameter yang menentukan posisi *enemy* . Apabila *enemy* sudah memasuki jarak tembak tower maka tower akan mulai melakukan perhitungan dengan menggunakan *fuzzy tsukamoto* berdasarkan inputan parameter berupa HPE , DPE dan JE . Nilai – nilai yang dihasilkan dalam perhitungan *fuzzy* kemudian akan masuk dalam proses *fuzzyfikasi* . Setelah proses *fuzzyfikasi* dilakukan nilai terebut akan dibandingkan dalam proses *inferensi fuzzy* yang mana akan di cari nilai MIN sesuai dengan aturan *fuzzy tsukamoto*. Setelah itu dilakukan proses *deffuzifikasi* yaitu perhitungan nilai keluaran dengan rata – rata terbobot yang kemudian akan menghasilkan *output* berupa prioritas serangan terhadap *enemy* .

Di awal permainan, pemain diberikan modal *gold* sebanyak 1000 *gold* . Setiap pemain membunuh *enemy* , pemain akan mendapatkan *gold* untuk membangun tower atau melakukan *upgrade tower* . Masing – masing tower maksimal hanyak bisa di-upgrade sebanyak 5 level. Setiap pemain meng-*upgrade* tower , maka akan menambah level , *damage* , kecepatan tembak , jarak tembak tower dan jumlah *HPE (Health Point Enemy).*

Di dalam permainan, terdapat 6 *wave* yang harus diselesaikan pemain. Setiap *wave* memiliki waktu jeda sebanyak 10 detik sebelum *wave* berikutnya keluar. Jadi selama rentang waktu tersebut, *player* harus membangun tower-tower untuk menghancurkan *enemy* dan juga terdapat beberapa jenis tower yang dapat dibangun oleh *player*. Player dapat membangun 3 jenis tower dengan kemampuan yang berbeda pada setiap tower. Diantaranya adalah Tower Api , Tower Es dan Tower Listrik . Tower api mempunyai *damage point* terbesar diantara tower – tower yang lain . Tower Es dapat memperlambat pergerakan *enemy* dan Tower Listrik mempunya kecepatan serangan yang tinggi.

Pada *stage 1* benteng diserang oleh *enemy* . Tiba – tiba *enemy* datang , mereka mencoba menghancurkan benteng takesi tempat umat manusia tinggal . Pada *stage 1* setting jalan menuju benteng , *player* mempunyai misi untuk menghancurkan *enemy* yang berusaha memasuki base dan memukul mundur *enemy* keluar menjauhi kota. *Player* akan memenangkan permainan apabila berhasil mempertahnkan kota dan memukul mundur *enemy* .

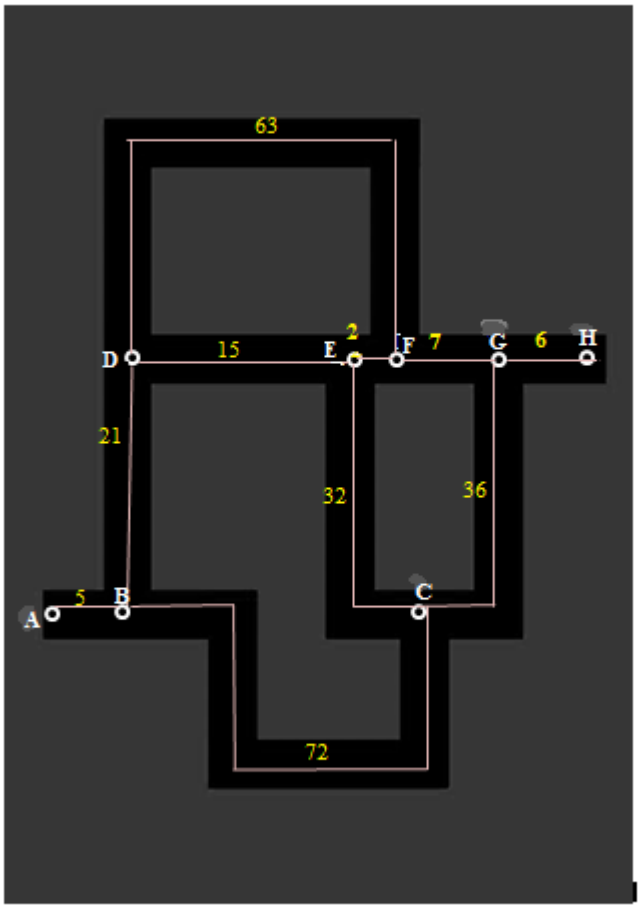
Pada *stage* sebelumnya *enemy* berhasil dipukul mundur oleh *player* menjauhi benteng. Pada *stage 2* ini perjuangan *player* belum selesai . *Player* harus menghancurkan sisa pasukan *enemy* yang masi berusaha memasuki benteng. Permainan akan berakhir apabila *player* berhasil menghancurkan semua sisa *enemy* yang berusaha memasuki kota .

3.3. Analisis Data Masukan

3.3.1. Data Masukan Algoritma Genetika

Pada kasus ini algoritma genetika di gunakan untuk menentukan best route *enemy* dalam menuju *goal point* . Pada gambar 3.1 dapat dilihat setiap node dinyatakan sebagai simpul graf, sedangkan sisi menyatakan jalan yang menghubungkan antara dua buah node . Bobot pada sisi menyatakan jarak antara dua

buah node . Pada setiap node mempunyai nilai yang akan menjadi acuan dalam perhitungan algoritma genetika.



Gambar 3.1 Play area node

3.3.2. Data Masukan *Fuzzy Logic*

Dalam penelitian ini, yang menjadi data masukan ialah faktor-faktor yang mempengaruhi keputusan atau aksi apa yang akan dilakukan oleh tower, Faktor

tersebut berasal dari parameter atau atribut yang dimiliki oleh *enemy*. Faktor-faktor yang menjadi data masukan tersebut antara lain :

a. JE (Jarak *Enemy*) adalah jarak tembak tower terhadap *enemy* . Apabila *enemy* telah memasuki jarak tembak tower maka secara otomatis tower akan bereaksi terhadap *enemy* dan mulai menyerang untuk menghentikan laju *enemy* menuju base *player*. Jarak e*nemy* berkisar antara 1 – 10 yang mana 1 merupakan jarak terdekat *enemy* terhadap tower dan 10 merupakan jarak terjauh yang bisa di serang oleh tower . Jarak dari tower ke *enemy* dihitung berdasarkan perhitungan antara posisi tower dan *enemy* dengan menggunakan rumus :

Jarak = $(X_2 - X_1)^2 + (Y_2 - Y_1)^2$

Dimana :

Jarak = Total Jarak

X_2 = Koordinat x Enemy

X_1 = Koordinat x Tower

Y_2 = Koordinat y Enemy

Y_1 = Koordinat y Tower

Pada table 3.1 akan di jelaskan tentang berapa nilai hitungan fuzzy dalam jarak dengan satuan px .

Tabel 3.1 Nilai jarak dalam perhitungan *fuzzy*

Nilai	Jarak (dalam px)
0	0 px
10	10 px
20	20 px
30	30 px
40	40 px
50	50 px

60	60 px
70	70 px
80	80 px
90	90 px
100	100 px

Apabila nilai dalam hitungan *fuzzy* didapatkan nilai 10 maka pada jarak antara tower dan *enemy* sebesar 10 px .

b. DPE (*Damage Point Enemy)* adalah jumlah point serangan yang diberikan oleh *enemy* terhadap tower. *Damage Point* pada setiap *enemy* dibangkitkan secara random dengan rentang nilai antara 1 – 10 . Jadi pada setiap enemy akan mempunyai *damage* point berkisar antara 1 sampai dengan 10 point . Apabila enemy mempunyai point serangan sebesar 10 point maka tower pemain akan berkurang sebanyak 10 point .

Assumsi *Enemy* dengan DPE sebesar 10 point menyerang Tower Api
Pengurangan HPT (*Health Point Tower)* dapat dihitung dengan rumus :
HPT = HPT sebelum – DPE
HPT untuk Tower Api sebesar 175

Tabel 3.2 Assumsi serangan *Enemy* terhadap Tower Api

NO	Tower Api	
	HP	DPE
1	175	10

2	165	10
3	155	10
4	145	10
5	135	10
6	125	10
7	115	10
8	105	10
9	95	10
10	85	10
11	75	10
12	65	10
13	55	10
14	45	10
15	35	10
16	25	10
17	15	10
18	5	10
19	-5	-

Pada tabel3.2 dapat dilihat proses pengurangan *Health Point (HP)* Tower Api ketika mendapat serangan dari *enemy* dengan damage sebesar 10 point .

Assumsi *Enemy* dengan DPE sebesar 10 point menyerang Tower Es

Pengurangan HPT (*Health Point Tower)* dapat dihitung dengan rumus :

HPT = HPT sebelum – DPE

HPT untuk Tower Es sebesar 150

Tabel 3.3 Assumsi serangan *Enemy* terhadap Tower Es

NO	Tower Es	
	HP	DPE
1	150	10
2	140	10
3	130	10
4	120	10
5	110	10
6	100	10
7	90	10
8	80	10
9	70	10
10	60	10
11	50	10
12	40	10
13	30	10
14	20	10
15	10	10
16	0	-

Pada tabel3.3 dapat dilihat proses pengurangan *Health Point (HP)* Tower Es ketika mendapat serangan dari *enemy* dengan damage sebesar 10 point .

Assumsi *Enemy* dengan DPE sebesar 10 point menyerang Tower Listrik
Pengurangan HPT (*Health Point Tower)* dapat dihitung dengan rumus :
HPT = HPT sebelum – DPE
HPT untuk Tower Listrik sebesar 100

Tabel 3.4 Assumsi serangan *Enemy* terhadap Tower Listrik

NO	Tower Listrik	
	HP	DPE
1	100	10
2	90	10
3	80	10
4	70	10
5	60	10
6	50	10
7	40	10
8	30	10
9	20	10
10	10	10
11	0	-

Pada tabel3.4 dapat dilihat proses pengurangan *Health Point (HP)* Tower Petir ketika mendapat serangan dari *enemy* dengan damage sebesar 10 point .

Keterangan :

DPE = Damage yang di hasilkan *enemy* terhadap tower

HPT = *Health Point* yang dimiliki oleh tower

HPT sebelum = *Health Point* yang dimiliki oleh tower sebelum terkena serangan oleh *enemy*

c. HPE (*Health Point Enemy)* adalah point yang dimiliki *enemy* agar tetap hidup dan bergerak menuju base *player*. *Enemy* akan berhenti bergerak dan kalah apabila HPE mencapai nilai 0. HPE yang dipunyai setiap *enemy* ketika pertama kali muncul adalah 100 dan akan terus berkurang apabila mendapatkan serangan dari tower .

Assumsi Tower Api menyerang *Enemy* .

Pengurangan HPE (*Health Point Enemy*) dapat dihitung dengan rumus :

HPE = HPE sebelum – DPT

Tabel 3.5 Assumsi Tower Api menyerang *Enemy*

NO	*Enemy*	
	HP	DPT
1	100	4
2	96	4
3	92	4
4	88	4

5	84	4
6	80	4
7	76	4
8	72	4
9	68	4
10	64	4
11	60	4
12	56	4
13	52	4
14	48	4
15	44	4
16	40	4
17	36	4
18	32	4
19	28	4
20	24	4
21	20	4
22	16	4
23	12	4
24	8	4
25	4	4
26	0	-

Pada tabel 3.5 dapat dilihat proses pengurangan *Health Point (HP) Enemy* ketika mendapat serangan dari Tower Api dengan damage sebesar 4 point .

Assumsi Tower Es menyerang *Enemy* .
Pengurangan HPE (*Health Point Enemy*) dapat dihitung dengan rumus :
HPE = HPE sebelum – DPT

Tabel 3.6 Assumsi Tower Es menyerang *Enemy*

NO	*Enemy*	
	HP	DPT
1	100	2
2	98	2
3	96	2
4	94	2
5	92	2
6	90	2
7	88	2
8	86	2
9	84	2
10	82	2
11	80	2
12	78	2
13	76	2
14	74	2

15	72	2
16	70	2
17	68	2
18	66	2
19	64	2
20	62	2
21	60	2
22	58	2
23	56	2
24	54	2
25	52	2
26	50	2
27	48	2
28	46	2
29	44	2
30	42	2
31	40	2
32	38	2
33	36	2
34	34	2
35	32	2
36	30	2
37	28	2
38	26	2

39	24	2
40	22	2
41	20	2
42	18	2
43	16	2
44	14	2
45	12	2
46	10	2
47	8	2
48	6	2
49	4	2
50	2	2
51	0	-

Pada tabel3.6 dapat dilihat proses pengurangan *Health Point (HP) Enemy* ketika mendapat serangan dari Tower Es dengan damage sebesar 2 point .

Assumsi Tower Listrik menyerang *Enemy* .

Pengurangan HPE (*Health Point Enemy*) dapat dihitung dengan rumus :

HPE = HPE sebelum – DPT

Tabel 3.7 Assumsi Tower Listrik menyerang *Enemy*

NO	*Enemy*	
	HP	DPT
1	100	5
2	95	5
3	90	5
4	85	5
5	80	5
6	75	5
7	70	5
8	65	5
9	60	5
10	55	5
11	50	5
12	45	5
13	40	5
14	35	5
15	30	5
16	25	5
17	20	5
18	15	5
19	10	5
20	5	5
21	0	-

Pada tabel3.7 dapat dilihat proses pengurangan *Health Point (HP) Enemy* ketika mendapat serangan dari Tower Listrik dengan damage sebesar 5 point .

Keterangan :

DPT = Damage yang di hasilkan tower terhadap *enemy*

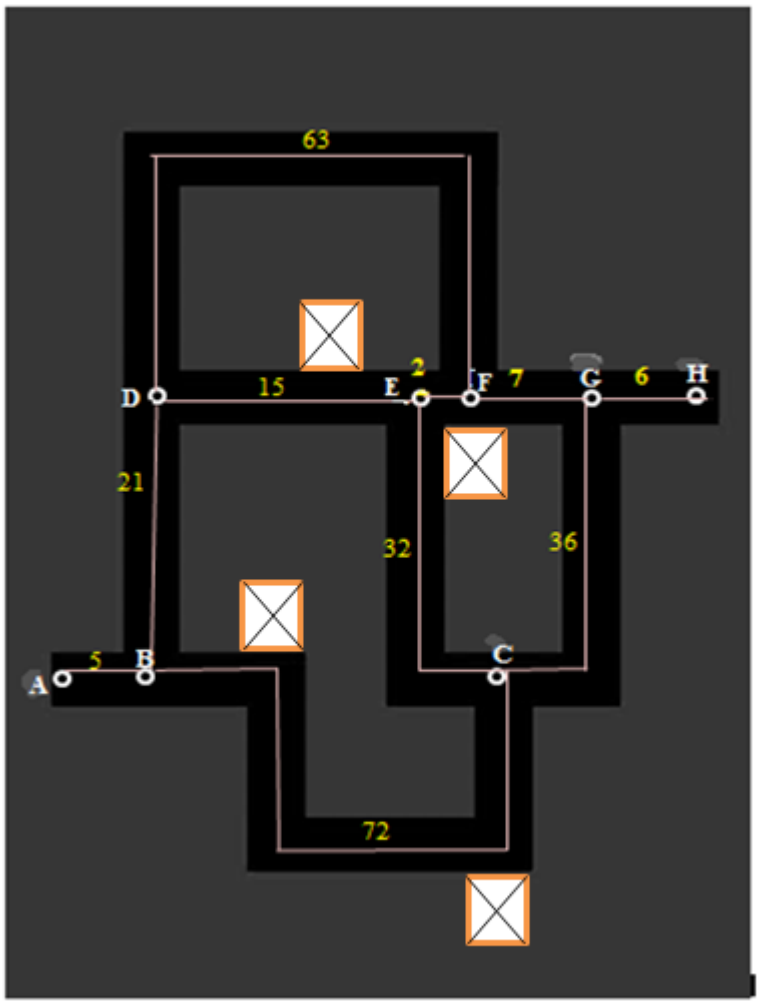
HPE = *Health Point* yang dimiliki oleh *enemy*

HPE sebelum = *Health Point* yang dimiliki oleh *enemy* sebelum terkena serangan oleh tower

3.4. Analisis Algoritma

3.4.1. Analisis Proses Algoritma Genetika

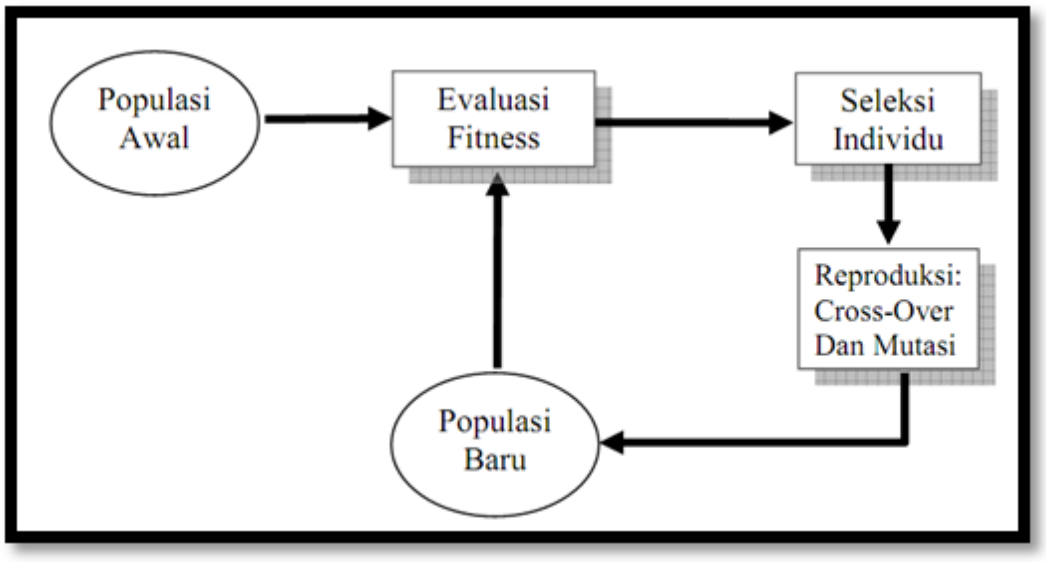
Persoalan pencarian jalur pada game ini diselesaikan dengan menggunakan algoritma genetika. Dalam algoritma genetika , pembentukan generasi baru (anak) dilakukan dengan tiga operasi , yaitu seleksi , crossover dan mutasi , kemudian melakukan evaluasi terhadap setiap populasi dengan menghitung nilai fitness setiap kromosom hingga kriteria berhenti dipenuhi. Bila kriteria berhenti belum terpenuhi , maka akan dibentuk lagi generasi baru dengan mengulangi langkah seleksi , crossover dan mutasi. Beberapa kriteria berhenti yang sering digunakan antara lain berhenti pada generasi tertentu , berhenti setelah dalam beberapa generasi berturut-turut didapatkan nilai fitness tertinggi/terendah, berhenti bila dalam n generasi berikutnya tidak diperoleh nilai fitness yang lebih tinggi/rendah.



Gambar 3.2 Play area dengan Asumsi Peletakan Tower

Gambar 3.2 menunjukan desain play area dimana terdapat pada setiap persimpangan jalur. Dalam desain terdapat 8 buah titik yang merupakan acuan dalam perhitungan pada setiap jalur yang akan di gunakan. *Start Point* pada map dimulai dari node 0 yang merupakan titik spawn enemy pertama kali, sedangkan *Goal Point* terletak pada node 7 merupakan tempat tujuan terakhir *enemy.* Kotak yang terdapat pada gambar 3.2 merupakan asumsi peletakan tower pada *tile – tile* yang ada. Untuk setiap pemasangan tower mempunyai nilai sebesar 20. Apabila pada jalur dari node B

ke node C terdapat tower yang di pasang maka jarak yang semula bernilai 72 akan di tambah dengan nilai 20 dan akan menjadi 92 . Dalam kasus peletekan tower pada persimpangan yang dilewati 2 jalur node pada tiap jalur node akan dikenakan nilai sebesar 20 . Contohnya , apabilah melakukan pemasangan tower pada jalur node E – F serta node C – E maka nilai untuk jalur node E – F yang semula mempunyai nilai 2 akan menjadi 22 sedangkan untuk node C – E yang semula mempunyai nilai sebesar 32 akan menjadi 52. Adapun proses yang terjadi dalam algoritma genetika dapat dilihat pada gambar 3.3 .



Gambar 3.3 Alur Proses Algoritma Genetika

3.4.1.1 Inisialisasi

Inisialisasi dilakukan dengan membentuk generasi baru (anak). Populasi inisialisasi dibangkitkan secara random , misalkan didapatkan 10 kromosom sebagai berikut :

Kromosom[1] : A B D E G C F H Kromosom[6] : A B D E G C F H
Kromosom[2] : A B C E D F G H Kromosom[7] : A B C G F E D H
Kromosom[3] : A B C G F E D H Kromosom[8] : A B D F E C G H
Kromosom[4] : A B C E D F G H Kromosom[9] : A B C G F E D H
Kromosom[5] : A B D F E C G H Kromosom[10]: A B D F E C G H

3.4.1.2 Evaluasi Kromosom

Evaluasi kromosom dilakukan dengan menghitung nilai fitness dari kromosom yang telah dibangkitkan. Perhitungan nilai fitness dari kromosom [1] yaitu A B D E G B F H diperoleh dengan cara menjumlahkan biaya dari node A – B , B – D , D – E , E – G , G – B , B – F , F – H serta menjumlah nilai tower yang terdapat pada tiap jalur .

Tabel 3.8 Nilai Jarak Antar Node

Node	Node yang di lalui	Nilai
A – B	A – B	5
B – C	B – C + tower + tower	112
B – D	B – D	21
C – E	C – E	34
C – G	C – G	36
D – E	D – E + tower	35
D – F	D – F	63
E – C	E – C + tower	52
E – D	E – D	15
F – E	F – E	22
F – G	F – G	7
G – C	G – C	36
G – F	G – F	7
G – H	G – H	6

Tabel 3.9 Nilai Fitness dari tiap Kromosom

Kromosom	Node Yang dilalui	Fitness
1	A B D E G C F H	213
2	A B C E D F G H	242
3	A B C G F E D H	261
4	A B C E D F G H	242
5	A B D F E C G H	205
6	A B D E G C F H	213
7	A B C G F E D H	261
8	A B D F E C G H	205
9	A B C G F E D H	261
10	A B D F E C G H	205

Nilai jarak antar node adalah nilai yang di dapat dari penjumlahan jarak antara 1 titik node ke titik node yang dituju sedangkan nilai fitness dari tiap kromosom didapatkan dengan cara menjumlahkan titik – titik node yang ada sesuai dengan node – node pada jalur yang dipilih.

3.4.1.3 Seleksi Kromosom

Hal yang pertama kali dilakukan dalam proses seleksi adalah mencari nilai perhitungan invers kromosom (Q) . Nilai perhitungan invers kromosom (Q) akan dihitung menggunakan persamaan 2.1 :

Nilai inverse kromosom pertama Q [1] :

$$Q\,[1] = \frac{1}{213} = 0,0046$$

Tabel 3.10 Nilai Invers Fitness dari tiap Kromosom

Kromosom	Nilai Invers [Q]
1	0.0046
2	0.0041
3	0.0038
4	0.0041
5	0.0048
6	0.0046
7	0.0038
8	0.0048
9	0.0038
10	0.0048

Total = Q [1] + Q [2] + Q [3] + Q [4] + Q [5] + Q [6] + Q [7] + Q [8] + Q [9]
+ Q [10]

= 0,0046 + 0,0041 + 0,0038 + 0,0041 + 0,0048 + 0,0046 + 0,0038 + 0,0048
+ 0,0038 + 0,0048

= 0,0432

Setelah didapatkan nilai total invers , kemudian di cari nilai probabilitas (P) dari individu yang kemudian akan digunakan untuk mencari nilai kumulatif probabilitas (C) yang akan digunakan dalam proses seleksei menggunakan metode *roulette – wheel* . Untuk mencari nilai probabilitas (P) digunakan persamaan 2.2 :

Nilai probabilitas kromosom pertama P [1] :

$$P\,[1] = \frac{0{,}0046}{0{,}0432} = 0{,}106$$

Sehingga pada tabel 3.11 diperoleh untuk nilai probabilitas dari 10 kromosom yang dibangkitkan.

Tabel 3.11 Nilai Probabilitas dari tiap Kromosom

Kromosom	Nilai Probabilitas [P]
1	0,106
2	0,094
3	0,087
4	0,094
5	0,111
6	0,106
7	0,087
8	0,111
9	0,087
10	0,111

Untuk proses seleksi digunakan roulette wheel , untuk itu terlebih dahulu dicari nilai kumulatif probabilitasnya (C) dengan persamaan 2.3 :

C[1] = 0,106

C[2] = 0,106 + 0,094 = 0,200

C[3] = 0,200 + 0,087 = 0,287

C[4] = 0,287 + 0,094 = 0,381

C[5] = 0,381 + 0,111 = 0,492

C[6] = 0,492 + 0,106 = 0,598

C[7] = 0,598 + 0,087 = 0.685
C[8] = 0,685 + 0,111 = 0,796
C[9] = 0,796 + 0,087 = 0,883
C[10]= 0,883 + 0,111 = 0,994

Setelah dihitung kumulatif probabilitasnya maka proses seleksi menggunakan roulette – wheel dapat dilakukan. Prosesnya adalah dengan membangkitkan bilangan acak R dalam range 0 – 1 . Putar roulette – wheel sebanyak jumlah populasi yaitu 10 kali (bangkitkan bilangan acak R) .

R [1] = 0,587
R [2] = 0,142
R [3] = 0,511
R [4] = 0,211
R [5] = 0,743
R [6] = 0,432
R [7] = 0,104
R [8] = 0,392
R [9] = 0,501
R[10]= 0,657

Angka acak pertama R[1] adalah lebih besar dari C[1] dan lebih kecil daripada C[2] . Maka kromosom [2] akan digunakan sebagai kromosom pada populai baru :

Kromosom [1] = Kromosom [2]
Kromosom [2] = Kromosom [1]
Kromosom [3] = Kromosom [4]
Kromosom [4] = Kromosom [3]
Kromosom [5] = Kromosom [6]

Kromosom [6] = Kromosom [5]
Kromosom [7] = Kromosom [7]
Kromosom [8] = Kromosom [8]
Kromosom [9] = Kromosom [9]
Kromosom [10] = Kromosom [10]

Populasi baru akan terbentuk :

Kromosom [1] = [2] = A B C E D F G H
Kromosom [2] = [1] = A B D E G C F H
Kromosom [3] = [4] = A B C E D F G H
Kromosom [4] = [3] = A B C G F E D H
Kromosom [5] = [6] = A B D E G C F H
Kromosom [6] = [5] = A B D F E C G H
Kromosom [7] = [7] = A B C G F E D H
Kromosom [8] = [8] = A B D F E C G H
Kromosom [9] = [9] = A B C G F E D H
Kromosom [10] = [10] = A B D F E C G H

3.4.1.4 Crossover

Melakukan operasi crossover , satu bagian kromosom dengan tetap menjaga urutan node yang bukan bagian dari kromosom tersebut. Misalkan ditentukan *crossover* probabilitas adalah sebesar 10%

Bangkitkan bilangan acak sebanyak jumlah populasi yaitu 10 kali .

R [1] = 0,637
R [2] = 0,192
R [3] = 0,551
R [4] = 0,252
R [5] = 0,792

R [6] = 0,480

R [7] = 0,265

R [8] = 0,425

R [9] = 0,598

R[10]= 0,752

Kromosom yang mempunyai R[k] terendah misalkan 2 dan 4 .Melakukan *Partial – Mapped Crossover (PMX)* .

Kromosom [2] >< Kromosom [4]

A B D E G C F H

A B C G F E D H

Disini substring yang akan di acak adalah substring ke 5 pada setiap kromosom

G

⇕

F

Keturunan 1 = A B D E F C G H

Keturunan 2 = A B C F G E D H

3.4.1.5 Mutasi

Proses mutasi yang akan digunakan adalah *swapping mutation* yaitu proses dengan menukarkan gen yang dipilih secara acak dengan gen sesudahnya.

Hitung total gen = jumlah gen dalam kromosom * jumlah kromosom

= 8 * 10

= 80

Memilih posisi gen yang mengalami mutasi dengan membangkitkan bilangan acak antara 1 sampai panjang total gen yaitu 1 – 80 , Apabila pm = 10%, maka jumlah gen yang akan dimutasi adalah = 0,1 x 80 = 8. Setelah diacak pada posisi 4 , 16 , 20 , 32 , 50 , 60 , 68 , 76

Proses mutasi

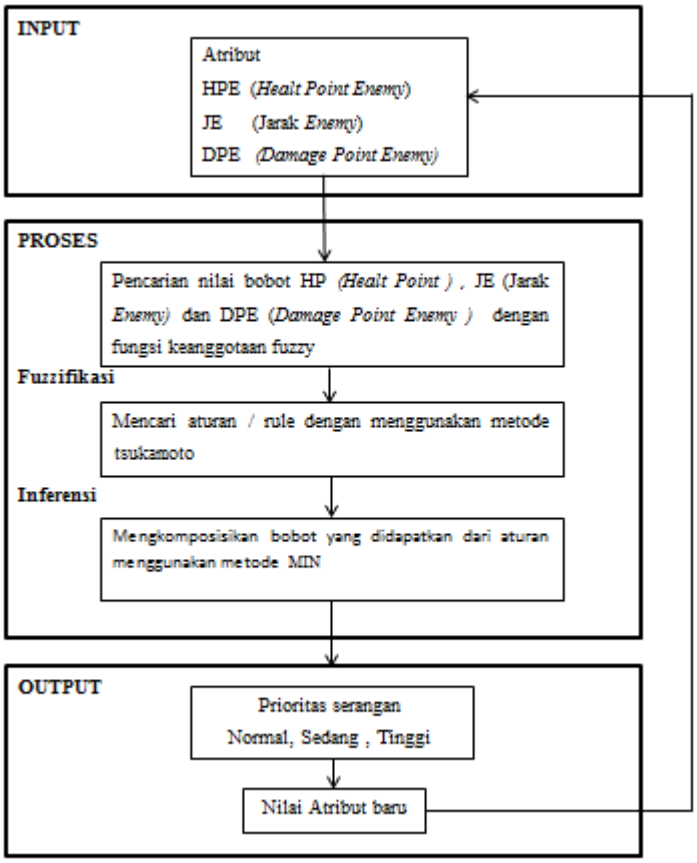
K [1] = [A B D E F C G H]

K [1] = [A B D E F C G H] ⟶ [A B D F E C G H]

Mengevaluasi setiap populasi dengan menghitung nilai fitness setiap individu pada generasi ke 10 . Dari hasil perhitungan didapatkan gen terbaik adalah A B D F E C G H dengan fitness sebesar 0.0048 dan probabilitas sebesar 0,111 . Jadi rute terbaik yang diambil adalah dari node A , kemudian node B , kemudian node D , kemudian node F , kemudian node E , kemudian node C , kemudian node G dan terakhir adalah node H.

3.4.2. Analisis Proses Fuzzy Logic

Game yang akan dibuat bercerita tentang tower defense dimana *player* akan meletakan 5 tower dengan posisi sesuai dengan strategi yang diinginkan *player*. Pada setiap *wave* akan tersedia *enemy* dengan tingkat kesulitan yang berbeda- beda . Setiap *enemy* akan mempunyai 3 parameter berupa *HPE (Health Point Enemy)* , JE (Jarak *Enemy)* dan DPE (*Damage Point Enemy)* yang akan menjadi inputan pada *fuzzy logic*. Gambar 3.4 menunjukkan proses *input* hingga *output* yang akan di proses oleh *fuzzy logic*.



Gambar 3.4 Proses Serangan tower terhadap *enemy*

Pada Gambar 3.4 , dimulai dari proses input , input atau masukan berupa HPE (*Health Point Enemy)* , JE (Jarak *Enemy)* dan DPE (*Damage Point Enemy)* . Inputan tersebut akan masuk ke proses *fuzziikasi* untuk didapatkan bobot berupa HPE (*Health Point Enemy)* , JE (Jarak *Enemy)* dan DPE (*Damage Point Enemy)* menggunakan fungsi keanggotaan *fuzzy* , setelah mendapatkan bobot , maka dicari aturan atau *rule* dengan menggunakan Metode *Tsukamoto.* Setelah aturan didapatkan , maka akan

dikomposisikan bobot yang didapatkan dengan operator zadeh , dalam kasus ini akan menggunakan operator *and* yang kemudian akan dicari nilai terkecil sesuai dengan metode MIN pada *fuzzy tsukamoto.* Setelah itu akan didapatkan nilai keluaran terhadap prioritas tower dalam menghancurkan *enemy.*

Data masukan yang ditunjukan pada Gambar 3.4 , harus dihitung dalam fungsi keanggotaan *fuzzy* , data masukan *fuzzy* sendiri terdiri dari HPE (*Health Point Enemy)* , JE (Jarak *Enemy)* dan DPE (*Damage Point Enemy).*

3.4.2.1. Basis Pengetahuan

Pada sistem ini berbasis pengetahuan berisi kriteria pengambilan keputusan dan himpunan *fuzzy* masing-masing kriteria. Kriteria tersebut digolongkan menjadi himpunan bahasa Variabel penentuan sebagai berikut :

1. HPE (*Health Point Enemy)* : Sedikit , Sedang dan Banyak
2. JE (Jarak *Enemy)* : Dekat , Sedang dan Jauh
3. DPE (*Damage Point Enemy)* : Kecil , Sedang dan Kuat

Sedangkan semesta pembicaraan yang menjelaskan tentang rentang nilai yang terdapat pada HPE , JE dan DPE dapat dilihat pada tabel 3.12.

Tabel 3.12 Semesta Pembicara

Nama Variabel	Semesta Pembicaraan	Keterangan
Health Point Enemy	[0 – 100]	*Health Point* yang dimiliki oleh *enemy*
Jarak Enemy	[0 – 100]	Jarak yang *enemy* dengan *Tower*
Damage Point Enemy	[0 – 10]	*Damage Point* yang dimiliki oleh *enemy*

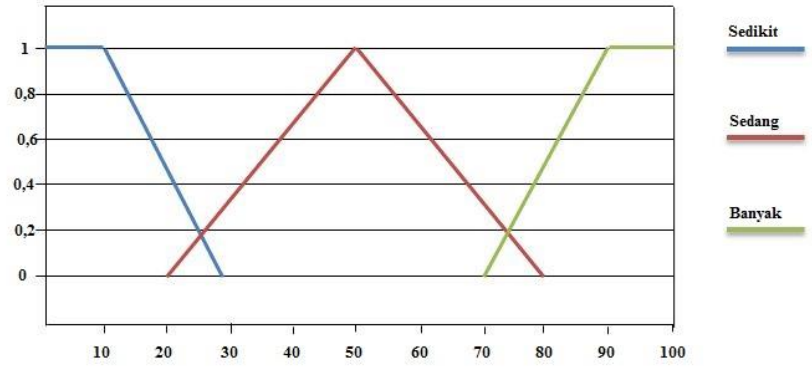
Fungsi Derajat Keanggotaan

Pada fungsi derajt keanggotaan menggunakan Tsukamoto dibagi menjadi keanggotaan jumlah HPE , JE dan DPE .

Fungsi keanggotaan DPE (*Damage Point Enemy)* diambil berdasarkan jumlah DPE yang didapatkan dari rentang 10 – 100.

Tabel 3.13 Fungsi Keanggotaan HPE

Fungsi Keanggotaan	Rentang Nilai
Sedikit	0 – 30
Sedang	20 – 80
Banyak	70 – 100



Gambar 3.5 Grafik Fungsi Keanggotaan HPE

a. Himpunan *Fuzzy* sedikit

Himpunan *Fuzzy* sedikit memiliki domain nilai antara 0 – 30 , dimana fungsi keanggotaanya dihitung berdasar fungsi transpesium .. (2.4) .

$$\mu\text{Sedikit}(x) = \begin{cases} 1; & ,x \leq 10 \\ \dfrac{30-x}{30-10} & ,10 \leq x \leq 30 \\ 0; & ,x \geq 30 \end{cases}$$

b. Himpunan *Fuzzy* sedang

Himpunan *Fuzzy* sedang memiliki domain nilai antara 20 – 80 , dimana fungsi keanggotaanya dihitung berdasar fungsi segitiga .. (2.5) .

$$\mu\text{Sedang}(x) = \begin{cases} 0; & ,x \leq 20 \text{ atau } x \geq 80 \\ \dfrac{x-20}{50-20} & ,20 \leq x \leq 50 \\ \dfrac{80-x}{80-50} & ,50 \leq x \leq 80 \end{cases}$$

c. Himpunan *Fuzzy* banyak

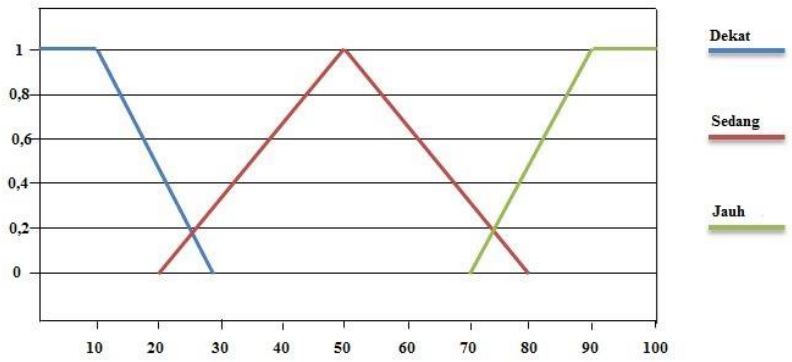
Himpunan *Fuzzy* banyak memiliki domain nilai antara 70 – 100 , dimana fungsi keanggotaanya dihitung berdasar fungsi transpesium .. (2.4) .

$$\mu\text{Banyak}(x) = \begin{cases} 0; & ,x \leq 70 \\ \dfrac{x-70}{100-70} & ,70 \leq x \leq 100 \\ 1; & ,x \geq 100 \end{cases}$$

Fungsi keanggotaan JE (Jarak *Enemy*) diambil berdasarkan jumlah JE yang didapatkan dari rentang 1 – 10.

Tabel 3.14 Fungsi Keanggotaan JE

Fungsi Keanggotaan	Rentang Nilai
Dekat	0 – 30
Sedang	20 – 80
Jauh	70 – 100



Gambar 3.6 Grafik Fungsi Keanggotaan JE

a. Himpunan *Fuzzy* dekat

Himpunan *Fuzzy* dekat memiliki domain nilai antara 0 – 30 , dimana fungsi keanggotaanya dihitung berdasar fungsi transpesium .. (2.4) .

$$\mu\text{Dekat}(x) = \begin{cases} 1; & ,x \leq 1 \\ \dfrac{30-x}{30-1} & ,10 \leq x \leq 30 \\ 0; & ,x \geq 30 \end{cases}$$

b. Himpunan *Fuzzy* sedang

Himpunan *Fuzzy* sedang memiliki domain nilai antara 20 – 80 , dimana fungsi keanggotaanya dihitung berdasar fungsi segitiga .. (2.5) .

$$\mu\text{Sedang}(x) = \begin{cases} 0; & ,x \leq 20 \text{ atau } x \geq 80 \\ \dfrac{x-20}{50-20} & ,20 \leq x \leq 50 \\ \dfrac{80-x}{80-50} & ,50 \leq x \leq 80 \end{cases}$$

c. Himpunan *Fuzzy* Jauh

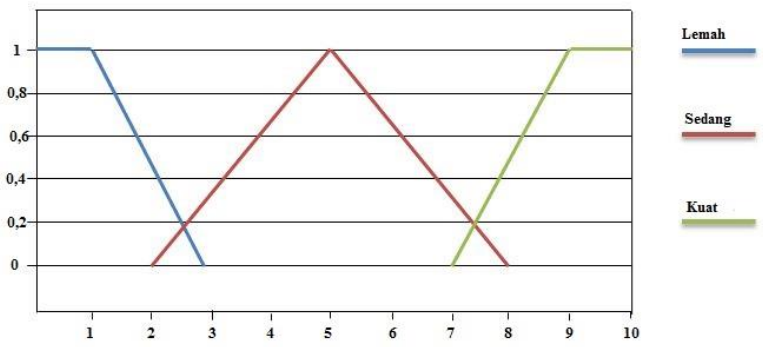
Himpunan *Fuzzy* Jauh memiliki domain nilai antara 70 – 100 , dimana fungsi keanggotaanya dihitung berdasar fungsi transpesium .. (2.4) .

$$\mu\text{Jauh}(x) = \begin{cases} 0; & ,x \leq 70 \\ \dfrac{x-70}{100-70} & ,70 \leq x \leq 100 \\ 1; & ,x \geq 100 \end{cases}$$

Fungsi keanggotaan DPE (*Damage Point Enemy)* diambil berdasarkan jumlah DPE yang didapatkan dari rentang 1 – 10.

Tabel 3.15 Fungsi Keanggotaan DPE

Fungsi Keanggotaan	Rentang Nilai
Lemah	0 – 3
Sedang	2 – 8
Kuat	7 – 10



Gambar 3.7 Grafik Fungsi Keanggotaan DPE

a. Himpunan *Fuzzy* Lemah

Himpunan *Fuzzy* Lemah memiliki domain nilai antara 0 – 3 , dimana fungsi keanggotaanya dihitung berdasar fungsi transpesium .. (2.4) .

$$\mu\text{Lemah}(x) = \begin{cases} 1; & ,x \leq 1 \\ \dfrac{3-x}{3-1} & ,1 \leq x \leq 3 \\ 0; & ,x \geq 3 \end{cases}$$

b. Himpunan *Fuzzy* sedang

Himpunan *Fuzzy* sedang memiliki domain nilai antara 2 – 8 , dimana fungsi keanggotaanya dihitung berdasar fungsi segitiga .. (2.5) .

$$\mu\text{Sedang}(x) = \begin{cases} 0; & ,x \leq 2 \text{ atau } x \geq 8 \\ \dfrac{x-2}{5-2} & ,2 \leq x \leq 5 \\ \dfrac{8-x}{8-5} & ,5 \leq x \leq 8 \end{cases}$$

c. Himpunan *Fuzzy* Kuat

Himpunan *Fuzzy* Kuat memiliki domain nilai antara 70 – 100 , dimana fungsi keanggotaanya dihitung berdasar fungsi transpesium .. (2.4) .

$$\mu\text{Kuat}(x) = \begin{cases} 0; & ,x \leq 7 \\ \dfrac{x-7}{10-7} & ,7 \leq x \leq 10 \\ 1; & ,x \geq 10 \end{cases}$$

3.4.2.2. Aturan Fuzzy

Setelah pembentukan himpunan *Fuzzy* , maka dilakukan pembentukan aturan *Fuzzy*. Aturan – aturan dibentuk untuk menyatakan relasi antara input dan output. Tiap aturan merupakan suatu implikasi . Operator yang digunakan untuk input – output adalah *IF – THEN* . Proposisi yang mengikuti *IF* disebut anteseden, sedangkan

proposisi yang mengikuti *THEN* disebut konsekuen. Berikut ini adalah tetapan aturan pemilihan target tembak pada NPC *Tower* .

Tabel 3.16 Tetapan Aturan Pemilihan Target Tembak NPC *Tower*

ANTESEDEN						**KONSEKUEN**	
Operator	**Jarak**	**Operator**	**HPE**	**Operator**	**DPE**	**Operator**	**Prioritas**
IF	Dekat	AND	Sedikit Sedang Banyak	AND	Lemah Sedang Kuat	THEN	Tinggi Normal Sangat Tinggi
	Sedang		Sedikit Sedang Banyak		Lemah Sedang Kuat		Tinggi Normal Sangat Tinggi
	Jauh		Sedikit Sedang Banyak		Lemah Sedang Kuat		Tinggi Normal Sangat Tinggi
ANTESEDEN						**KONSEKUEN**	
Operator	**HPE**	**Operator**	**JE**	**Operator**	**DPE**	**Operator**	**Prioritas**
IF	Sedikit	AND	Dekat Sedang Jauh	AND	Lemah Sedang Kuat	THEN	Tinggi Normal Sangat Tinggi
	Sedang		Dekat Sedang Jauh		Lemah Sedang Kuat		Tinggi Normal Sangat Tinggi
	Banyak		Dekat Sedang		Lemah Sedang		Tinggi Normal

			Jauh		Kuat		Sangat Tinggi
ANTESEDEN						**KONSEKUEN**	
Operator	**DPE**	**Operator**	**HPE**	**Operator**	**JE**	**Operator**	**Prioritas**
IF	Lemah	AND	Sedikit Sedang Banyak	AND	Dekat Sedang Jauh	THEN	Tinggi Tinggi Tinggi
	Sedang		Sedikit Sedang Banyak		Dekat Sedang Jauh		Tinggi Normal Normal
	Kuat		Sedikit Sedang Banyak		Dekat Sedang Jauh		Sangat Tinggi Sangat Tinggi Sangat Tinggi

3.4.2.3. Analisis Proses

Apabila terdapat 3 buah *enemy*. *Enemy* pertama di asumsikan mempunyai HPE dengan nilai 21 , JE dengan nilai 10 dan DPE dengan nilai 6 . *Enemy* kedua di asumsikan mempunyai HPE dengan nilai 30 , JE dengan nilai 30 dan DPE dengan nilai 5. *Enemy* ketiga di asumsikan mempunyai HPE dengan nilai 80 , JE dengan nilai 70 dan DPE dengan nilai 10 :

***Enemy* pertama :**

$$\mu hpe(sedikit) = \frac{30-21}{30-10} = \frac{9}{20} = 0{,}45$$

$$\mu JE(Dekat) = \frac{30-10}{30-10} = \frac{20}{20} = 1$$

$$\mu DPE(Sedang) = \frac{8-6}{8-5} = \frac{2}{3} = 0{,}66$$

***Enemy* kedua :**

$$\mu hpe(sedang) = \frac{50-30}{50-30} = \frac{20}{20} = 1$$

$$\mu JE(Sedang) = \frac{50-30}{50-30} = \frac{20}{20} = 1$$

$$\mu DPE(Sedang) = \frac{8-5}{8-5} = \frac{3}{3} = 1$$

***Enemy* ketiga :**

$$\mu hpe(Banyak) = \frac{100-80}{100-70} = \frac{20}{30} = 0{,}66$$

$$\mu JE(Jauh) = \frac{100-70}{100-70} = \frac{30}{30} = 1$$

$$\mu DPE(Kuat) = \frac{10-10}{10-7} = \frac{0}{3} = 0$$

Setelah proses *fuzzifikasi* dilakukan dan didapatkan nilai bobot *fuzzy* , maka proses selanjutnya merupakan pencarian aturan (*rule*) dan dilanjutkan pada proses inferensi , yaitu proses memperhitungkan semua aturan yang ada dalam basis pengetahuan.

IF HPE = Sedikit
AND JE = Dekat
AND DPE = Sedang
THEN ….

$\alpha_{predikat1}$ = μ $HPE\ Sedikit$ ∩ μ $JE\ Dekat$ ∩ μ $DPE\ Sedang$
= min (μ $HPE\ Sedikit$ ∩ μ $JE\ Dekat$ ∩ μ $DPE\ Sedang$
= min (0,45 ; 1 ; 0,66)
= 0,45

IF HPE = Sedang
AND JE = Sedang
AND DPE = Sedang
THEN ….

$\alpha_{predikat2}$ = μ $HPE\ Sedang$ ∩ μ $JE\ Jauh$ ∩ μ $DPE\ Sedang$
= min (μ $HPE\ Sedang$ ∩ μ $JE\ Jauh$ ∩ μ $DPE\ Sedang$
= min (1 ; 1 ; 1)
= 1

IF HPE = Banyak
AND JE = Jauh
AND DPE = Kuat
THEN ….

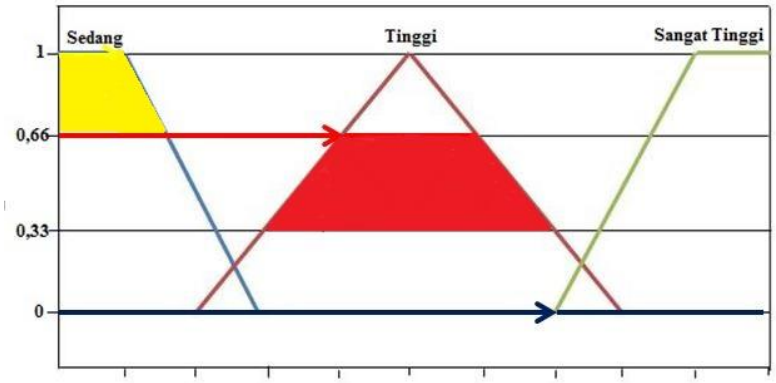
$\alpha_{predikat3}$ = μ $HPE\ Banyak$ ∩ μ $JE\ Jauh$ ∩ μ $DPE\ Kuat$

= min (μ $HPE\ Banyak$ ∩ μ $JE\ Jauh$ ∩ μ $DPE\ Kuat$

= min (0,66 ; 1 ; 0)

= 0

Proses inferensi pada *fuzzy tsukamoto* menggunakan metode MIN , yaitu mengambil nilai terkecil dari predikat , seingga nilai predikat1 = 0,45 nilai predikat2 = 1 nilai predikat3 = 0 . Maka nilai terkecil terdapat pada predikat2 yaitu sebesar 0 . Hasil kesimpulan aturan predikat3 dapat dilihat pada gambar 3.8 .



Gambar 3.8 Daerah Hasil Komposisi Aturan Metode Min Predikat2

Berdasarkan Gambar 3.8 *enemy* dengan predikat3 terletak pada area dengan prioritas “Sangat Tinggi” dan akan menjadi prioritas tower dalam melakukan serangan .

3.5. Analisis Kebutuhan Sistem

Analisis kebutuhan sistem ini mencakup kebutuhan fungsional dan kebutuhan non-fungsional. Kebutuhan fungsional akan dijelaskan menggunakan *Object Oriented Modeling (OOM)* dengan *Unified Modeling Language*.

Kebutuhan fungsional merupakan rangkaian kebutuhan mencakup reaksi objek terhadap masukan dan apa yang harus dilakukan objek ketika menemukan

kondisi tertentu. Kebutuhan fungsional dilihat dari sudut pandang pengguna terhadap sistem. Berikut analisis kebutuhan fungsional sistem yang akan dirancang:

a) Sistem atau *game* dapat dimainkan oleh siapa saja dan hanya satu pemain.
b) Sistem menggunakan kaidah-kaidah interaksi manusia dan komputer (IMK) atau *user friendly*.

3.5.1. Use Case

Use case adalah cara penggambaran sistem yang digambarkan berdasarkan sudut pandang aktor atau objek yang bekerja.

3.5.1.1. Definisi Aktor

Dalam *game* atau sistem ini, terdapat satu aktor yang berperan, yaitu *Player* atau pemain. *Player* atau pemain adalah orang yang befungsi untuk memainkan permainan dengan mengikuti *gameplay* yang ada.

3.5.1.2. Definisi Use Case

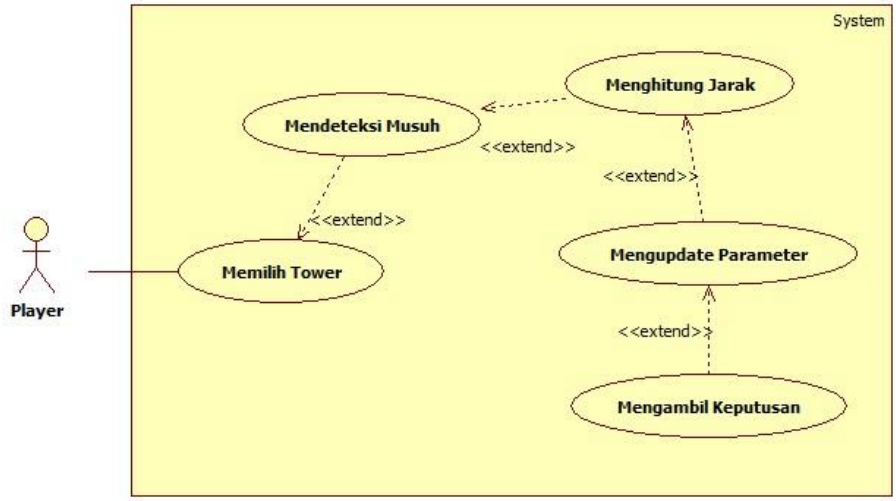
Tabel 3.17 adalah deskripsi pendefinisian *use case* pada *Game "Benteng Bertahan"* :

Tabel 3.17 Definisi Use Case

No	Nama	Deskripsi
1.	Memilih Tower	Proses untuk memilih Tower ke Stage
2.	Mendeteksi Musuh	Proses untuk menedetksi musuh ke area Tower
3.	Menghitung Jarak	Proses untuk jarak antara Tower dan musuh
4.	Mengupdate Parameter	Proses untuk mengupdate nilai Fuzzy
5.	Mengambil Keputusan	Proses untuk mengambil keputusan

3.5.1.3. Use Case Diagram

Terdapat 5 (Lima) buah *use case* seperti pada Tabel 3.18. Dalam *use case* tersebut proses perhitungan logika fuzzy dan alagoritma Genetika akan terlihat penggunaannya. Gambar 3.9 merupakan *use case* dari *game* yang akan dibuat.



Gambar 3.9 Use case diagram Benteng Bertahan

3.5.2. Skenario Use Case

Skenario setiap bagian pada *use case* menunjukkan proses apa yang terjadi pada setiap bagian didalam *use case* tersebut, dimana *user* memberikan perintah pada setiap bagian dan respon apa yang diberikan oleh sistem kepada *user* setelah *user* memberikan perintah pada setiap bagian-bagian *use case*.

a) Skenario Use Case Memilih Tower

Skenario *use case* Memilih Tower digambarkan seperti pada Tabel 3.18.

Tabel 3.18 Skenario Use Case viewTutorial

Identifikasi	
Nomor	1
Nama	Memilih Tower
Tujuan	Memilih tower untuk dipasangkan ke dalam stage permainan
Deskripsi	Proses untuk memilih tower
Aktor	*Player*
Skenario Utama	
Kondisi Awal	Belum terdapat tower di area permainan
Aksi Aktor	**Reaksi Sistem**
1. Pemain menekan tombol Tower 3. Pemain menekan tombol kiri mouse	2. Tower yang dipilih aktif dan dapat di drag ke area permainan. 4. Tower ditempatkan di area permainan sesuai posisi yang dipilih pemain.
Kondisi Akhir	Tower berada di area stage permainan
Skenario Alternatif	
Kondisi Awal	Belum terdapat tower di area permainan
Aksi Aktor	**Reaksi Sistem**
1. Pemain menekan tombol Tower 3. Pemain menekan tombol	2. Tower yang dipilih aktif dan dapat di drag ke area permainan. 4. Tower kembali tidak aktif.

Kondisi Akhir	Tower tidak dipasangkan di area permainan

b) Skenario Use Case Mendeteksi Musuh

Skenario *use case* Mendeteksi Musuh digambarkan seperti pada Tabel 3.19.

Tabel 3.19 Skenario Use Case Mendeteksi Musuh

Identifikasi	
Nomor	2
Nama	Mendeteksi Musuh
Tujuan	Tower mendeteksi musuh yang masuk ke dalam areanya
Deskripsi	Proses untuk mendeteksi musuh
Aktor	Tower
Skenario Utama	
Kondisi Awal	Tower sudah terpasang di area permainan, dan musuh mulai bergerak
Aksi Aktor	**Reaksi Sistem**
1. Tower mendeteksi musuh terdekat	2. Melakukan pengecekan terhadap monster yang mendekat
Kondisi Akhir	Tower mendeteksi musuh yang masuk ke area nya
Skenario Alternatif	
Kondisi Awal	Tower sudah terpasang di area permainan, dan musuh mulai

	bergerak
Aksi Aktor	**Reaksi Sistem**
1. Tower mendeteksi musuh terdekat	2. Melakukan pengecekan terhadap monster yang mendekat
Kondisi Akhir	Musuh keluar dari area Tower

c) Skenario Use Case Menghitung Jarak

Skenario *use case* Menghitung Jarak digambarkan seperti pada Tabel 3.20.

Tabel 3.20 Skenario Use Case Menghitung Jarak

Identifikasi	
Nomor	3
Nama	Menghitung Jarak
Tujuan	Menghitung jarak antara Tower dan musuh
Deskripsi	Proses untuk menghitung jarak antara tower dan musuh
Aktor	Tower
Skenario Utama	
Kondisi Awal	Musuh berada di area Tower
Aksi Aktor	**Reaksi Sistem**
1. Tower mendeteksi musuh yang masuk ke dalam area	2. Menyatakan musuh berada di dalam area Tower
3. Tower menghitung jarak antara musuh tower	4. Menghitung jarak antara musuh dan tower
Kondisi Akhir	Jarak terhitung antara jarak musuh dan tower
Skenario Alternatif	
Kondisi Awal	Musuh berada di area Tower

Aksi Aktor	Reaksi Sistem
1. Tower mendeteksi musuh yang masuk ke dalam area	2. Menyatakan musuh berada di dalam area Tower
3. Musuh keluar dari area Tower	4. Menyatakan musuh berada di luar area Tower
Kondisi Akhir	Jarak tidak diperhitungkan

d) Skenario Use Case Mengupdate Parameter

Skenario *use case* Mengupdate Parameter digambarkan seperti pada Tabel 3.21 .

Tabel 3.21 Skenario Use Case Mengupdate Parameter

Identifikasi	
Nomor	3
Nama	Mengupdate Parameter
Tujuan	Mengupdate nilai nilai parameter fuzzy dari musuh
Deskripsi	Proses perhitungan dan pembaharuan nilai untuk parameter fuzzy
Aktor	Tower
Skenario Utama	
Kondisi Awal	Musuh berada di dalam area Tower
Aksi Aktor	**Reaksi Sistem**
1. Mendeteksi musuh berada di area nya	2. Menyatakan musuh berada di area Tower
3. Menghitung jarak antara musuh dan tower	4. Menghitung jarak antara musuh dan tower
5. Mengupdate nilai parameter dari musuh	6. Melakukan pengupdatean nilai parameter fuzzy dari

	musuh
Kondisi Akhir	Nilai parameter fuzzy dari musuh telah terupdate
Skenario Alternatif	
Kondisi Awal	Musuh berada di dalam area Tower
Aksi Aktor	**Reaksi Sistem**
1. Mendeteksi musuh berada di area nya	2. Menyatakan musuh berada di area Tower
3. Mendeteksi musuh berada di luar area nya	4. Tidak melakukan pengupdatean parameter fuzzy dari musuh
Kondisi Akhir	Nilai parameter fuzzy dari musuh tetap

e) Skenario Use Case Mengambil Keputusan

Skenario *use case* Mengambil Keputusan digambarkan seperti pada Tabel 3.22 .

Tabel 3.22 Skenario Use Case Mengambil Keputusan

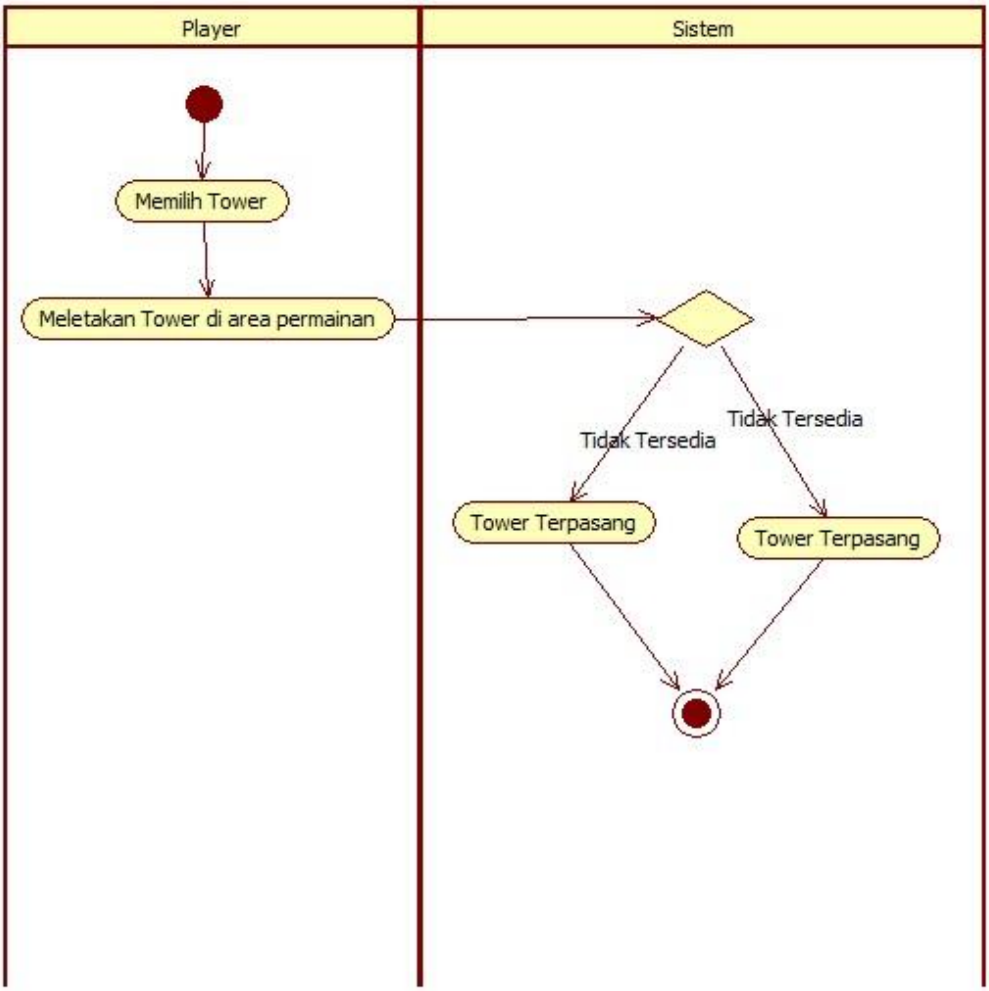
Identifikasi	
Nomor	5
Nama	Mengambil Keputusan
Tujuan	Memilih musuh yang akan diserang
Deskripsi	Proses untuk memilih musuh yang akan diserang
Aktor	Tower
Skenario Utama	
Kondisi Awal	Musuh-musuh berada di area Tower

Aksi Aktor	Reaksi Sistem
1. Mengupdate nilai parameter Fuzzy dari musuh	2. Melakukan perhitungan Fuzzy dan memberikan keputusan musuh yang diserang
3. Tower menyerang musuh	
Kondisi Akhir	Tower melakukan penyerangan terhadap musuh yang sesuai perhitungan Fuzzy
Skenario Alternatif	
Kondisi Awal	Musuh-musuh berada di area Tower
Aksi Aktor	**Reaksi Sistem**
1. Mengupdate nilai parameter Fuzzy dari musuh	
2. Mendeteksi musuh keluar dari area Tower	3. Musuh tidak masuk dalam prioritas penyerangan
Kondisi Akhir	Musuh keluar dari area Tower

3.5.3. Activity Diagram

a) Activity Diagram Memilih Tower

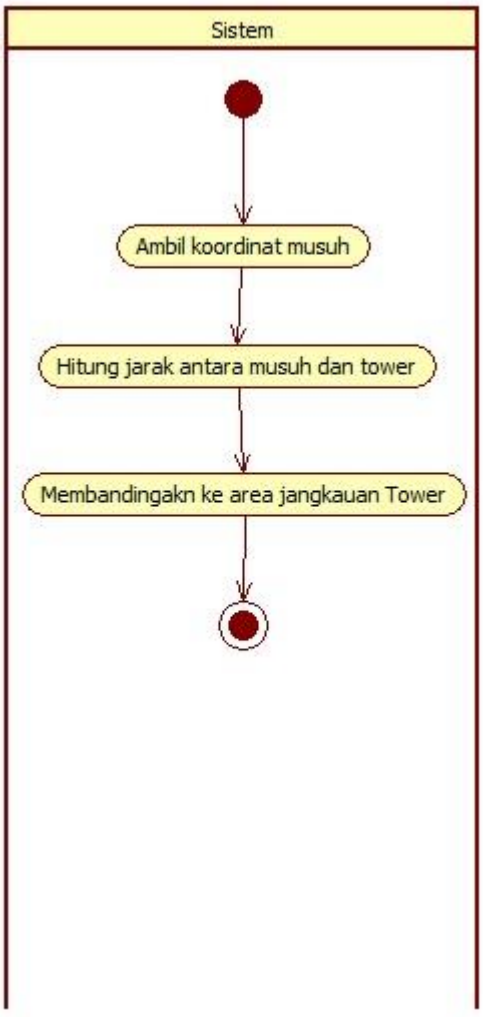
Gambar 3.10 menggambarkan aktivitas sistem pada saat proses Pemilihan tower.



Gambar 3.10 Activity diagram Pemilihan Tower

b) Activity Diagram Mendekteksi Musuh

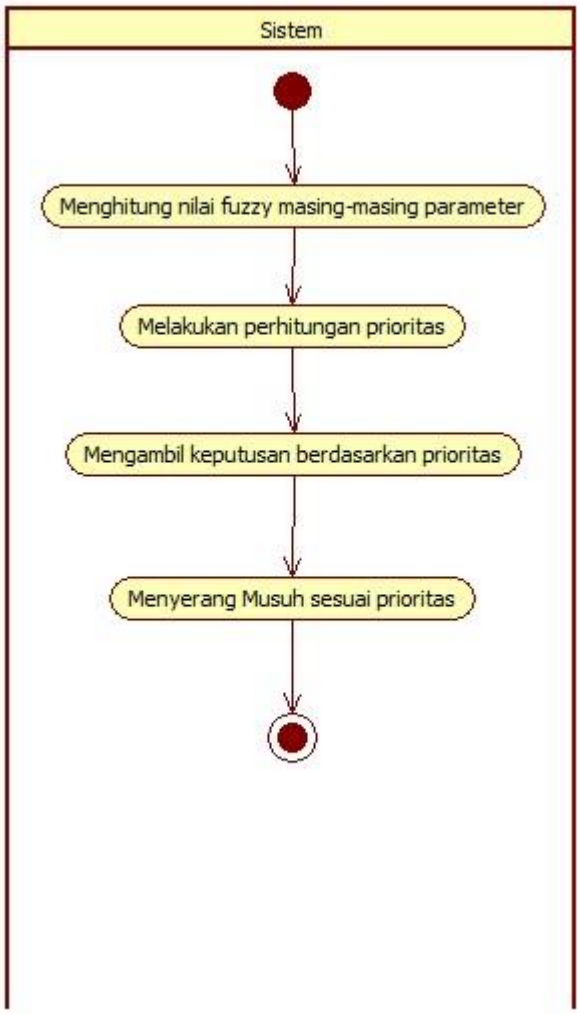
Gambar 3.11 menggambarkan aktivitas sistem pada saat proses tower mendekteksi musuh .



Gambar 3.11 Activity diagram Mendeteksi Musuh

c) Activity Diagram Mengambil Keputusan

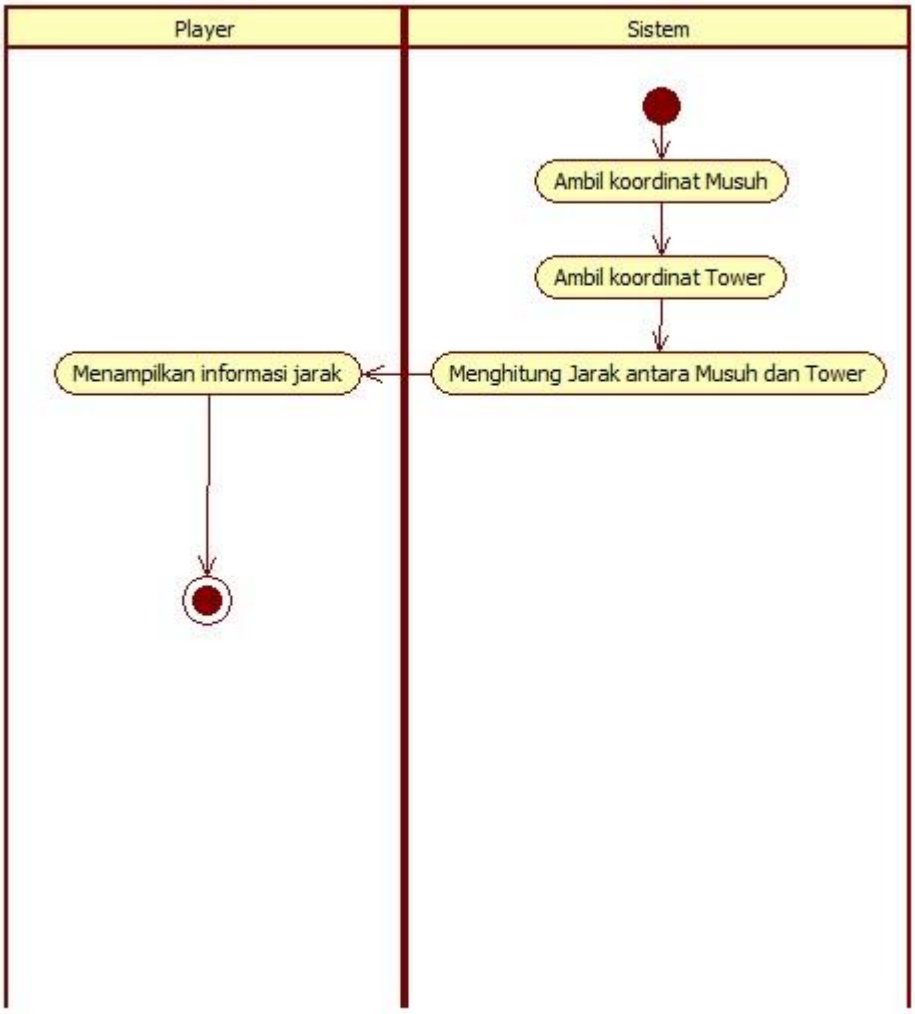
Gambar 3.12 menggambarkan aktivitas diagram tower dalam mengambil keputusan.



Gambar 3.12 Activity diagram Mengambil Keputusan

d) Activity Diagram Menghitung Jarak

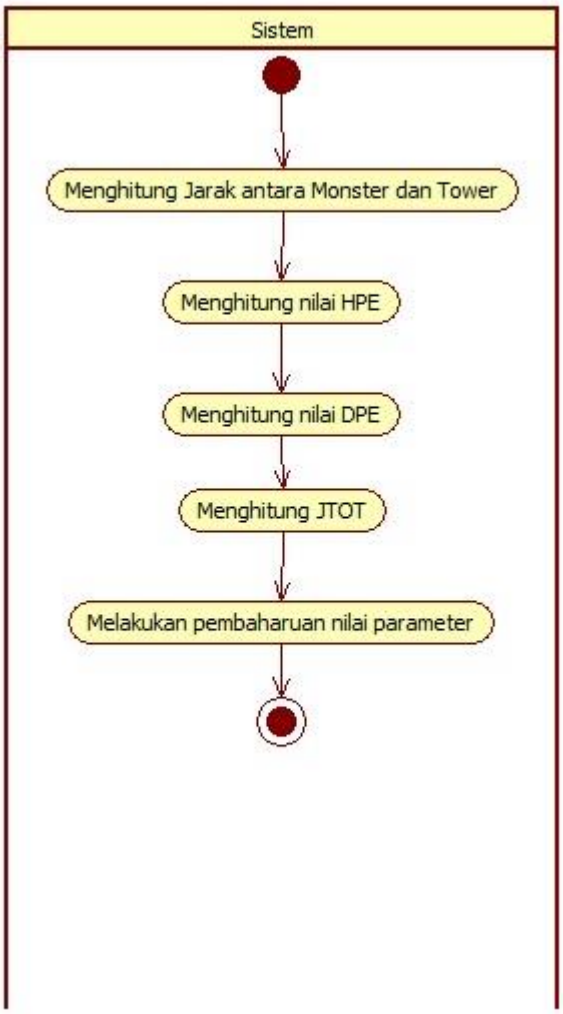
Gambar 3.13 menggambarkan aktivitas dalam menampilkan perhitungan jarak.



Gambar 3.13 Activity diagram Menghitung Jarak

e) Activity Diagram Mengupdate Parameter

Gambar 3.14 menggambarkan aktivitas dalam mengupdate parameter yang dimilikih oleh monster.



Gambar 3.14 Activity Diagram Mengupdate Parameter

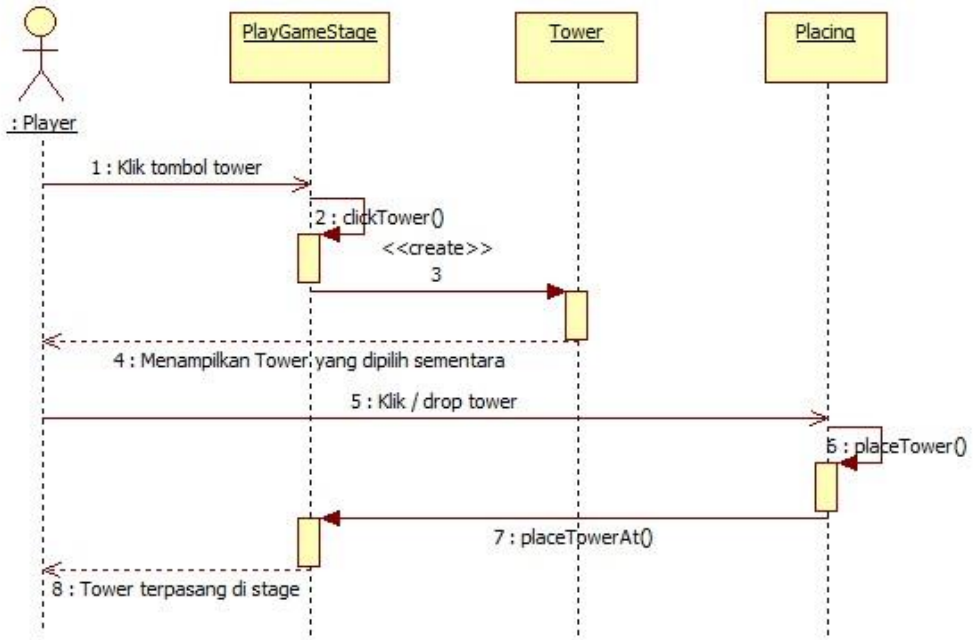
3.5.4. Sequence Diagram

a) Sequence Diagram Pemilihan Tower

Gambar 3.15 menggambarkan interaksi *Player* ketika melakukan pemilihan terhadap tower. Interaksi pertama dimulai ketika player menekan tombol klik pada tower lalu kemudian masuk kedalam class PlayGameStage.

Di kelas tersebut, *method* clickTower() berfungsi untuk mengirim *event* NavigationEvent untuk membuat objek dari kelas Tower. Lalu *player* meletakan tower di tempat yang diinginkan . Pada kelas placing akan terjadi *method* placeTower() dimana *player* berhasil membuat tower yang dia inginkan

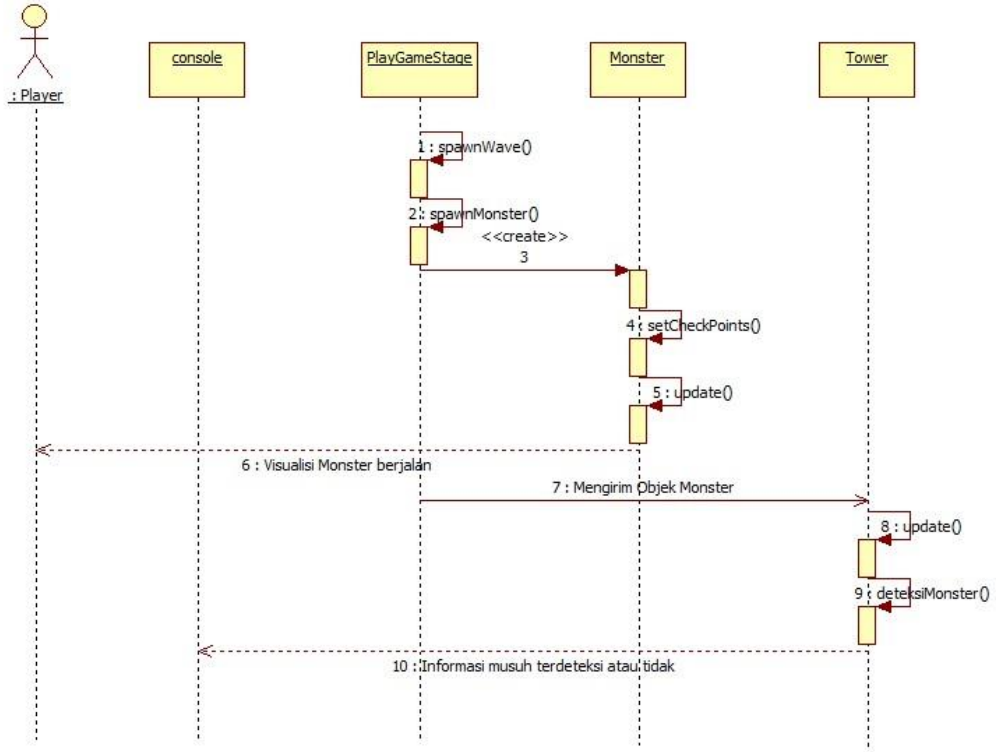
Kelas yang terlibat dalam proses ini adalah kelas PlayGameStage kelas Tower dan kelas Placing.



Gambar 3.15 Sequence diagram Pemilihan Tower

b) Sequence Diagram Mendekteksi Musuh

Gambar 3.16 menggambarkan sequence diagram mendekteksi musuh. Pada saat kelas PlayGameStage *method* spawn monster akan meng create monster. Kelas tower akan melakukan deteksi terhadap monster.



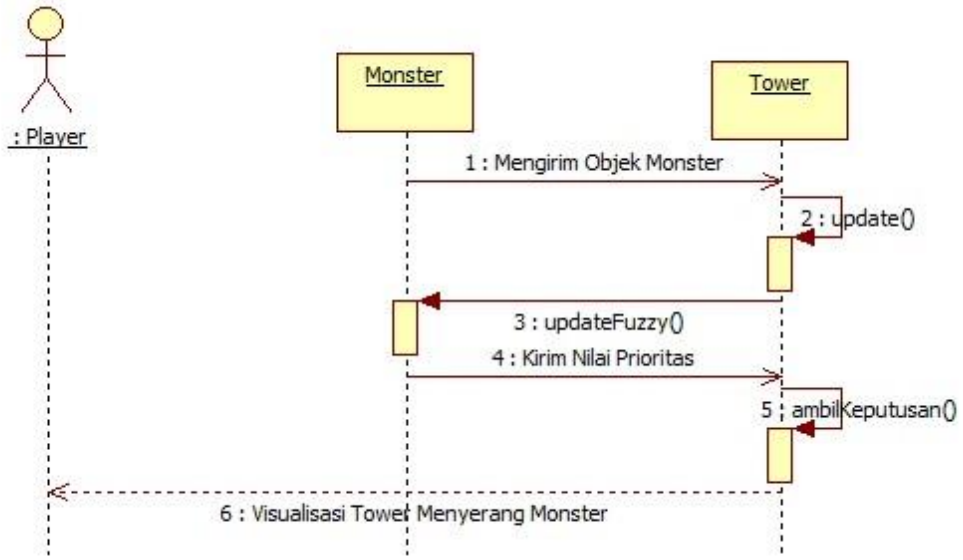
Gambar 3.16 Sequence diagram Mendekteksi Musuh

c) Sequence Diagram Mengambil Keputusan

Gambar 3.17 menggambarkan bagaiaman tower dalam mengambil keputusan. Kelas monster akan mengirim object monster yang sudah di create .

Di kelas tersebut, *method* updateFuzzy() berfungsi untuk mengupdate informasi tentang nilai – nilai fuzzy yang terdapat di dalam monster kemudian dari kelas monster akan mengirim event nilai prioritas dan terakhir tower akan melakukan

method ambilKeputusan() untuk mengambil keputusan apa yang akan dilakukan terhadap monster berdasarkan prioritas serangan.

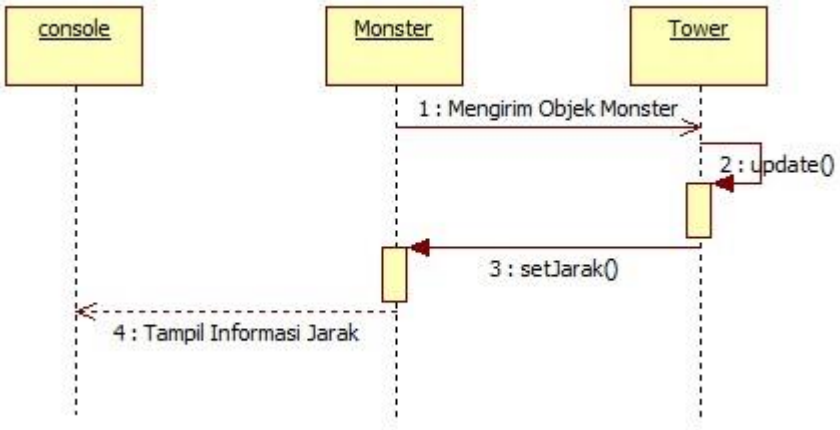


Gambar 3.17 Sequence diagram Mengambil Keputusan

d) Sequence Diagram viewInfo

Gambar 3.18 menggambarkan Proses perhitungan jarak monster terhadap tower.

Di kelas monster mengirimkan objek monster , kemudian tower akan melakukan *method* update() yang kemudian akan mengirimkan *method* SetJarak() dan akan menampilkan informasi jarak pada kelas console.

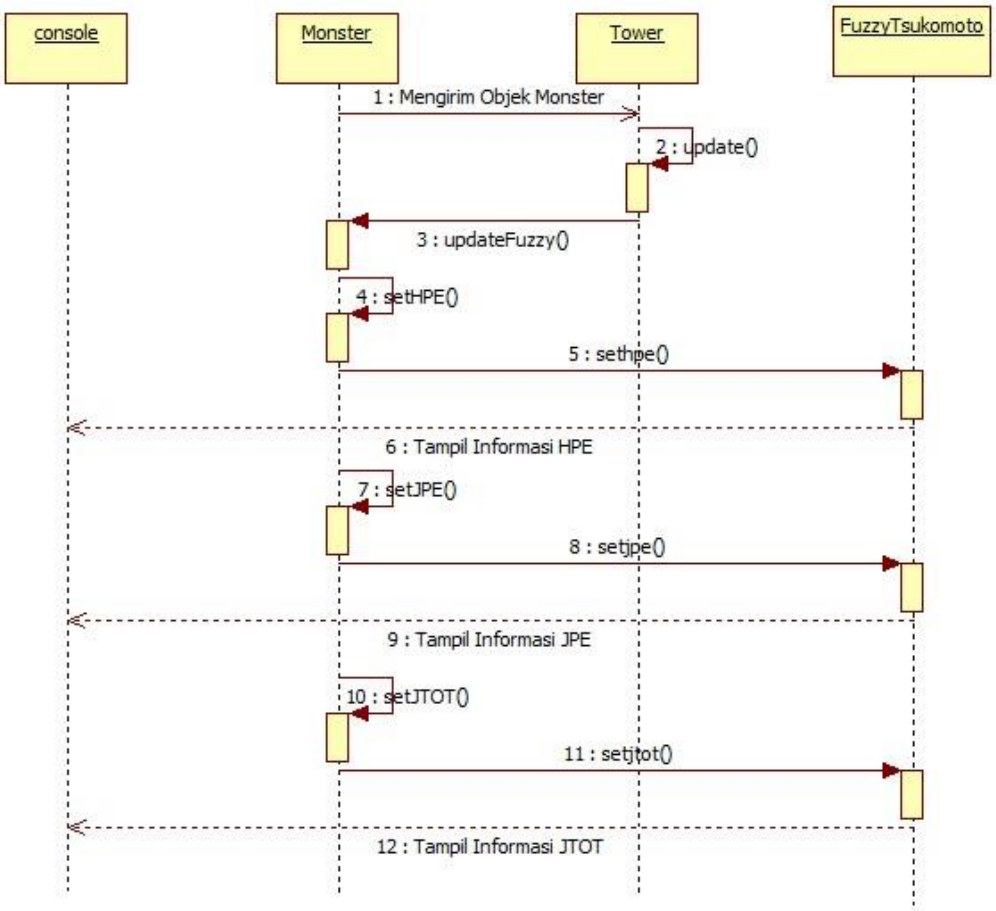


Gambar 3.18 Sequence diagram Menghitung Jarak

e) Sequence Diagram Mengupdate Parameter

Gambar 3.14 menggambarkan proses terjadinya update parameter pada console.

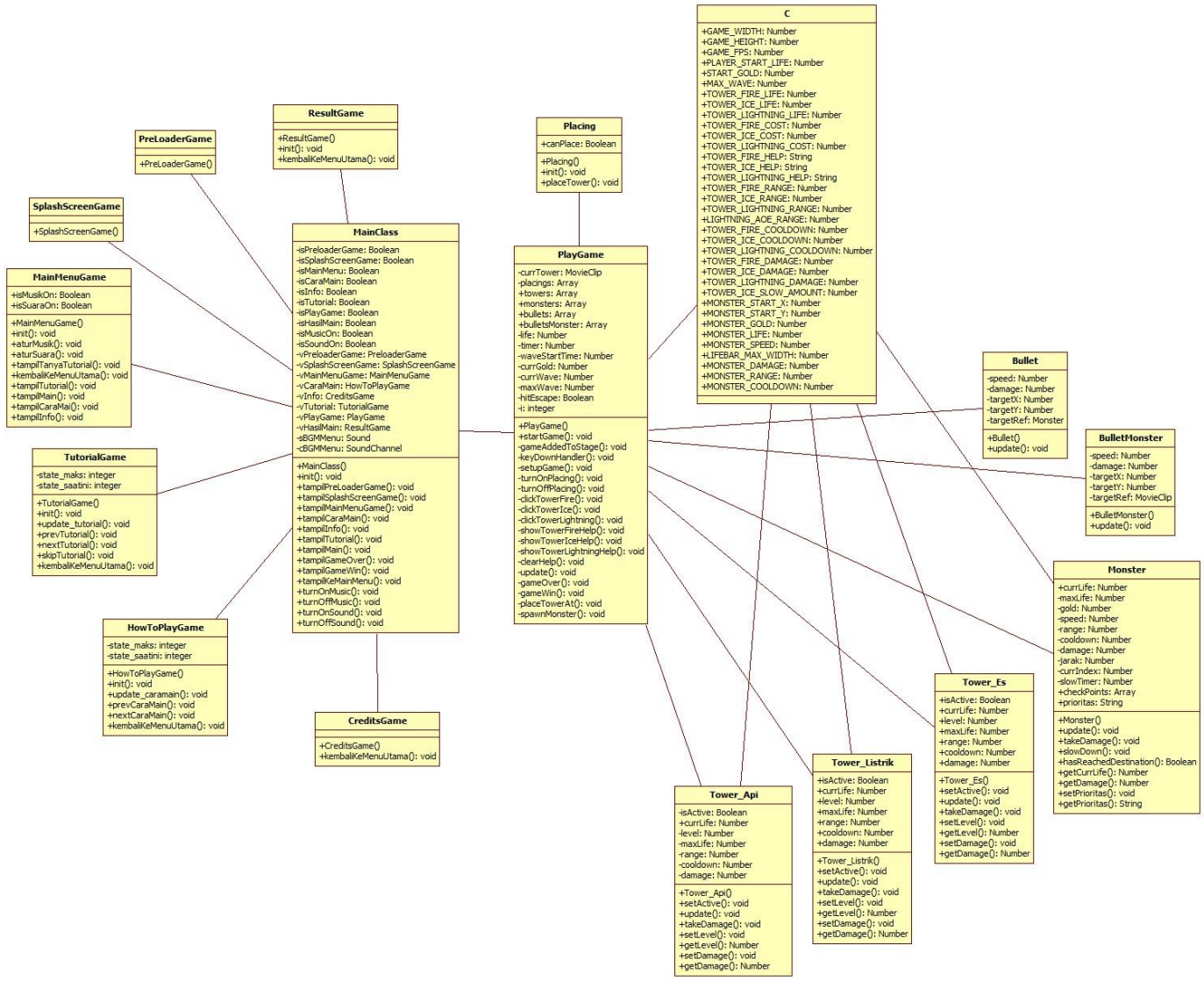
Di kelas monster akan mengirimkan event objek monster terhadapat tower , lalu tower akan melakukan *method* update berupa hpe , dpe dan jarak enemy terhadap tower.



Gambar 3.14 Sequence diagram Mengupdate Parameter

3.5.5. Class Diagram

Class Diagram menggambarkan hubungan antar kelas dalam *game*. Dalam diagram kelas, dibuat kelas-kelas yang saling berkaitan antara satu kelas dengan kelas lainnya. Penentuan jumlah kelas dalam *class diagram* berdasarkan analisis terhadap *use case* yang telah dibuat sebelumnya dimana, dalam *game* setiap antarmuka terbungkus dalam satu kelas.

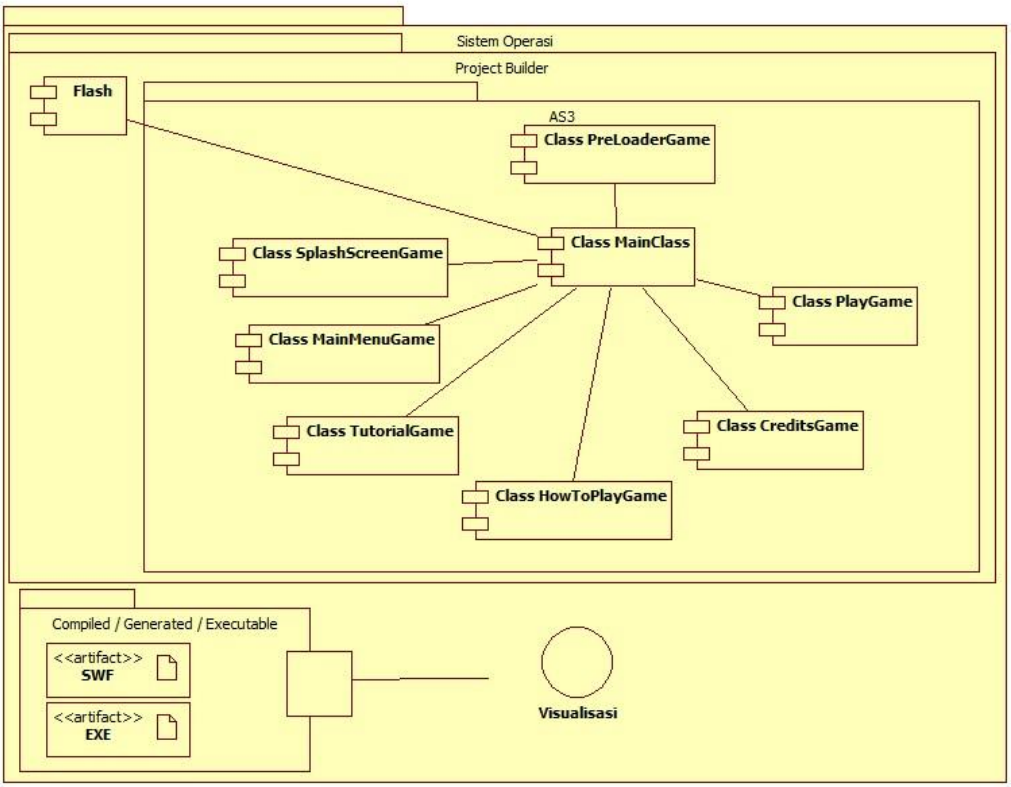


Gambar 3.20 Class diagram Game Benteng Bertahan

Selain itu, kelas-kelas lainnya berdasarkan kebutuhan dari *gameplay* utama permainan ini, seperti jenis-jenis tower dan *enemy* kelasnya masing-masing. Secara lengkap, *class diagram* digambarkan seperti pada Gambar 3.20.

3.5.6. Component Diagram

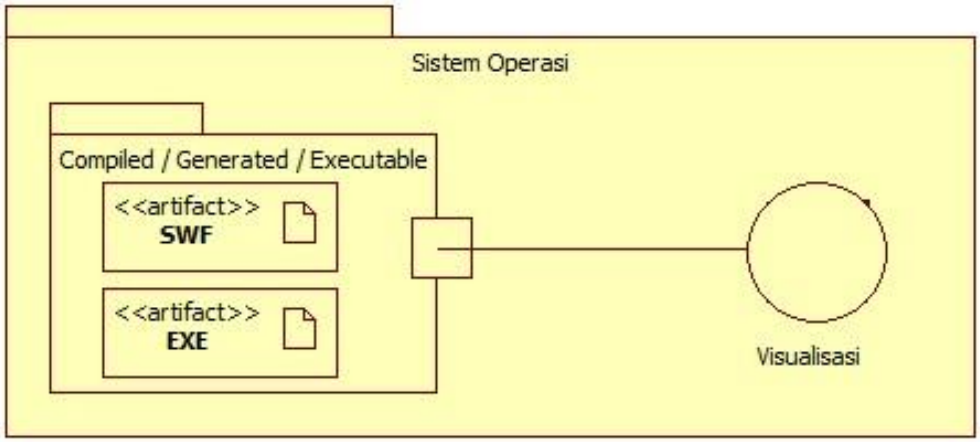
Component Diagram menggambarkan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem. Pada penelitian ini, IDE (*Integrated Development Environtment*) yang digunakan adalah *Adobe Flash CS6* dengan bahasa pemrograman *Actionscript 3.0*. Gambar 3.21 memperlihatkan bagaimana kelas utama *MainClass* yang dipanggil oleh IDE *Flash*. Sementara objek-objek lain diturunkan dari kelasnya masing-masing di kelas utama tersebut. Setelah semua kode program dijalankan, maka akan menghasil sebuah *executable* file yang menjadi aplikasi utama berupa visual terhadap *player*.



Gambar 3.21 Component diagram Game Benteng Bertahan

3.5.7. Deployment Diagram

Deployment diagram menggambarkan detail bagaimana komponen dibuat dalam infrastruktur sistem. Pada dasarnya kompilasi *actionscript* sangatlah sederhana, yaitu dapat menghasilkan satu *file executable .swf* ataupun *.exe*, yang mana keduanya telah memiliki *interface* untuk langsung berkomunikasi dengan pengguna. Gambar 3.22 merupakan gambar *component diagram* untuk *Game Benteng Bertahan.*



Gambar 3.22 Deployment diagram

3.6 Perancangan Antarmuka

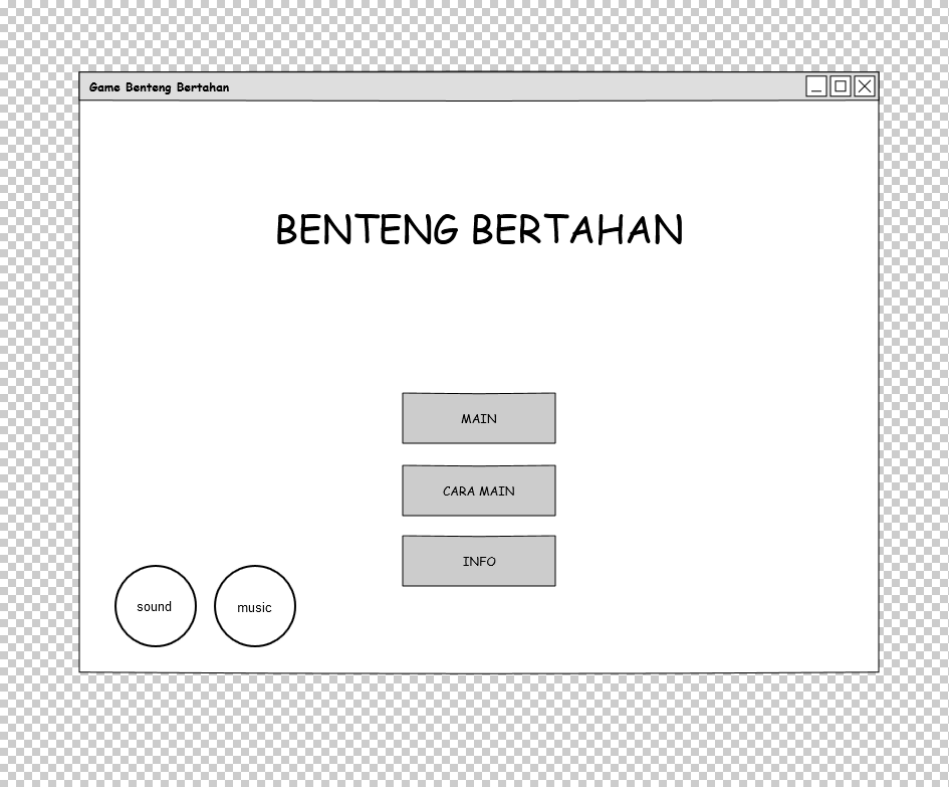
Antarmuka merupakan sarana komunikasi langsung antara pengguna dengan sistem. Oleh karena itu, antarmuka menjadi sangat penting pada sebuah sistem berbasis *visual*.

3.6.1 Perancangan Antarmuka Awal

Gambar 3.23 merupakan rancangan untuk antarmuka awal dimana terdapat tiga buah button yaitu *Button Main , Button Cara Main ,* dan *Button Info. Button Main* berfungsi untuk masuk ke tampilan PreMain, *Button Cara Main* berfungsi

untuk masuk ke tampilan bantuan sedangkan *Button Info* merupakan informasi tentang pembuat.

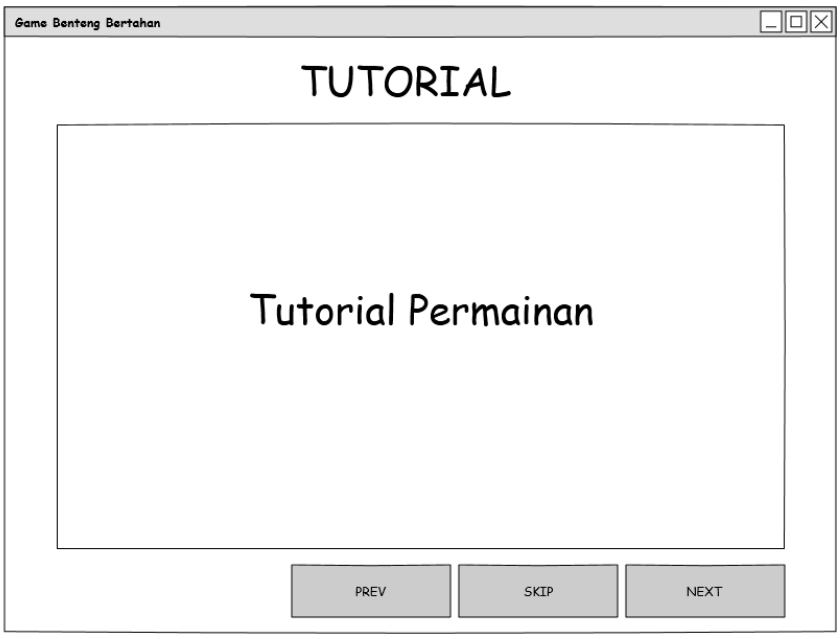
Pada sebelah kiri layar juga terdapat *Button Sound* dan *Button Music* yang berfungsi untuk mematikan atau menghidupkan sound atau music.



Gambar 3.23 Perancangan Antarmuka Awal

3.6.2 Perancangan Antarmuka Tutorial

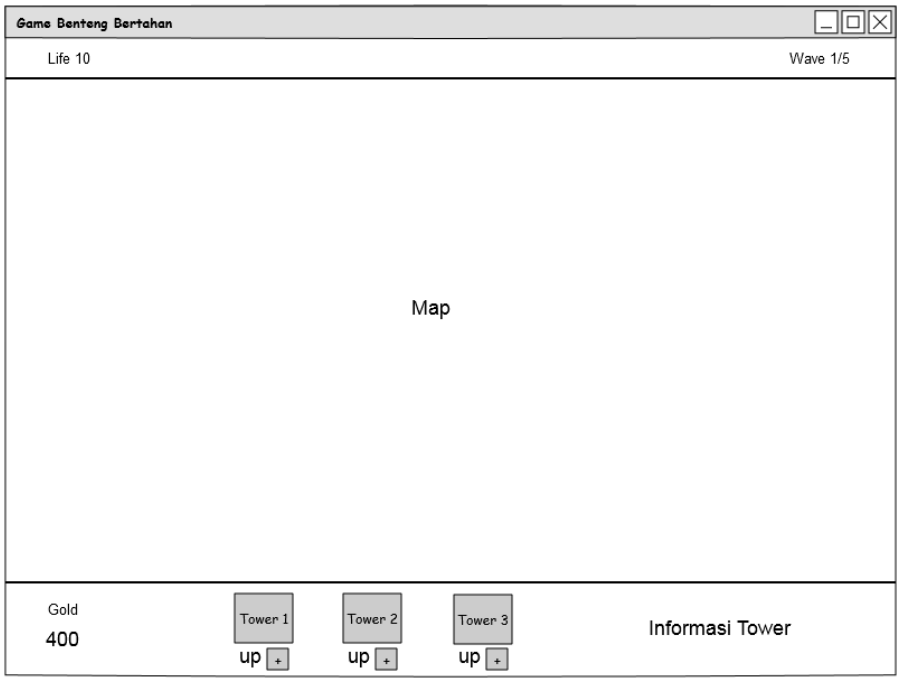
Gambar 3.24 merupakan rancangan antarmuka ketika pemain telah menekan *Button Main* , pemain akan memasuki rancangan antarmuka tutorial menjelaskan secara singkat komponen di dalam gameplay dari cara meletakan tower , melakukan upgrade terhadap tower , jumlah wave *enemy* dan penjelasan tentang tower – tower yang ada .



3.24 Perancangan Antarmuka Tutorial

3.6.3 Perancangan Antarmuka Permainan

Gambar 3.25 merupakan rancangan antarmuka permainan. Pada sudut kiri atas terdapat jumlah *life point* pemain. Disini terdapat map yang merupakan tempat arena pemain dapat meletakan tower – tower yang ada . Terdapat 3 jenis *Button Tower* yaitu tower api , tower ice dan tower listrik yang dapat di upgrade untuk memperkuat pertahanan *player* .



3.25 Perancangan Antarmuka Permainan

3.6.4 Perancangan Antarmuka Menang

Pada gambar 3.26 adalah tampilan antarmuka apabila pemain memenangkan permainan . Pemain akan menang apabila berhasil menghalau *enemy* dalam memasukin goal point selama waktu yang di tentukan.



3.26 Perancangan Antarmuka Menang

3.6.5 Perancangan Antarmuka Kalah

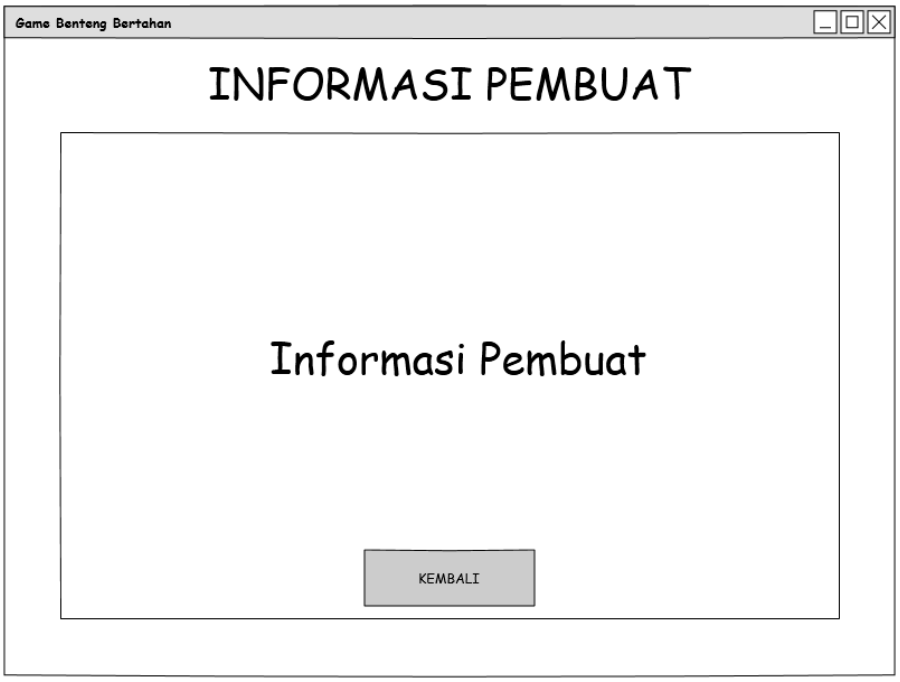
Pada gambar 3.27 adalah tampilan antarmuka apabila kalah dalam permainan. Pemain akan kalah apabila tidak berhasil menghalau *enemy* dalam memasukin goal point selama waktu yang di tentukan.



3.27 Perancangan Antarmuka Kalah

3.6.6 Perancangan Antarmuka Informasi Pembuat

Gambar 3.28 merupakan rancangan untuk tampilan Informasi Pembuat dimana informasi yang ditampilkan berupa pembuat *game*, dosen pembimbing, ucapan terimakasih dan beberapa sumber-sumber yang digunakan dalam *game*. Untuk kembali ke menu sebelumnya dapat menggunakan *Button Kembali*.

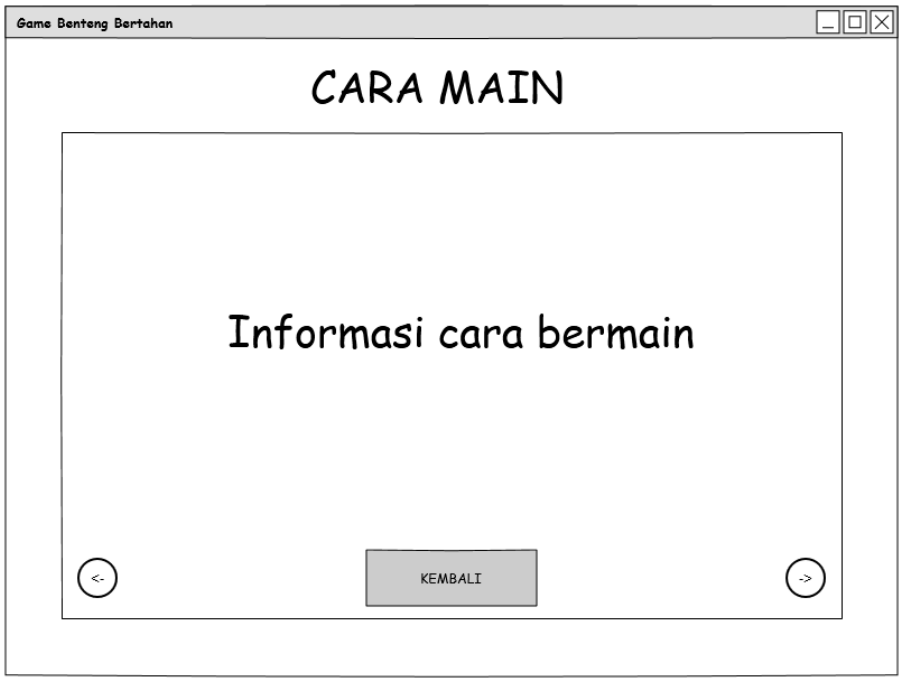


3.28 Perancangan Antarmuka Informasi Pembuat

3.6.7 Perancangan Antarmuka Cara Main

Gambar 3.29 merupakan perancangan antarmuka untuk Cara Bermain. Dalam menu ini berisi mengenai cara-cara bermain seperti apa yang harus dilakukan saat permainan berlangsung hingga kontrol permainan. Pemain hanya perlu menekan tombol kiri mouse di area informasi cara bermain untuk berpindah dari satu cara bermain ke cara bermain selanjutnya.

Untuk kembali ke menu sebelumnya, dapat menggunakan *Button Kembali* yang berada di bagian bawah informasi dari cara bermain.



3.29 Perancangan Antarmuka Cara Main