BAB 4

IMPLEMENTASI DAN PENGUJIAN

Pada bab ini akan dilakukan implementasi dan pengujian terhadap aplikasi permainan hasil pengembangan yaitu ArkAId XIV. Tahapan ini dilakukan setelah tahap perancangan selesai dilakukan. Adapun hasil dari perancangan tersebut kemudian diimplementasikan pada bahasa pemrograman. Setelah implementasi, maka dilakukan pengujian terhadap aplikasi hasil pengembangan dan akan dilihat kekurangan-kekurangannya untuk pengembangan aplikasi tersebut selanjutnya. Pengujian tersebut terdiri dari pengujian alpha dan beta.

4.1 Implementasi

Tahap implementasi merupakan tahap penciptaan perangkat lunak dan juga tahap kelanjutan dari kegiatan perancangan aplikasi. Tahap ini merupakan tahap dimana aplikasi siap untuk dioperasikan, yang terdiri dari penjelasan mengenai lingkungan implementasi, dan implementasi program.

4.1.1 Implementasi Perangkat Keras

Perangkat keras yang akan digunakan untuk menjalankan aplikasi permainan ArkAId XIV ini tidak harus komputer yang berspesifikasi tinggi. Hal itu disebabkan oleh penggunaan sumber daya yang dioptimalkan pada CPU, walaupun aplikasi ini sebenarnya mengimplementasikan *multithreading*. Selain itu, aplikasi ini tidak membutuhkan kartu grafis kelas atas karena hanya memakai

sumberdaya grafis yang tidak terlalu konsumtif. Sebagai ilustrasi sistem minimum yang dapat menjalankan aplikasi adalah sebagai berikut:

1. *Processor* dengan kecepatan ~ 1GHz
2. RAM dengan kapasitas 1GB
3. VGA 32MB *onboard*
4. *Harddisk* 80 GB
5. Monitor dengan resolusi 1024 X 768
6. *Mouse*
7. *Keyboard*

4.1.2 Implementasi Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan untuk menjalankan aplikasi permainan ArkAId XIV ini adalah sebagai berikut :

1. Sistem Operasi berbasis 32 bit
2. JVM (Java Virtual Machine)
3. Perambah Internet

4.1.3 Implementasi Aplikasi

ArkAId XIV adalah aplikasi permainan berjenis *arcade-shooter* yang merupakan merupakan pengembangan dari aplikasi *arcade-shooter* konvensional. Oleh karena itu alur permainannya mirip dengan aplikasi permainan konvensional. Pengguna aplikasi berperan sebagai pemain yang mengendalikan karakter berupa pesawat yang terjebak di area rawan suatu ruang angkasa. Area tersebut dikatakan rawan karena disana sedang terjadi badai asteroid. Selain itu di area ini juga terdapat banyak *alien* yang sedang berpatroli dan mereka siap

menembak siapa pun yang melanggar garis kedaulatan wilayahnya. Pemain pun tentunya harus dapat menghindari asteroid yang berterbangan beserta *alien-alien* yang menjadi musuhnya itu untuk dapat selamat.

Aplikasi ini berbasis Java-Applet yang dibangun dengan menggunakan bahasa pemrograman JAVA. Oleh karena itu, aplikasi permainan ini dapat dijalankan di *platform* apapun yang telah dipasangi JVM (Java Virtual Machine) tentunya. Untuk menjalankan aplikasi ini tidak perlu dilakukan instalasi, cukup dengan membukanya melalui perambah internet atau menggunakan JDK appletviewer.exe. Apabila JVM belum terpasang, maka aplikasi ini pun tak akan bisa dijalankan.

Berikut ini merupakan fungsionalitas pada aplikasi permainan ArkAId XIV beserta sejumlah *file* yang merupakan komponen pengimplementasiannya:

Tabel 4.1 Tabel Implementasi Aplikasi

Nama Fungsionalitas	Nama *File*
Menampilkan Menu Utama	ArkAIdXIV.html
	Game.Java
	ArkAIdXIV.java
Mulai Permainan Baru	Game.Java
	ArkAIdXIV.java
	Point2D.java
	BaseGameEntity.java
	ImageEntity.java
	Sprite.java
	AnimateSprite.java
	MidiSequence.java
	SoundClip.java
Lihat Petunjuk	Game.java
	ArkAIdXIV.java
Lihat Kredit	Game.java
	ArkAIdXIV.java

4.1.4 Implementasi Antarmuka

Antarmuka aplikasi ini terdiri dari beberapa bagian seperti yang akan dijelaskan pada sub bab ini, sedangkan sisanya bisa dilihat di lampiran antarmuka.

Saat pengguna membuka aplikasi pertama kalinya, maka antarmuka yang ditampilkan oleh sistem adalah antarmuka menu utama seperti pada gambar 4.1:



Gambar 4.1 Tampilan Menu Utama ArkAId XIV

Jika pengguna ingin langsung bermain, cukup dengan menekan tombol enter untuk memilih mulai permainan baru, dan akan tampilannya seperti yang terlihat pada gambar 4.2:



Gambar 4.2 Permainan ArkAId XIV

Adapun interaksi antara pengguna dengan karakter pesawat seperti mengontrol pergerakan pesawat, menembak musuh, dan menghampiri power up dapat dilihat pada gambar 4.3:



Gambar 4.3 Interaksi Pemain dengan aplikasi ArkAId XIV

Perubahan *state* entitas permainan pada aplikasi ArkAId XIV dapat dilihat pada gambar 4.4:



Gambar 4.4 Reaksi perubahan *State* Aplikasi ArkAId XIV

4.2 Pengujian

Pengujian yang akan digunakan dalam menguji aplikasi ArkAId XIV ini adalah dengan menggunakan metode pengujian *white box.* Pengujian *white box* merupakan pengujian yang didasarkan pada pengecekan terhadap detail perancangan, menggunakan struktur kontrol dari desain aplikasi secara procedural untuk membagi pengujian ke dalam beberapa kasus pengujian. Penentuan kasus uji disesuaikan dengan struktur sistem, pengetahuan mengenai aplikasi digunakan untuk mengidentifikasikan kasus uji tambahan.

Tujuan penggunaan *white box* adalah untuk menguji semua *statement* program. Penggunaan metode pengujian *white box* dilakukan untuk:

1. Memberikan jaminan bahwa semua *independent paths* suatu modul dikerjakan minimal satu kali.
2. Semua keputusan logis untuk kondisi *true* atau *false* pada aplikasi telah dikerjakan.
3. Seluruh kondisi perulangan pada aplikasi telah dikerjakan berdasarkan pada batasan nilai tertentu.
4. Memberikan jaminan bahwa struktur data internal telah digunakan untuk menjamin validitas jalur keputusan.

Apabila setiap *independent paths* telah dikerjakan, maka aplikasi yang bersangkutan telah benar. Tetapi jika terdapat satu saja *node* pada suatu *independent paths* yang tidak dikerjakan, maka masih terdapat kesalahan pada aplikasi tersebut.

4.2.1 Pengujian Alpha

Rencana pengujian yang akan dilakukan dengan menguji sistem menggunakan metode *white box* untuk mengetahui apakah setiap *node* pada seluruh *independent paths* dikerjakan. Serta untuk menguji alur logika, perulangan, dan validitas parameter masukannya. Berikut ini tabel rencana pengujian dari aplikasi ArkAId XIV:

Tabel 4.2 Rencana Pengujian Aplikasi ArkAId XIV

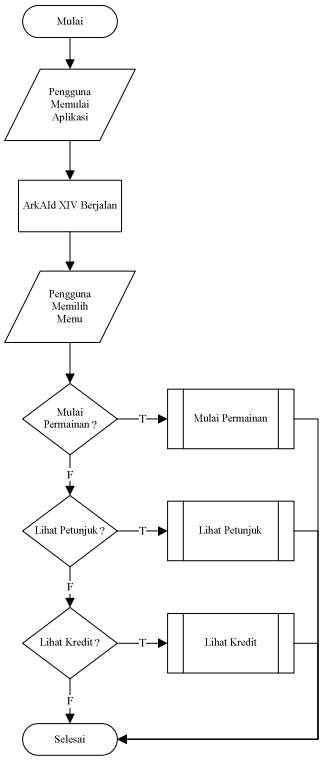
Kelas Uji	Kasus Uji	Jenis Pengujian
Menu Permainan	Reaksi Aplikasi terhadap Perintah Masukkan dari Pengguna	*White Box*
Interaksi Pengguna	Reaksi Entitas Pesawat terhadap Perintah Masukkan dari Pengguna	*White Box*

Alur Permainan	Reaksi Keseluruhan Entitas yang Terlibat serta Pengaruhnya Terhadap Alur Permainan	*White Box*
Collision Detection	Pendeteksian Tabrakan pada Entitas Pesawat terhadap Entitas Lain	*White Box*
	Pendeteksian Tabrakan pada Entitas Peluru terhadap Entitas Lain	*White Box*
	Pendeteksian Tabrakan pada Entitas Asteroid terhadap Entitas Lain	*White Box*
	Pendeteksian Tabrakan pada Entitas Musuh terhadap Entitas Lain	*White Box*
	Pendeteksian Tabrakan pada Entitas Power up terhadap Entitas Lain	*White Box*
Finite State Machine	Perubahan *State* Entitas Asteroid (FSM Spawned)	*White Box*
	Perubahan *State* Akibat Pengaruh Power Up (FSM Power Up Effect)	*White Box*
	Perubahan *State* Entitas Peluru (FSM Bullet Upgrade)	*White Box*
	Perubahan *State* Entitas Pesawat (FSM Shielded)	*White Box*

4.2.1.1 Menu Permainan

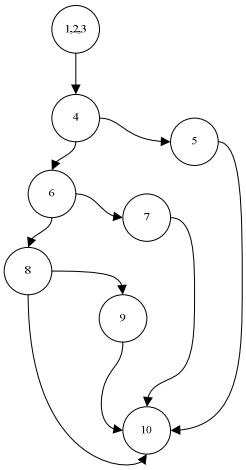
Pengujian Menu permainan dilakukan untuk melihat apakah aplikasi akan menampilkan menu yang sesuai dengan yang diinginkan pengguna.

a. *Flow Chart*



Gambar 4.5 *Flow Chart* Pengujian Menu Permainan

b. *Flow Graph*



Gambar 4.6 *Flow Graph* Pengujian Menu Permainan

c. *Cyclomatic Complexity* V(G)

E = 10, N = 8

V(G) = E – N + 2 = 10 – 8 + 2 = 2 + 2 = 4

d. *Independent Paths*

Path 1 = 1,2,3 – 4 – 5 – 10

Path 2 = 1,2,3 – 4 – 6 – 7 – 10

Path 3 = 1,2,3 – 4 – 6 – 8 – 9 – 10

Path 4 = 1,2,3 – 4 – 6 – 8 – 10

e. Hasil Uji

Tabel 4.3 Hasil Pengujian *White Box* Kelas Uji Menu Permainan

Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1,2,3, (void) 4, (true) 5, (void) 10(void)	Permainan Dimulai	Permainan Dimulai	*Path* Dikerjakan
2	1,2,3, (void) 4, (false) 6, (true) 7, (void) 10(void)	Menampilkan Menu Lihat Petunjuk	Menampilkan Menu Lihat Petunjuk	*Path* Dikerjakan
3	1,2,3, (void) 4, (false) 6, (false) 8, (true) 9, (void) 10(void)	Menampilkan Menu Lihat Kredit	Menampilkan Menu Lihat Kredit	*Path* Dikerjakan
4	1,2,3, (void) 4, (false) 6, (false) 8, (false) 10(void)	Menampilkan Menu Utama	Menampilkan Menu Utama	*Path* Dikerjakan

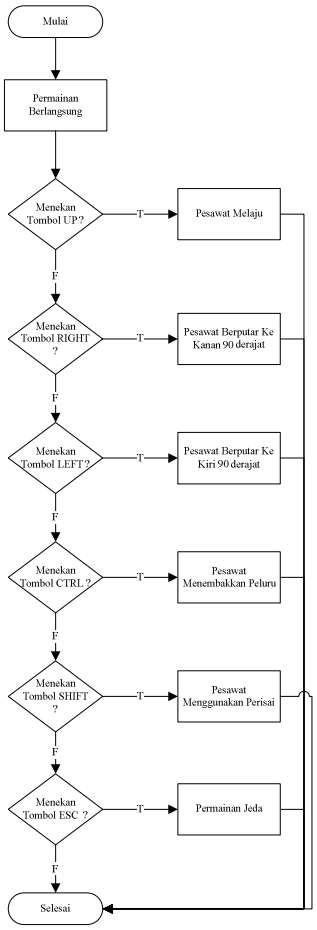
f. Kesimpulan

Dari hasil pengujian menu permainan yang dilakukan, didapat kesimpulan bahwa seluruh *node* yang ada pada setiap *independent path* telah dikerjakan, serta tidak terjadi pengulangan tak berhingga.

4.2.1.2 Interaksi Pengguna

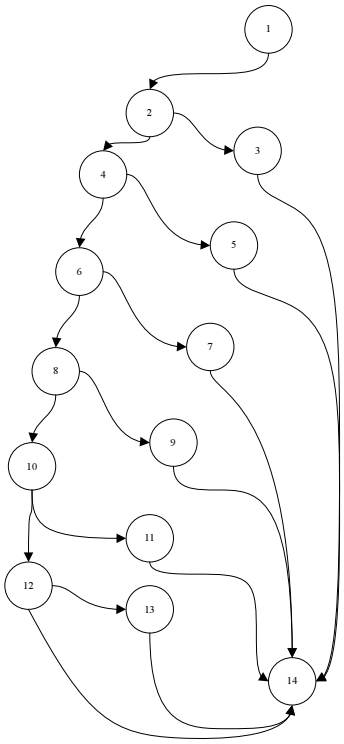
Pengujian interaksi pengguna dilakukan untuk melihat apakah aplikasi menjalankan perintah sesuai yang dimasukkan oleh pengguna. Hal tersebut dapat dilhat pada reaksi entitas pesawat pada arena permainan.

a. *Flow Chart*



Gambar 4.7 *Flow Chart* Pengujian Interaksi Pengguna

b. *Flow Graph*



Gambar 4.8 *Flow Graph* Pengujian Interaksi Pengguna

c. *Cyclomatic Complexity* V(G)

E = 19, N = 14

V(G) = E – N + 2 = 19 – 14 + 2 = 5 + 2 = 7

d. *Independent Paths*

Path 1 = 1 – 2 – 3 – 14

Path 2 = 1 – 2 – 4 – 5 – 14

Path 3 = 1 – 2 – 4 – 6 – 7 – 14

Path 4 = 1 – 2 – 4 – 6 – 8 – 9 – 14

Path 5 = 1 – 2 – 4 – 6 – 8 – 10 – 11 – 14

Path 6 = 1 – 2 – 4 – 6 – 8 – 10 – 12 – 13 – 14

Path 7 = 1 – 2 – 4 – 6 – 8 – 10 – 12 – 14

e. Hasil Uji

Tabel 4.4 Hasil Pengujian *White Box* Kelas Uji Interaksi Pengguna

Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1, (void) 2, (true) 3, (void) 14, (void)	Pesawat Melaju	Pesawat Melaju	*Path* Dikerjakan
2	1, (void) 2, (false) 4, (true) 5, (void) 14, (void)	Pesawat Berputar Ke Kanan	Pesawat Berputar Ke Kanan	*Path* Dikerjakan
3	1, (void) 2, (false) 4, (false) 6, (true) 7, (void) 14, (void)	Pesawat Berputar Ke Kiri	Pesawat Berputar Ke Kiri	*Path* Dikerjakan
4	1, (void) 2, (false) 4, (false) 6, (false) 8, (true) 9, (void) 14, (void)	Pesawat Menembakkan Peluru	Pesawat Menembakkan Peluru	*Path* Dikerjakan
5	1, (void) 2, (false) 4, (false) 6, (false) 8, (false) 10, (true) 11, (void) 14, (void)	Pesawat Menggunakan Perisai	Pesawat Menggunakan Perisai	*Path* Dikerjakan
6	1, (void) 2, (false) 4, (false) 6, (false) 8, (false) 10, (false)	Permainan Jeda	Permainan Jeda	*Path* Dikerjakan

	12, (true) 13, (void) 14, (void)			
7	1, (void) 2, (false) 4, (false) 6, (false) 8, (false) 10, (false) 12, (false) 14, (void)	Menunggu Perintah dari Pengguna	Menunggu Perintah dari Pengguna	*Path* Dikerjakan

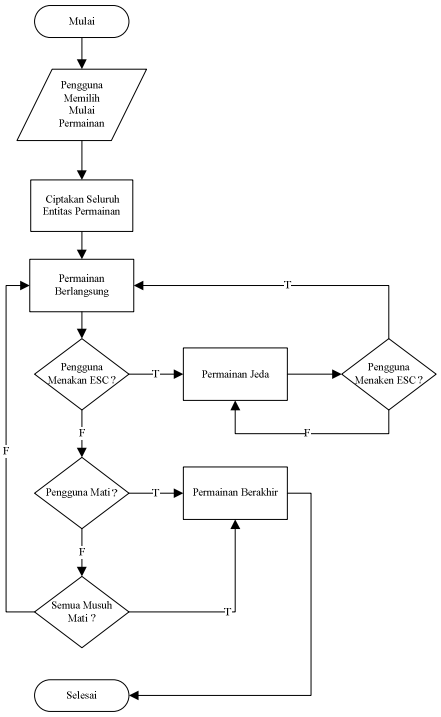
f. Kesimpulan

Dari hasil pengujian interaksi pengguna yang dilakukan, didapat kesimpulan bahwa seluruh *node* yang ada pada setiap *independent path* telah dikerjakan, serta tidak terjadi pengulangan tak berhingga.

4.2.1.3 Alur Permainan

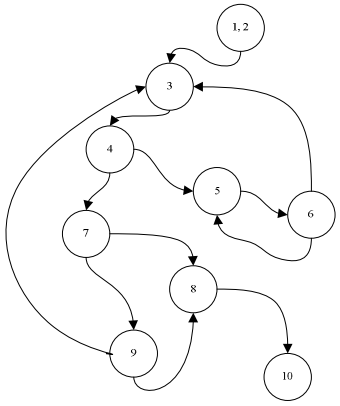
Pengujian alur permainan dilakukan untuk melihat apakah alur permainan pada aplikasi telah berjalan sesuai dengan yang diharapkan.

a. *Flow Chart*



Gambar 4.9 *Flow Chart* Pengujian Alur Permainan

b. *Flow Graph*



Gambar 4.10 *Flow Graph* Pengujian Alur Permainan

c. *Cyclomatic Complexity* V(G)

E = 12, N = 9

V(G) = E – N + 2 = 12 – 9 + 2 = 3 + 2 = 5

d. *Independent Paths*

Path 1 = 1,2 – 3 – 4 – 5 – 6 – 5

Path 2 = 1,2 – 3 – 4 – 5 – 6 – 3

Path 3 = 1,2 – 3 – 4 – 7 – 8 – 10

Path 4 = 1,2 – 3 – 4 – 7 – 9 – 8 – 10

Path 5 = 1,2 – 3 – 4 – 7 – 9 – 3

e. Hasil Uji

Tabel 4.5 Hasil Pengujian *White Box* Kasus Uji Alur Permainan

Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1, 2, (void) 3, (void) 4, (true) 5, (true) 6, (true) 5, (conditional)	Permainan Jeda (Paused)	Permainan Jeda (Paused)	*Path* Dikerjakan
2	1, 2, (void) 3, (void) 4, (true) 5, (true) 6, (false) 3, (conditional)	Permainan Berlangsung	Permainan Berlangsung	*Path* Dikerjakan
3	1, 2, (void) 3, (false) 4, (false) 7, (true) 8, (void) 10, (void)	Permainan Berakhir	Permainan Berakhir	*Path* Dikerjakan
4	1, 2, (void) 3, (false) 4, (false) 7, (true)	Permainan Berakhir	Permainan Berakhir	*Path* Dikerjakan

	9, (true) 8, (void) 10, (void)			
5	1, 2, (void) 3, (false) 4, (false) 7, (true) 9, (true) 3, (conditional)	Permainan Berlangsung	Permainan Berlangsung	*Path* Dikerjakan

f. Kesimpulan

Dari hasil pengujian alur permainan yang dilakukan, didapat kesimpulan bahwa seluruh *node* yang ada pada setiap *independent path* telah dikerjakan, serta tidak terjadi pengulangan tak berhingga.

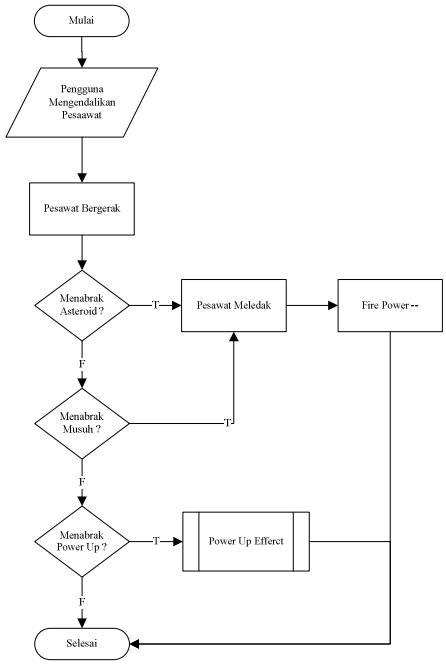
4.2.1.4 Collision Detection

Pengujian *collision detection* dilakukan untuk melihat apakah metode *bounding rectangle collision detection* yang diterapkan sudah berjalan sesuai dengan yang diharapkan. Berikut ini merupakan pengujian *collision detection* yang yang dilakukan terhadap setiap entitas permainan:

1. Pesawat

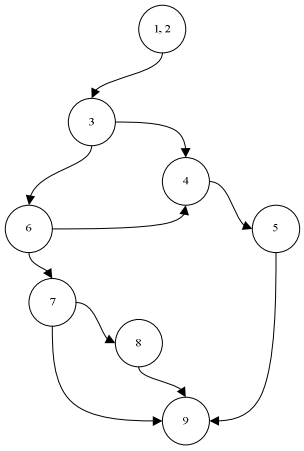
Menguji apakah *collision* pada entitas pesawat terdeteksi ketika *bounding recrtangle*-nya bertabrakan dengan *bounding recatangle* entitas lain, serta pengaruh-pengaruhnya terhadap alur permainan.

a. *Flow Chart*



Gambar 4.11 *Flow Chart* Pengujian *Collision Detection* Entitas Pesawat

b. *Flow Graph*



Gambar 4.12 *Flow Graph* Pengujian *Collision Detection* Entitas Pesawat

c. *Cyclomatic Complexity* V(G)

E = 9, N = 7

V(G) = E – N + 2 = 9 – 7 + 2 = 2 + 2 = 4

d. *Independent Paths*

Path 1 = 1,2 – 3 – 4 – 5 – 9

Path 2 = 1,2 – 3 – 6 – 4 – 5 – 9

Path 3 = 1,2 – 3 – 6 – 7 – 8 – 9

Path 4 = 1,2 – 3 – 6 – 7 – 9

e. Hasil Uji

Tabel 4.6 Hasil Pengujian *White Box Collision Detection* Entitas Pesawat

Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1, 2, (void) 3, (true) 4, (void) 5, (void) 9, (void)	Pesawat Meldak, Fire Power Berkurang	Pesawat Meldak, Fire Power Berkurang	*Path* Dikerjakan
2	1, 2, (void) 3, (false) 6, (true) 4, (void) 5, (void) 9, (void)	Pesawat Meldak, Fire Power Berkurang	Pesawat Meldak, Fire Power Berkurang	*Path* Dikerjakan
3	1, 2, (void) 3, (false) 6, (false) 7, (true) 8, (void) 9, (void)	FSM Power Up Effect Terpicu	FSM Power Up Effect Terpicu	*Path* Dikerjakan
4	1, 2, (void) 3, (false) 6, (false) 7, (false) 9, (void)	*Collision* tidak terdeteksi	*Collision* tidak terdeteksi	*Path* Dikerjakan

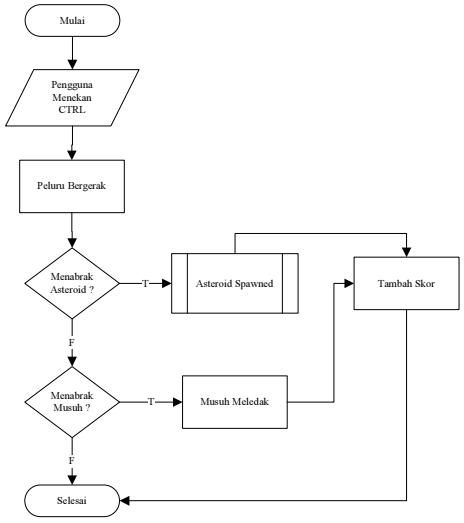
f. Kesimpulan

Dari hasil pengujian *collision detection* entitas pesawat yang dilakukan, didapat kesimpulan bahwa seluruh *node* yang ada pada setiap *independent path* telah dikerjakan, serta tidak terjadi pengulangan tak berhingga.

2. Peluru

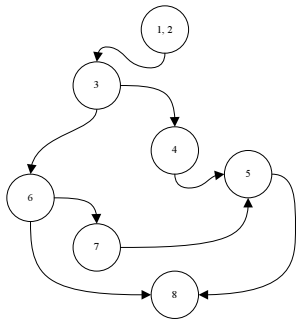
Menguji apakah *collision* pada entitas peluru terdeteksi ketika *bounding recrtangle*-nya bertabrakan dengan *bounding recatangle* entitas lain, serta pengaruh-pengaruhnya terhadap alur permainan.

a. *Flow Chart*



Gambar 4.13 *Flow Chart* Pengujian *Collision Detection* Entitas Peluru

b. *Flow Graph*



Gambar 4.14 *Flow Graph* Pengujian *Collision Detection* Entitas Peluru

c. *Cyclomatic Complexity* V(G)

E = 8, N = 7

V(G) = E – N + 2 = 8 – 7 + 2 = 1 + 2 = 3

d. *Independent Paths*

Path 1 = 1,2 – 3 – 4 – 5 – 8

Path 2 = 1,2 – 3 – 6 – 7 – 5 – 8

Path 3 = 1,2 – 3 – 6 – 8

e. Hasil Uji

Tabel 4.7 Hasil Pengujian *White Box Collision Detection* Entitas Peluru

Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1, 2, (void) 3, (true) 4, (void) 5, (void) 8, (void)	Skor Bertambah, FSM Spawned Asteroid Terpicu	Skor Bertambah, FSM Spawned Asteroid Terpicu	*Path* Dikerjakan
2	1, 2, (void) 3, (false) 6, (true) 7, (void) 5, (void)	Skor Bertambah, Musuh Meledak	Skor Bertambah, Musuh Meledak	*Path* Dikerjakan

	8, (void)			
3	1, 2, (void) 3, (false) 6, (false) 8, (void)	*Collision* tidak terdeteksi	*Collision* tidak terdeteksi	*Path* Dikerjakan

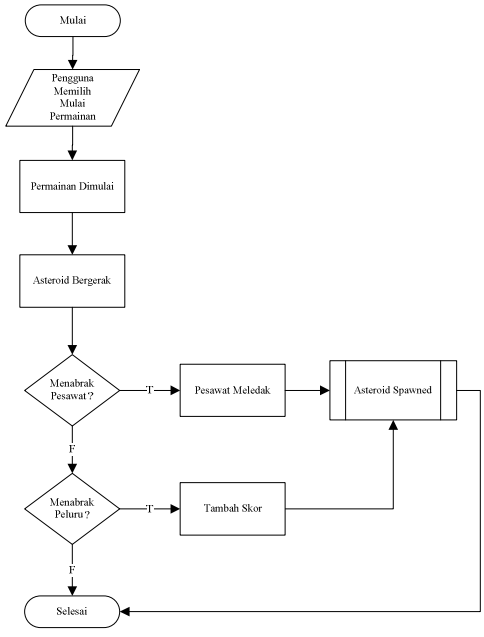
f. Kesimpulan

Dari hasil pengujian *collision detection* entitas peluru yang dilakukan, didapat kesimpulan bahwa seluruh *node* yang ada pada setiap *independent path* telah dikerjakan, serta tidak terjadi pengulangan tak berhingga.

3. Asteroid

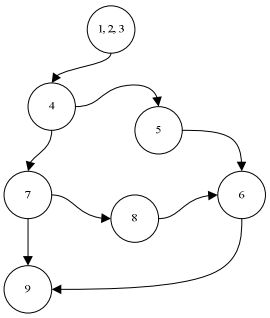
Menguji apakah *collision* pada entitas Asteroid terdeteksi ketika *bounding recrtangle*-nya bertabrakan dengan *bounding recatangle* entitas lain, serta pengaruh-pengaruhnya terhadap alur permainan.

a. *Flow Chart*



Gambar 4.15 *Flow Chart* Pengujian *Collision Detection* Entitas Asteroid

b. *Flow Graph*



Gambar 4.16 *Flow Graph* Pengujian *Collision Detection* Entitas Asteroid

c. *Cyclomatic Complexity* V(G)

E = 8, N = 7

V(G) = E – N + 2 = 8 – 7 + 2 = 1 + 2 = 3

d. *Independent Paths*

Path 1 = 1,2,3 – 4 – 5 – 6 – 9

Path 2 = 1,2,3 – 4 – 7 – 8 – 6 – 9

Path 3 = 1,2,3 – 4 – 7 – 9

e. Hasil Uji

Tabel 4.8 Hasil Pengujian *White Box Collision Detection* Entitas Asteroid

Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1, 2, 3 (void) 4, (true) 5, (void) 6, (void) 9, (void)	Pesawat Meledak, FSM Spawned Asteroid Terpicu	Pesawat Meledak, FSM Spawned Asteroid Terpicu	*Path* Dikerjakan
2	1, 2, 3 (void) 4, (false) 7, (true) 8, (void) 6, (void) 9, (void)	Asteroid Meledak, FSM Spawned Asteroid Terpicu,	Asteroid Meledak, FSM Spawned Asteroid Terpicu,	*Path* Dikerjakan

		Skor Bertambah	Skor Bertambah	
3	1, 2, 3 (void) 4, (false) 7, (false) 9, (void)	*Collision* tidak terdeteksi	*Collision* tidak terdeteksi	*Path* Dikerjakan

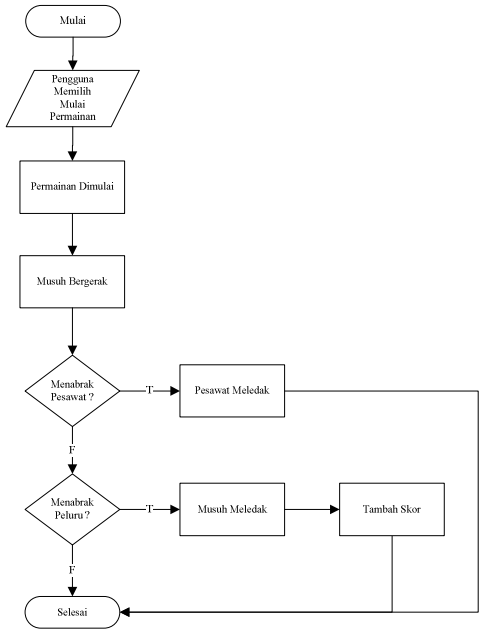
f. Kesimpulan

Dari hasil pengujian *collision detection* entitas asteroid yang dilakukan, didapat kesimpulan bahwa seluruh *node* yang ada pada setiap *independent path* telah dikerjakan, serta tidak terjadi pengulangan tak berhingga.

4. Musuh

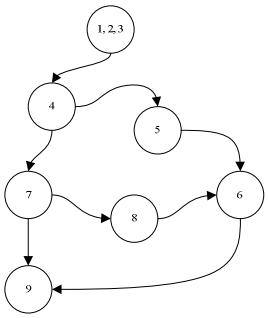
Menguji apakah *collision* pada entitas Musuh terdeteksi ketika *bounding recrtangle*-nya bertabrakan dengan *bounding recatangle* entitas lain, serta pengaruh-pengaruhnya terhadap alur permainan.

a. *Flow Chart*



Gambar 4.17 *Flow Chart* Pengujian *Collision Detection* Entitas Musuh

b. *Flow Graph*



Gambar 4.18 *Flow Graph* Pengujian *Collision Detection* Entitas Musuh

c. *Cyclomatic Complexity* V(G)

E = 8, N = 7

V(G) = E – N + 2 = 8 – 7 + 2 = 1 + 2 = 3

d. *Independent Paths*

Path 1 = 1,2,3 – 4 – 5 – 6 – 9

Path 2 = 1,2,3 – 4 – 7 – 8 – 6 – 9

Path 3 = 1,2,3 – 4 – 7 – 9

e. Hasil Uji

Tabel 4.9 Hasil Pengujian *White Box Collision Detection* Entitas Musuh

Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1, 2, 3 (void) 4, (true) 5, (void) 6, (void) 9, (void)	Pesawat Meledak	Pesawat Meledak	*Path* Dikerjakan
2	1, 2, 3 (void) 4, (false) 7, (true) 8, (void) 6, (void) 9, (void)	Musuh Meledak, Skor Bertambah	Musuh Meledak, Skor Bertambah	*Path* Dikerjakan

3	1, 2, 3 (void) 4, (false) 7, (false) 9, (void)	*Collision* tidak terdeteksi	*Collision* tidak terdeteksi	*Path* Dikerjakan

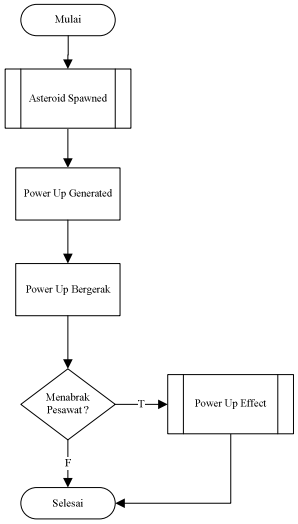
f. Kesimpulan

Dari hasil pengujian *collision detection* entitas musuh yang dilakukan, didapat kesimpulan bahwa seluruh *node* yang ada pada setiap *independent path* telah dikerjakan, serta tidak terjadi pengulangan tak berhingga.

5. Power Up

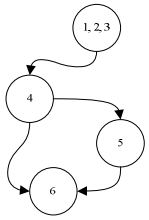
Menguji apakah *collision* pada entitas Power Up terdeteksi ketika *bounding recrtangle*-nya bertabrakan dengan *bounding recatangle* entitas lain, serta pengaruh-pengaruhnya terhadap alur permainan.

a. *Flow Chart*



Gambar 4.19 *Flow Chart* Pengujian *Collision Detection* Entitas Power Up

b. *Flow Graph*



Gambar 4.20 *Flow Graph* Pengujian *Collision Detection* Entitas Power Up

c. *Cyclomatic Complexity* V(G)

E = 4, N = 4

V(G) = E – N + 2 = 4 – 4 + 2 = 0 + 2 = 2

d. *Independent Paths*

Path 1 = 1,2,3 – 4 – 5 – 6

Path 2 = 1,2,3 – 4 – 6

e. Hasil Uji

Tabel 4.10 Hasil Pengujian *White Box Collision Detection* Entitas Power Up

Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1, 2, 3 (void) 4, (true) 5, (void) 6, (void)	FSM Spawned Asteroid Terpicu	FSM Spawned Asteroid Terpicu	*Path* Dikerjakan
2	1, 2, 3 (void) 4, (false) 6, (void)	FSM Spawned Asteroid Terpicu	FSM Spawned Asteroid Terpicu	*Path* Dikerjakan

f. Kesimpulan

Dari hasil pengujian *collision detection* entitas power up yang dilakukan, didapat kesimpulan bahwa seluruh *node* yang ada pada setiap *independent path* telah dikerjakan, serta tidak terjadi pengulangan tak berhingga.

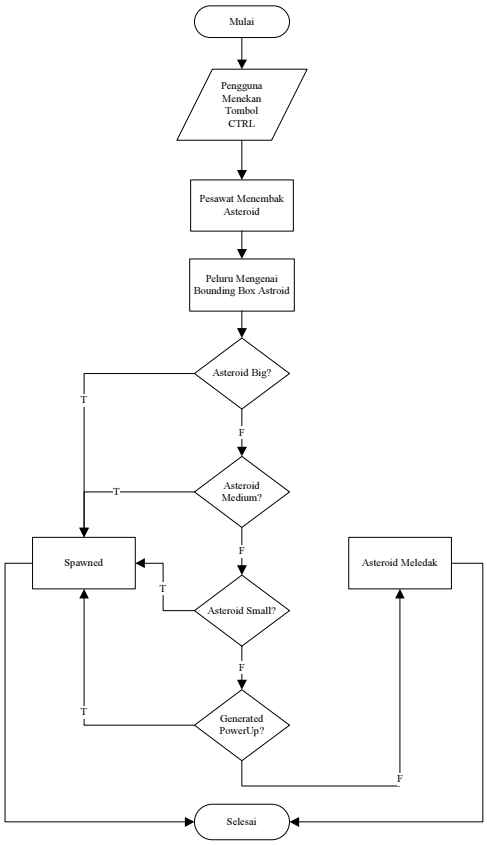
4.2.1.5 Finite State Machine

Pengujian Finite State Machine dilakukan untuk melihat apakah entitas permainan yang menerapkannya telah berubah *state* sesuai yang diharapkan tergantung dari kondisi tertentu. Berikut ini merupakan pengujian terhadap metode FSM yang diterapkan pada aplikasi ArkAId XIV:

1. Asteroid Spawned

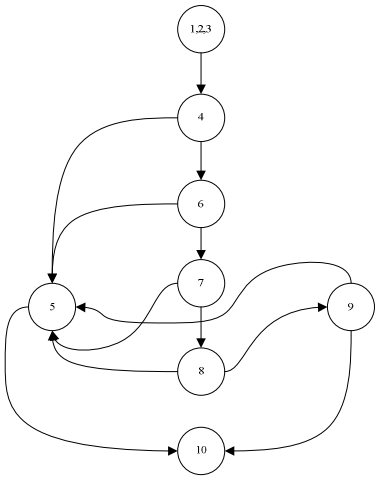
Menguji apakah *state* asteroid akan berubah ketika event pemicunya, yaitu *collision detection*, terjadi.

a. *Flow Chart*



Gambar 4.21 *Flow Chart* Pengujian FSM Asteroid Spawned

b. *Flow Graph*



Gambar 4.22 *Flow Graph* Pengujian FSM Asteroid Spawned

c. *Cyclomatic Complexity* (V(G))

E = 11, N = 8

V(G) = E – N + 2 = 11 – 8 + 2 = 3 + 2 = 5

d. *Independent Paths*

Path 1 = 1,2,3 – 4 – 5 – 10

Path 2 = 1,2,3 – 4 – 6 – 5 – 10

Path 3 = 1,2,3 – 4 – 6 – 7 – 5 – 10

Path 4 = 1,2,3 – 4 – 6 – 7 – 8 – 5 – 10

Path 5 = 1,2,3 – 4 – 6 – 7 – 8 – 9 – 10

e. Hasil Uji

Tabel 4.11 Hasil Pengujian *White Box* FSM Asteroid Spawned

Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1, 2, 3 (void) 4, (true) 5, (void)	Asteroid Big terpecah menjadi 4	Asteroid Big terpecah menjadi 4	*Path* Dikerjakan

	10, (void)	Asteroid Medium	Asteroid Medium	
2	1, 2, 3 (void) 4, (false) 6, (true) 5, (void) 10, (void)	Asteroid Medium terpecah menjadi 4 Asteroid Small	Asteroid Medium terpecah menjadi 4 Asteroid Small	*Path* Dikerjakan
3	1, 2, 3 (void) 4, (false) 6, (false) 7, (true) 5, (void) 10, (void)	Asteroid Small terpecah menjadi 4 Asteroid Tiny	Asteroid Small terpecah menjadi 4 Asteroid Tiny	*Path* Dikerjakan
4	1, 2, 3 (void) 4, (false) 6, (false) 7, (false) 8, (true) 5, (void) 10, (void)	Asteroid Tiny terpecah menjadi Power Up (terjadi secara random)	Asteroid Tiny terpecah menjadi Power Up (terjadi secara random)	*Path* Dikerjakan
5	1, 2, 3 (void) 4, (false) 6, (false) 7, (false) 8, (false) 9, (void) 10, (void)	Asteroid Meledak	Asteroid Meledak	*Path* Dikerjakan

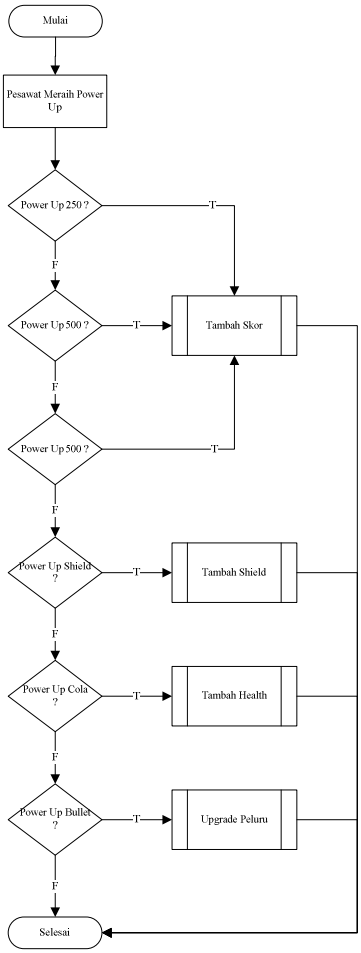
f. Kesimpulan

Dari hasil pengujian FSM spawned *state* yang dilakukan, didapat kesimpulan bahwa seluruh *node* yang ada pada setiap *independent path* telah dikerjakan, serta tidak terjadi pengulangan tak berhingga.

2. Power Up Effect

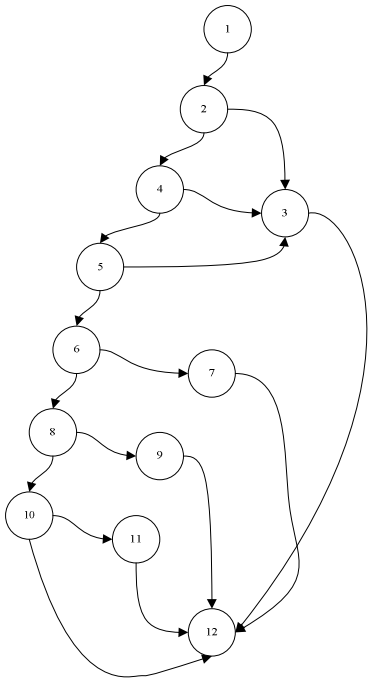
Menguji apakah sejumlah *state* pada permainan akan berubah sesuai dengan efek dari jenis power up tertentu ketika event pemicunya, yaitu *collision detection*, terjadi.

a. *Flow Chart*



Gambar 4.23 *Flow Chart* Pengujian FSM Power Up Effect

b. *Flow Graph*



Gambar 4.24 *Flow Graph* Pengujian FSM Power Up Effect

c. *Cyclomatic Complexity* V(G)

E = 17, N = 12

V(G) = E – N + 2 = 17 – 12 + 2 = 5 + 2 = 7

d. *Independent Paths*

Path 1 = 1 – 2 – 3 – 12

Path 2 = 1 – 2 – 4 – 3 – 12

Path 3 = 1 – 2 – 4 – 5 – 3 – 12

Path 4 = 1 – 2 – 4 – 5 – 6 – 7 – 12

Path 5 = 1 – 2 – 4 – 5 – 6 – 8 – 9 – 12

Path 6 = 1 – 2 – 4 – 5 – 6 – 8 – 10 – 11 – 12

Path 7 = 1 – 2 – 4 – 5 – 6 – 8 – 10 – 12

e. Hasil Uji

Tabel 4.12 Hasil Pengujian *White Box* FSM Power Up Effect

Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1, (void) 2, (true) 3, (void) 12, (void)	Skor Bertambah 250 Poin	Skor Bertambah 250 Poin	*Path* Dikerjakan
2	1, (void) 2, (false) 4, (true) 3, (void) 12, (void)	Skor Bertambah 500 Poin	Skor Bertambah 500 Poin	*Path* Dikerjakan
3	1, (void) 2, (false) 4, (false) 5, (true) 3, (void) 12, (void)	Skor Bertambah 1000 Poin	Skor Bertambah 1000 Poin	*Path* Dikerjakan
4	1, (void) 2, (false) 4, (false) 5, (false) 6, (true) 7, (void) 12, (void)	Shield pada Shield Bar Bertambah	Shield pada Shield Bar Bertambah	*Path* Dikerjakan
5	1, (void) 2, (false) 4, (false) 5, (false) 6, (false) 8, (true) 9, (void) 12, (void)	Health pada Health Bar Bertambah	Health pada Health Bar Bertambah	*Path* Dikerjakan
6	1, (void) 2, (false) 4, (false) 5, (false) 6, (false) 8, (false) 10, (true) 11, (void) 12, (void)	Memicu FSM Bullet Upgrade	Memicu FSM Bullet Upgrade	*Path* Dikerjakan
7	1, (void)	Power up	Power up	*Path*

	2, (false) 4, (false) 5, (false) 6, (false) 8, (false) 10, (false) 12, (void)	tidak memengaruhi permainan	tidak memengaruhi permainan	Dikerjakan

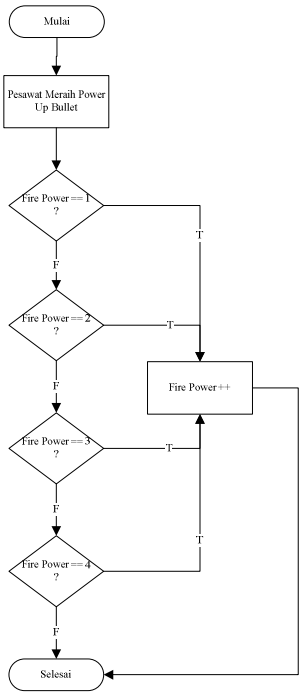
f. Kesimpulan

Dari hasil pengujian FSM power up effect yang dilakukan, didapat kesimpulan bahwa seluruh *node* yang ada pada setiap *independent path* telah dikerjakan, serta tidak terjadi pengulangan tak berhingga.

3. Bullet Upgrade

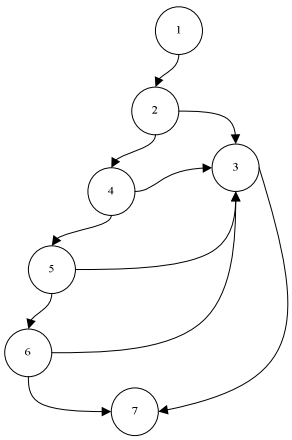
Menguji apakah *state* peluru akan berubah ketika event pemicunya, yaitu *collision detection*, terjadi.

a. *Flow Chart*



Gambar 4.25 *Flow Chart* Pengujian FSM Bullet Upgrade

b. *Flow Graph*



Gambar 4.26 *Flow Graph* Pengujian FSM Bullet Upgrade

c. *Cyclomatic Complexity* V(G)

E = 10, N = 7

V(G) = E – N + 2 = 10 – 7 + 2 = 3 + 2 = 5

d. *Independent Paths*

Path 1 = 1– 2 – 3 – 7

Path 2 = 1– 2 – 4 – 3 – 7

Path 3 = 1– 2 – 4 – 5 – 3 – 7

Path 4 = 1– 2 – 4 – 5 – 6 – 3 – 7

Path 5 = 1– 2 – 4 – 5 – 6 – 7

e. Hasil Uji

Tabel 4.13 Hasil Pengujian *White Box* FSM Bullet Upgrade

Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1, (void) 2, (true) 3, (void) 7, (void)	Fire Power Peluru Meningkat ke Level 2	Fire Power Peluru Meningkat ke Level 2	*Path* Dikerjakan
2	1, (void) 2, (false) 4, (true) 3, (void) 7, (void)	Fire Power Peluru Meningkat ke Level 3	Fire Power Peluru Meningkat ke Level 3	*Path* Dikerjakan
3	1, (void) 2, (false) 4, (false) 5, (true) 3, (void) 7, (void)	Fire Power Peluru Meningkat ke Level 4	Fire Power Peluru Meningkat ke Level 4	*Path* Dikerjakan
4	1, (void) 2, (false) 4, (false) 5, (false) 6, (true) 3, (void)	Fire Power Peluru Meningkat ke Level 5	Fire Power Peluru Meningkat ke Level 5	*Path* Dikerjakan

	7, (void)			
5	1, (void) 2, (false) 4, (false) 5, (false) 6, (true) 7, (void)	Fire Power Tetap Berada pada Batas Level	Fire Power Tetap Berada pada Batas Level	*Path* Dikerjakan

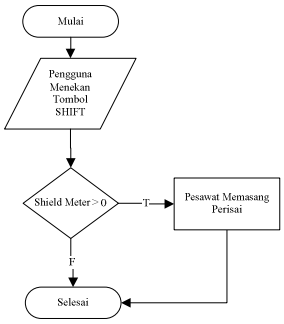
f. Kesimpulan

Dari hasil pengujian FSM bullet upgrade yang dilakukan, didapat kesimpulan bahwa seluruh *node* yang ada pada setiap *independent path* telah dikerjakan, serta tidak terjadi pengulangan tak berhingga.

4. Ship Shielded

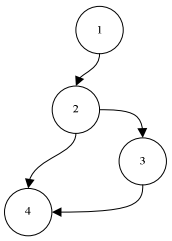
Menguji apakah perisai telah berfungsi yang ditandai dengan berubahnya *state* pesawat menjadi *shielded* ketika event pemicunya, yaitu *collision detection*, terjadi.

a. *Flow Chart*



Gambar 4.27 *Flow Chart* Pengujian FSM Ship Shielded

b. *Flow Graph*



Gambar 4.28 *Flow Chart* Pengujian FSM Ship Shielded

c. *Cyclomatic Complexity* V(G)

E = 4, N = 4

V(G) = E – N + 2 = 4 – 4 + 2 = 0 + 2 = 2

d. *Independent Paths*

Path 1 = 1– 2 – 3 – 4

Path 2 = 1– 2 – 4

e. Hasil Uji

Tabel 4.14 Hasil Pengujian *White Box* FSM Ship Shielded

Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1, (void)	Pesawat	Pesawat	*Path*

	2, (true) 3, (void) 4, (void)	Memasang Perisai	Memasang Perisai	Dikerjakan
2	1, (void) 2, (false) 4, (void)	Pesawat Tidak Dapat Memasang Perisai	Pesawat Tidak Dapat Memasang Perisai	*Path* Dikerjakan

f. Kesimpulan

Dari hasil pengujian FSM *ship shielded* yang dilakukan, didapat kesimpulan bahwa seluruh *node* yang ada pada setiap *independent path* telah dikerjakan, serta tidak terjadi pengulangan tak berhingga.

4.2.1.6 Kesimpulan Pengujian Alpha

Dari hasil pengujian yang dilakukan, dapat disimpulkan bahwa aplikasi permainan ArkAId XIV sudah berjalan sesuai dengan yang diharapkan. Hal tersebut berdasarkan dari hasil pengujian setiap kelas uji yang dalam hasilnya menyatakan bahwa seluruh node yang ada pada setiap independent path telah dikerjakan, serta tidak terjadi pengulangan tak berhingga. Secara fungsional aplikasi sudah dapat menghasilkan keluaran sesuai yang diharapkan.

4.2.2 Pengujian Beta

Pengujian beta merupakan pengujian yang dilakukan secara objektif, dimana dilakukan pengujian secara langsung terhadap pengguna dengan menggunakan kuisioner mengenai kepuasan pengguna atas aplikasi yang telah dibangun. Adapun metode penilaian pengujian yang digunakan adalah metode kuantitatif berdasarkan data dari pengguna. Pengujian dilakukan pada 30 responden dengan rentang usia 6 – 41 tahun dan setiap responden diberikan 5 pertanyaan.

4.2.2.1 Pertanyaan Pengguna

Adapun pertanyaan yang diajukan dalam kuesioner untuk pengguna yang menguji aplikasi ArkAId XIV adalah sebagai berikut:

1. Apakah tampilan aplikasi permainan ini secara umum sudah baik?
    a. Sangat Baik
    b. Baik
    c. Kurang Baik
    d. Tidak Baik
2. Apakah alur permainannya seru?
    a. Sangat Seru
    b. Seru
    c. Kurang Baik
    d. Tidak Baik
3. Apakah karakter pesawatnya mudah dikendalikan?
    a. Sangat Mudah
    b. Mudah
    c. Kurang Mudah
    d. Tidak Mudah
4. Apakah audionya enak didengar?
    a. Sangat Enak
    b. Enak
    c. Kurang Enak
    d. Tidak Enak

5. Apakah Anda menyukai aplikasi permainan ini?
    a. Sangat Suka
    b. Suka
    c. Kurang Suka
    d. Tidak Suka

4.2.2.2 Hasil Pengujian Beta

Berdasarkan data hasil kuisioner tersebut, dicari prosentase masing-masing jawaban dengan menggunakan rumus R = P/Q*100%, dimana :

P = Banyaknya jawaban responden tiap soal

Q = Jumlah responden

R = Nilai prosentase

4.2.2.3 Hasil Kuesioner

Berikut ini merupakan data yang didapat dari hasil kuesioner:

1. Apakah tampilan aplikasi permainan ini secara umum sudah baik?

Tabel 4.15 Hasil Kuesioner Tampilan Aplikasi

No	Jawaban	Responden	Prosentase (%)
1	Sangat Baik	14	46,67
2	Baik	16	53,33
3	Tidak Baik	0	0
4	Kurang Baik	0	0

2. Apakah alur permainannya seru?

Tabel 4.16 Hasil Kuesioner Alur Permainan

No	Jawaban	Responden	Prosentase (%)
1	Sangat Baik	11	36,67
2	Baik	18	60

3	Tidak Baik	1	3,33
4	Kurang Baik	0	0

3. Apakah karakter pesawatnya mudah dikendalikan?

Tabel 4.17 Hasil Kuesioner Navigasi Pesawat

No	Jawaban	Responden	Prosentase (%)
1	Sangat Baik	6	20
2	Baik	24	80
3	Tidak Baik	0	0
4	Kurang Baik	0	0

4. Apakah audionya enak didengar?

Tabel 4.18 Hasil Kuesioner Audio

No	Jawaban	Responden	Prosentase (%)
1	Sangat Baik	3	10
2	Baik	25	83,33
3	Tidak Baik	2	6,67
4	Kurang Baik	0	0

5. Apakah Anda menyukai aplikasi permainan ini?

Tabel 4.19 Hasil Kuesioner Tingkat Ketertarikan

No	Jawaban	Responden	Prosentase (%)
1	Sangat Baik	4	13,33
2	Baik	26	83,33
3	Tidak Baik	0	86,67
4	Kurang Baik	0	0

4.2.2.4 Kesimpulan Pengujian Beta

Berdasarkan Hasil dari pengujian beta dapat disimpulkan bahwa aplikasi yang telah dibangun dapat menumbuhkan kembali dan meningkatkan tingkat ketertarikan pengguna aplikasi permainan terhadap permainan ber-*genre arcade-shooter*.