Pasinformatik, Volume 3, Issue 2, July 2024
E-ISSN : 2986-5360

Analisis dan Simulasi Komputasi Kecerdasan *Non-Player Character Boss-Berial* dalam *Game Devil May Cry* 4

Ririn Dwi Agustin [1] , Azhar Fiqri Dwiyana [2]

Jurusan Teknik Informatika, Fakultas Teknik, Universitas Pasundan
Jln. Dr. Setiabudhi no. 193 Bandung, Jawa Barat
[1] ririn_dwia@unpas.ac.id, [2] 193040107.azhar@mail.unpas.ac.id

Abstrak : Kecerdasan NPC sangat penting dalam meningkatkan kualitas pengalaman bermain dengan respons yang baik terhadap tindakan pemain dan tantangan yang sesuai.. *Devil May Cry* 4 merupakan *game* yang dikembangkan oleh *Capcom*, dan menjadi salah satu *game* yang populer dalam *genre* action. Berial adalah salah satu NPC cerdas dalam *Devil May Cry* 4, yang berperan sebaga boss iblis raksasa berbentuk naga yang dilengkapi dengan cakar dan tanduk besar dan memiliki penampilan yang mengesankan mencerminkan kekuatannya sebagai salah satu bos dalam permainan. Berial memiliki tugas untuk bertahan dan membunuh lawan dalam kondisi jarak pendek maupun jarak jauh, dengan cara beraksi yang adaptif dengan kondisi kekuatan pemain maupun kondisi kekuatannya sendiri. Penelitian ini fokus pada analisis perilaku kecerdasan NPC Boss Berial menggunakan model komputasi Finite State Machine (FSM) dan mengkonstruksinya ke dalam sebuah modul perangkat lunak yang dideskripsikan menggunakan UML. Penelitian Perilaku Berial direpresentasikan memiliki delapan state dan tujuh kejadian yang menjadi pemicu perubahan state. State tersebut yakni *idle, walking,* serangan pedang api, serangan cakar api, AOE attack, bertahan dengan pedang api, *counter attack*, dan mati. Sedangkan ragam kejadian yang menjadi perhatian Berial adalah tidak ada pemain, pemain memasuki area pantau, pemain menyerang, pemain terkena serangan, pemain mendekat, pemain menjauh, pertahanan berial melemah. Dikarenakan sebuah event bisa muncul di lebih dari satu state yang berbeda maka karena digunakan struktur data untuk tahap konstruksi perangkat lunak adalah *incidency matriks*, maka dilakukan normalisasi dengan cara membuat sebanyak N nama yang berbeda untuk event yang muncul di sebanyak N state. Hasil simulasi berapa perangkat lunak web yang dibangun dengan laravel dan DBMS mysql untuk menyimpan instansiasi dari class. Aplikasi simulasi kecerdasarn NPC memiliki fitur tampilan awal, tampilkan *current state*, dan pilih kejadian. Semua state beserta ragam pilihan yang disajikan telah diuji coba dan menunjukkan perubahan ke *next state* yang sesuai dengan *incidency matrix* dari FSM.

Kata Kunci : *Game*, *Game* aksi, kecerdasan buatan, Karakter NPC

I. PENDAHULUAN

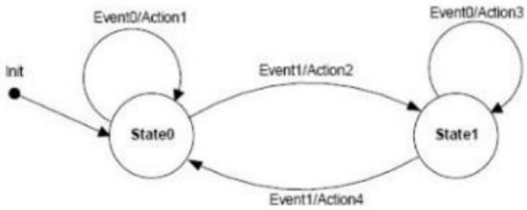
Devil May Cry 4 merupakan *game* yang dikembangkan oleh *Capcom*, dan menjadi salah satu *game* yang populer dalam *genre* action. *Game* ini menawarkan pengalaman bermain yang mendalam dan taktis di mana pemain dapat menghadapi berbagai musuh, termasuk karakter NPC, yang memiliki tingkat kecerdasan buatan. NPC merupakan salah satu elemen karakter pada game, yang berdasarkan perannya dipilah menjadi enemies, suuport partner, dan allied [1]. Kecerdasan NPC menjadi faktor penting dalam menentukan kualitas pengalaman bermain. Sebuah karakter NPC yang cerdas dapat merespons tindakan pemain dengan baik, memberikan tantangan yang sesuai, dan menghadirkan pengalaman yang lebih mendalam dan dinamis [2]. Kecerdasan NPC juga mempengaruhi sejauh mana cerita permainan dapat berkembang dan seberapa imersif dunia permainan ini bagi pemain. Dalam *game Devil May Cry* 4 karakter NPC memiliki perilaku kecerdasan yang sesuai dengan konteks *game*. NPC perlu dapat memahami strategi pemain, beradaptasi dengan taktik yang berbeda, dan menjaga tingkat kesulitan yang seimbang agar pemain tidak merasa terlalu mudah atau terlalu sulit. Tujuan penelitian yang hasilnya dituangkan dalam tulisan adalah memodelkan kecerdasan NPC ke dalam salah satu model komputasi FSM dan mensimulasikan bagaimana cara konstruksi FSM ke dalam tabel basis data dan konstruksi kelas-kelas MVC-nya. Tulisan ini diharapkan bisa bermanfaat dalam pembelajaran matematika diskret, teori otomata, serta pemrograman khususnya untuk topik graph, FSM. Dengan contoh nyata kebutuhan menggunakan FSM dalam aplikasi game diharapkan terjadi bahan pembelajaran kontekstual yang lebih memotivasi pembelajar. [3]

II. METODE PENELITIAN

Penelitian ini diawali dengan studi literatur tentang elemen game, gameplay, beberapa model agen cerdas seabgai teknik merepresentasikan perilaku NPC. Langkah 2 eksplorasi *game Devil May Cry 4* untuk bahan menyusun anatomi game tersebut pada level konsep dan desain. Langkah 3 menganalisis model kecerdasan NPC Boss *game Devil May Cry* 4 dan merepresentasikannya menggunakan FSM. Langkah 4 perancangan model prototipe. Perancangan model prototipe berdasarkan temuan pada saat perumusan aspek tingkat kecerdasan NPC di *game Devil may Cry* 4 untuk selanjutnya dibuatkan prototipenya. Langkah 5 Kesimpulan penelitian tentang model komputasi representasi kecerdasan perilaku NPC Cerdas dan teknik konstruksinya.

Model komputasi yang biasanya digunakan untuk merepresentasikan kecerdasan perilaku NPC adalah FSM (Finite State Machine). Game loop akan menelusuri FSM dengan memeriksa apakah kondisi transisi terpenuhi atau tidak dengan menangkap event yang terjadi dan di respon oleh system game melalui state yang aktif. [4] [5]

Finate State Machines (FSM) adalah sebuah metodologi perancangan sistem kontrol yang menggambarkan tingkah laku atau prinsip kerja sistem dengan menggunakan state (Keadaan), event (kejadian) dan action (aksi). Dalam state machine sistem menempati satu state (keadaan). Sistem akan beralih atau bertransisi menuju ke state lain jika mendapatkan masukan event tertentu atau dilakukan aksi tertentu. Sistem akan tetap melakukan aksi yang sama pada suatu state sampai sistem menerima event tertentu baik yang berasal dari perangkat luar atau komponen dari sistem itu sendiri. Setiap state terhubung oleh transisi dan setiap transisinya mengarah ke satu state lainnya. Transisi keaadan ini umumnya juga disertai oleh aksi yang dilakukan oleh sistem ketika menanggapi masukan yang terjadi. Aksi yang dilakukan tersebut dapat berupa aksi sederhana yang melibatkan rangkaian proses yang relatif rumit. [6] Secara formal FSM dapat dinyatakan sebagai M (S ,I, O, *f*, *g*,S_0) terdiri dari enam tuple, yakni (1)sekumpulan *state*, (2)input (opsional), (3)output, (4)fungsi transisi yang memetakan *state-c + input* ke *state-n*, (5) fungsi output, (6)state awal. [7] Input biasanya berupa karakter atau klik button di UI namun bisa juga berupa waktu. Dalam Synchronous FSM, transisi antar keadaan dikendalikan oleh sinyal jam atau waktu. [8] Jika ditinjau dari fungsi transisi ada dua macam FSM, yakni mesin Moore dan mesin Mealy. Pada mesin moore, output hanya bergantung kepada *state* sehingga seakan tidak ada fungsi transisi. Sebaliknya pada mesin Mealy output dipengaruhi juga oleh sinyal input selain state yang terpilih. Mesin Moore lebih stabil dan sederhana, sedangkan mesin Mealy lebih responsif terhadap nilai input. Diagram pada ***Gambar 1***, memperlihatkan contoh FSM, yang jika dinyatakan dalam definisi formal adalah sebagai berikut :
S=[State0, State1]; I=[event0,event1] ; O=[action1, action2, action3, action4] ;
f1(State0, Event0, State0), f2(State0,Event1,State1), f3(state1,Event0, state1), f4(state1,Event1,state0)
g1(state0,Event0; Action1) , g2(State0,Event1,Action1), f3(state1,Event0, Action3), f4(state1,Event1, Action4)
S_0 (State0)
Ketika sistem mulai dihidupkan, sistem akan berada di state0, pada keadaan ini sistem akan menghasilkan Action1 jika terjadi masukan Event0, sedangkan jika terjadi Event1 maka Action2 akan dieksekusi kemudian sistem selanjutnya bertransisi ke keadaan State1 dan seterusnya. [9]



Gambar 1 Contoh FSM

Adjacency matriks atau matriks ketetanggaan, yakni matriks yang menyimpan informasi pasangan simpul yang terhubung melalui sebuah busur. Untuk Graph berarah, pada matriks M[j,k] , j merupakan simpul awal dan k merupakan simpul tujuan. *Adjacency list* akan mengefisienkan penyimpanan graph terlebih ketika mayoritas sel di *adjacency matriks* bernilai 0. Pendekatan *Adjacency list* cocok untuk kebutuhan penyimpanan di database, dimana record yang disimpan dari matriks hanya yang bernilai tidak 0.
Tabel 1 Representasi FSM pada **Gambar 1** dalam bentuk Adjacency Matriks

Tabel 1 Ilustrasi Representasi Adjacency Matriks

	State0	State1
State0	1	1
State1	1	1

Jika disimpan dalam database maka definisi atribut tabelnya adalah R-FSM (stateawal, statetujuan, labelbusur, informasibusur). Ilustrasinya dapat dilihat pada

Tabel 2 Representasi Adjacency Matriks dalam Tabel Relasional

Stateawal	Statetujuan	labelbusur	Informasibusur
State0	State0	Event0	Action1
State0	State1	Event1	Action2
State1	State1	Event2	Action3
State1	State0	Event3	Action4

Definisi dari *incidency matriks* untuk sebuah graph G **berarah** dengan simpul S_1, S_2,S_3 …S_N dan busur E_1, E_2, E_3, ..E_K adalah sebuah matrik M berukuran NxK dengan barisnya menyimpan simpul S_1 s.d S_N dan kolomnya adalah semua busur yang menyimpan E_1 s.d E_K [6], dimana

i) pada $M_{[I,J]}$ berisi -1 jika S_I adalah awal dari busur E_J.
ii) pada $M_{[I,J]}$ berisi 1 jika S_I adalah akhir dari busur E_J
iii) pada $M_{[I,J]}$ berisi 0 jika pada S_I bukan awal atau akhir dari busur E_J

Definisi tersebut belum mengakomodasi adanya *self loop*. Salah satu modifikasi yang bisa dilakukan adalah mendefinisikan isi baru, semisal nilai integer 2 untuk menandai bahwa *event/action* tertentu tidak menyebabkan perpindahan *state*.

Tabel 3 Representasi FSM Gambar 1 dalam Incidency Matriks

	Event0-Action1	Event0-Action3	Event1-Action2	Event1-Action4
State0	2		-1	1
State1		2	1	-1

Pada **Gambar 1** diatas *Event0* ataupun *Event1* sebenarnya muncul dua kali, namun ketika digabungkan dengan action, semua busur menjadi menjadi unik. Jika busur yang sama, satu jenis *event-action* muncul lebih dari satu kali, maka perlu melakukan penandaan sedemikian hingga *event-action* menjadi unik. Jika Incidency diatas disimpan dalam database ilustrasinya dapat dilihat pada gambar dibawah ini

Tabel 4 Ilustrasi Tabel Realisonal untuk Menyimpan Incidency Matriks

IdBusur	LabelBusur	SimpulAwal	Simpultujuan	Informasi
E-1	Event0	State0	State0	Action1
E-2	Event0	State1	State1	Action3
E-3	Event1	State0	State1	Action2
E-4	Event1	State1	State1	Action4

Karena pada FSM informasi *event* dan *action* adalah hal penting dan sering menjadi dasar proses *searching* maka *incidency matriks* adalah lebih tepat untuk game.

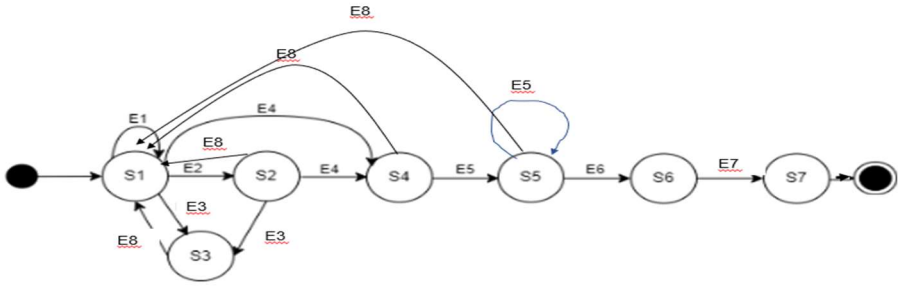
III. HASIL DAN PEMBAHASAN

A. Analisis Kecerdasan NPC Boss Berial

Analisis ini dibatasi dalam mode NPC sebagai *single entity* dalam arti mengasumsikan NPC yang dianalisis tidak berinteraksi dengan NPC lainnya.. Dalam *game* Devil May Cry 4, Berial adalah salah satu NPC demon yang memberikan tantangan sulit untuk ditaklukkan. Berial adalah memiliki tubuh bagian bawah seperti banteng besar, kaki dan telapak kaki seperti kadal atau naga, tubuh bagian atas seperti manusia, dan wajah seperti singa, dengan tanduk ganas di atasnya, dan dia menjulang tinggi di atas rumah-rumah satu lantai di medan perang. Tubuhnya hitam, seperti lava yang dingin, dengan semburan api besar seperti sayap yang menyembur dari bahu humanoidnya dan lebih banyak api yang mengalir dari ekornya. Ketika apinya padam, api-api ini menghilang dan lipatan-lipatan di tubuh Berial berubah menjadi lebih seperti cahaya jingga yang membara. Selain cakarnya, Berial bertarung dengan pedang api besar, yang penampilannya mirip dengan Zweihander, tetapi ukuran dan kekuatannya yang besar memungkinkannya untuk menggunakannya dengan satu tangan. Serangannya lambat, tetapi kuat dan dengan jangkauan yang luas; dua aspek yang tidak diragukan lagi membantunya menaklukkan Neraka Api [10]. Personality Berial ; Berial tampaknya adalah iblis yang penasaran dengan banyak sifat kepribadian yang terhormat dan pejuang. Awalnya ia menganggap Nero tidak mengancam dan mengabaikannya, tetapi ketika Nero mengayunkan pedangnya untuk memadamkan api, ia menjadi penasaran dan menghadapi Nero. Ia tidak suka direndahkan dan melampiaskan amarahnya pada Nero dengan melibatkannya dalam pertempuran. Berial tidak setuju dengan pandangan bahwa Nero dan manusia lainnya bercita-cita menjadi Dewa. Kekuatan dan Kemampuan Berial : Berial adalah iblis tingkat tinggi dan dianggap ahli dalam hal api. Dengan tubuhnya yang besar seperti singa, ia memiliki serangan yang lebih kuat dan daya tahan yang lebih tinggi daripada iblis api yang lebih kecil. Ia juga menggunakan pedang besar yang menciptakan ledakan saat terkena benturan. Berial juga dapat menghasilkan ledakan api besar saat perlindungannya hilang, memulihkan semua apinya. Selain itu, ia sendiri dapat berubah menjadi api dan terbang dengan kecepatan tinggi, seperti yang terlihat saat ia melarikan diri kembali ke Gerbang Neraka. Secara audio visual perilaku Berial menghadapi Nero dapat dilihat di youtube [11]

B. Model *Finite State Machine* untuk *Boss Berial*

Boss Berial memiliki tiga tujuan, yakni menyerang pemain, bertahan, malakukan adaptasi terhadap pemain. Dalam melaksanakan tujuan menyerang pemain, Berial akan menyerang pemain dengan melakukan serangan jarak dekat menggunakan pedang apinya atau serangan jarak jauh menggunakan cakarnya yang akan mengeluarkan api dari dasar tanah menuju ke arah pemain. Saat pemain berada dalam jangkauan yang jauh maka Berial akan melakukan serangan jarak jauh dengan melompat dan bersiap untuk menangkap pemain dengan cakarnya. Berikut merupakan diagram Finite State Machine(FSM) yang mencakup *state* (Keadaan), *event* (kejadian) dan *action* (aksi).



Gambar 2 Model FSM dari Perilaku Boss Berial

Tujuan Berial adalah mengalahkan pemain. Pada FSM Berial *state* digambarkan berbentuk bulat dan *event* yang digambarkan berbentuk panah. Proses perpindahan *state-state* pada Berial dapat dilihat pada gambar 2.
Keterangan pada gambar 2 :

State	Event -Aksi
S1 - Idle S2 - Walking S3 – menyerang dg pedang api S4 - Serangan menyergap cakar api S5 - bertahan menggunakan pedang S6 - bertahan menggunakan ledakan api S7 - NPC melarikan diri ke gerbang Neraka	E1 - Tidak ada pemain di area berial - *Idle* E2 - pemain memasuki area berial – *Walking to Player* E3 - pemain mengayunkan pedang dan berada di jarak dekat dari berial -*Attack with Fire Sword* E4 - pemain mengayunkan pedang dan berada di jarak jauh dari berial – *Attack by ambushing with Jumps and Claws* E5 –Berial terkena serangan – *defend with sword* E6 - pertahanan berial melemah – *defend with large bursts of fire* E7 - Berial menjadi sangat lemah– *Talk to Nero and Retreat from battlefield* E8 - pemain keluar dari area Berial – *Go to Idle*

B.1 Adjacency Matriks

Pada FSM di **Gambar 2**, jika direpresentasikan dengan *adjacency matriks* menjadi **Tabel *2***. Baris dan kolomnya adalah setiap simpul sehingga ukuran matriksnya adalah M_{7x7}. $Sel_{i,j}$ diisi dengan identitas busur , dimana i merepresentasikan simpul awal dan j merepresentasikan simpul tujuan. *Self loop* tergambar dari adanya identitas busur yang di pasang di sel i=j .

Tabel 5 Adjacency Matriks dari FSM pada Gambar 2

State	State						
	S1	S2	S3	S4	S5	S6	S7
S1	E1	E2	E3	E4			
S2	E8		E3	E4			
S3	E8						
S4	E8				E5		
S5	E8				E5	E6	
S6							E7
S7							

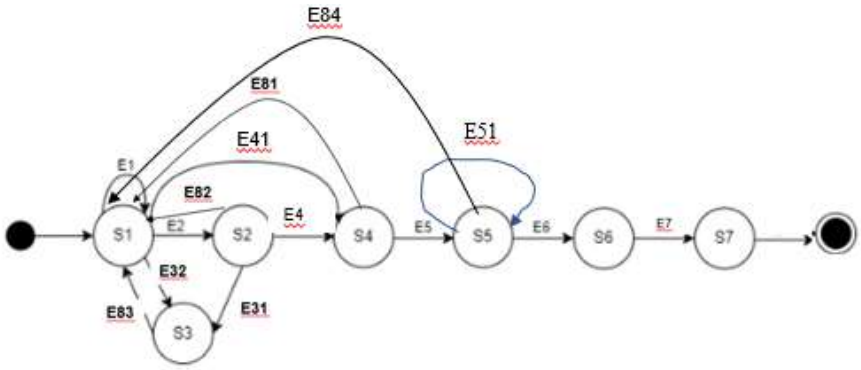
Proses penyimpanan ke dalam tabel relasional dengan memperhatikan optimasi ruang penyimpanan dilakukan dengan mekanisme (1) setiap sel yang ada identitas busurnya akan menjadi satu baris di dalam tabel. Jika ada satu sel yang berisi lebih dari satu busur, maka untuk setiap busurnya menjadi satu baris. (2) Kolom dari tabel adalah i sebagai kolom pertama dengan nama simpul awal , j sebagai kolom kedua dengan nama simpul tujuan , dan kolom ketiga adalah identitas busur.

Tabel 6 Isi Tabel Penyimpanan Matriks Adjacency di **Tabel 5**

No	Simpul Awal	Simpul tujuan	Identitas-busur	Informasi
1	S1	S1	E1	*Idle*
2	S1	S2	E2	*Walking to Player*
3	S1	S3	E3	*Attack with Fire Sword*
4	S1	S4	E4	*Attack by ambushing with Jumps and Claws*
5	S2	S1	E8	*Go to Idle*
6	S2	S4	E4	*Attack by ambushing with Jumps and Claws*
7	S2	S3	E3	*Attack with Fire Sword*
8	S3	S1	E8	*Go to Idle*
9	S4	S5	E5	*defend with sword*
10	S4	S1	E8	*Go to Idle*
11	S5	S5	E5	*defend with sword*
12	S5	S6	E6	*defend with large bursts of fire*
13	S5	S1	E8	*Go to Idle*
14	S6	S7	E7	*Talk to Nero and Retreat from battlefield*

B2.Incidency Matriks

Pada FSM di **Gambar 2**, E_8 muncul tiga kali menghubungkan S_2-S_1, S_3-S_1, dan S_4 ke S_1 dan E_3 muncul dua kali menghubungkan S_1 ke S_2 dan S_2 ke S_3, E_4 muncul dua kali, yakni dari S_2 ke S_4 dan dari S_1 ke S_4. Untuk bisa direpresentasikan ke dalam *incidency matriks* sebagaimana diuraikan pada subab kajian teori maka FSM tersebut perlu dinormalisasi dengan membuat busur menjadi nama yang unik.



Gambar 3 Model FSM Boss Berial Hasil Normalisasi

Tabel 7 Ilustrasi Incidency Matriks dari FSM Gambar 3

State	EVENT-ACTION													
	E1	E2	E31	E32	E4	E42	E5	E51	E6	E7	E81	E82	E83	E84
S1	2	-1		-1	-1						1	1	1	1
S2		1	-1			-1						-1		
S3			1	1									-1	
S4					1	1	-1				-1			
S5							1	2	-1					-1
S6									1	-1				
S7										1				

State awal diinformasikan dari definisi FSM, yakni S1, dicari pada kolom sebelah kiri yakni dari S_1 dan berurutan kebawah sampai S_8. *Event* berada pada kolom bagian atas yang berwarna biru, dimulai dari E_1 dan berakhir di E_8. Dari baris S_1, cari kolom yang bernilai -1

atau 2. Hasilnya ada empat pilihan event yang disajikan kepada pemain untuk dipilih, yakni E_1 yang bernilai 2, E_2, E_{32}, dan E_4 , yang semuanya bernilai -1. Andaikan dipilih E_2 maka dicari baris yang bernilai 1 di kolom E_2, yakni state 2. Pada baris S_2, berikutnya cari kolom yang bernilai -1. Hasilnya ada dua pilihan event yang disajikan untuk dipilih karena ada E_{31} dan E_{82} yang nilainya -1. Misalnya dipilih E_{31} maka di kolom E_{31} di cari baris yang bernilai 1. Demikian seterusnya simulasi dijalankan berdasarkan pilihan user yang disajikan pilihannya berdasarkan *incidency matriks*. Pada proses optimasi penyimpanan dan akses di tabel basis data relasional, hanya sel yang memiliki nilai dari *incidency matriks* yang disimpan (lihat**Tabel *8*)**.

Tabel 8 Tabel Relasional DB untuk menyimpan incidency matriks pada tabel 5

No	IdBusur	Label Busur	Simpul Awal	Simpul tujuan	Informasi
1	E1	E1	S1	S1	*Idle*
2	E2	E2	S1	S2	*Walking to Player*
3	E3	E31	S2	S3	*Attack with Fire Sword*
4	E3	E32	S1	S3	*Attack with Fire Sword*
5	E4	E4	S1	S4	*Attack by ambushing with Jumps and Claws*
6	E4	E41	S2	S4	*Attack by ambushing with Jumps and Claws*
7	E5	E5	S4	S5	*defend with sword*
8	E5	E5-1	S5	S51	*defend with sword*
9	E6	E6	S5	S6	*defend with large bursts of fire*
10	E7	E7	S6	S7	*Talk to Nero and Retreat from battlefield*
11	E8	E8-1	S4	S1	*Go to Idle*
12	E8	E8-2	S2	S1	*Go to Idle*
13	E8	E8-3	S3	S1	*Go to Idle*
14	E8	E8-4	S3	S1	*Go to Idle*

B3.Analisis Perbandingan

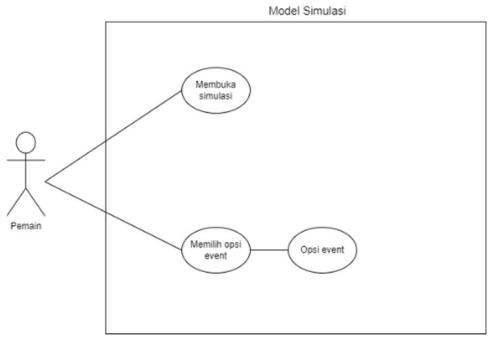
Perbandingan dari **Tabel 6** dengan **Tabel *8*** terlihat bahwa jumlah barisnya sama, yakni 14. Setiap *tupple* diantara kedua tabel sebenarnya merepresentasikan *entity* yang sama. Kesamaan ini akan lebih terlihat jika keduanya diurutkan berdasarkan **simpul awal**. Algoritma komputasi untuk kendali diantara kedua tabel akan sama, yakni

(1) Current State (CS) diisi dengan State awal dari definisi FSM
(2) Pilih kelompok baris (KB) yang kolom state awal = CS
(3) Jika KB tidak kosong maka ambil kolom IdBusur,StateTujuan dari setiap baris KB simpan di daftar pilihan (LO)
    a. Tampilkan LO ke user untuk dipilih , simpan di indeks Terpilih di (Itp)
    b. Update CS dengan kolom state tujuan di baris TP
(4) Jika KB kosong maka berhenti (*final state*)

C. Pengembangan Modul Simulasi

a. Analisis dan Rancangan Sistem

Analisis sistem merupakan suatu kegiatan untuk mendefinisikan kebutuhan fungsional modul simulasi yang akan dikembangkan dengan cara mengidentifikasi atau menyelidiki proses yang terjadi. Diagram Use Case

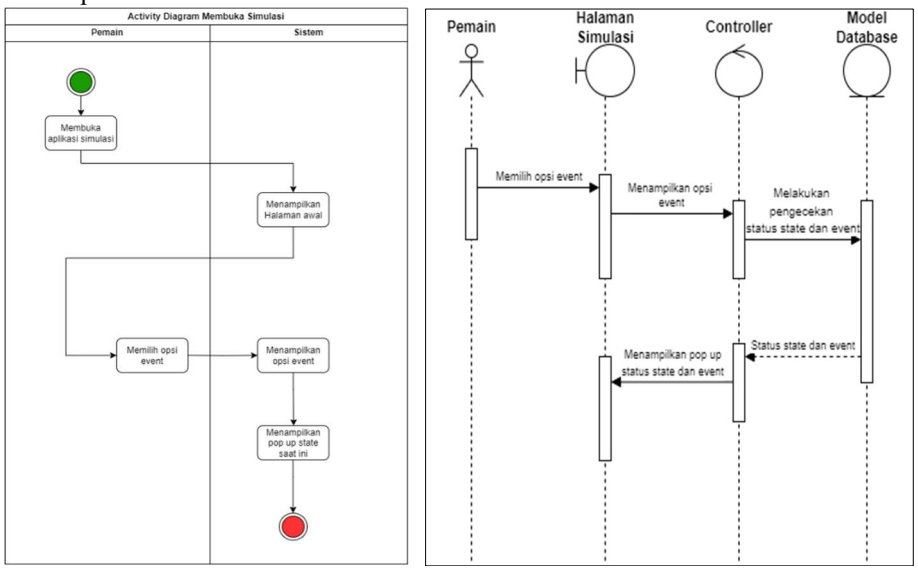


Gambar 4 Use Case Modul Simulasi

Use case pada Gambar 4 “membuka simulasi” berfungsi untuk menampilkan tampilan awal aplikasi simulasi dan menginisiasi state NPC ke state awal. Use case “opsi event” menampilkan *next state* yang *enabled* sesuai dengan FSM. Use case “memilih opsi event”

menerima pilihan event berikutnya dan menentukan state berikutnya berdasarkan FSM.

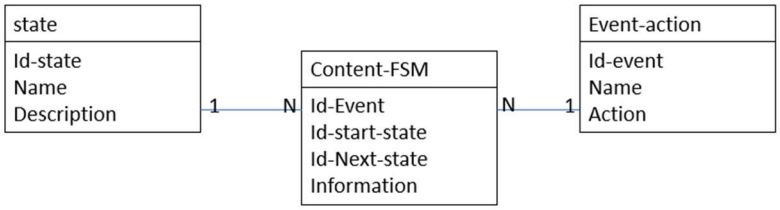
(1) Diagram Activity dan Sequence



Gambar 5 Diagram a) activity dan b) Sequence modul simulasi

Diagram di Gambar 5 bagian kiri, menggambarkan apa yang dilakukan oleh aktor untuk simulasi. Pemain membuka simulasi. Sistem akan menampilkan halaman awal, lalu Pemain dapat memilih opsi event maka sistem akan menampilkan beberapa opsi event untuk melakukan simulasi. Kemudian pemain memilih opsi event maka sistem akan menampilkan sebuah pop up state saat ini.
Pada diagram sequence di Gambar 5 bagian kanan, bisa dibaca bahwa jika aktor memilih opsi event, halaman akan menampilkan opsi event dan di proses oleh controller. controller akan melakukan pengecekan status state dan event yang akan ditampilkan. Dan controller akan memproses menampilkan pop up status state dan event.

(2) Diagram Class



Gambar 6 Diagram Class

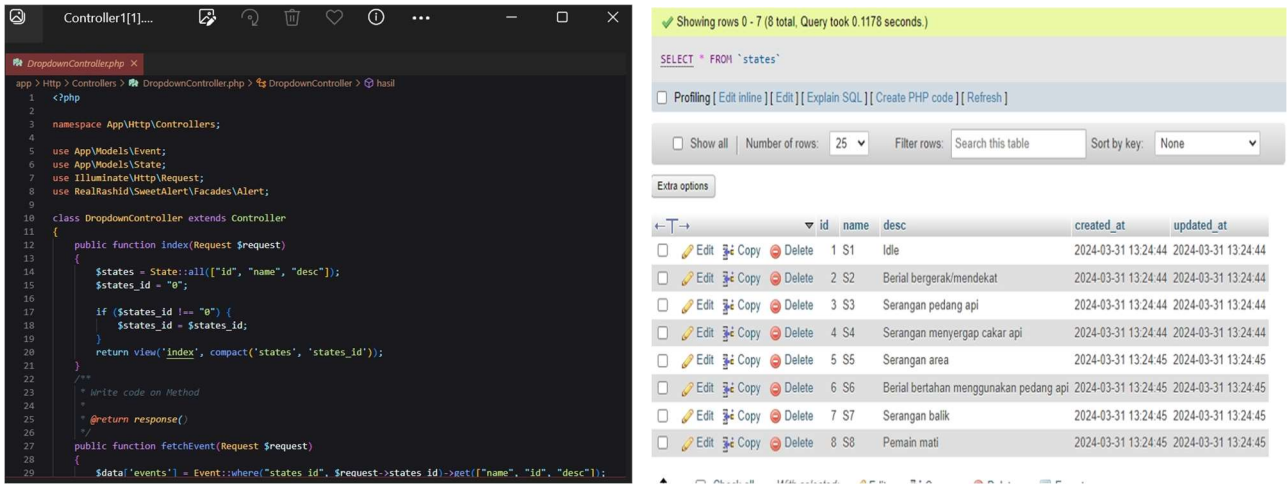
Diagram class mendeskripsikan entitas yang perlu disimpan dinamika perannnya di dalam sistem aplikasi. Class pada layer konstruksi akan direpresentasikan menjadi tabel atau relasi di dalam basis data. Pada *Gambar 6* ada tiga tabel yang diperlukan untuk mengelola kecerdasan NPC dengan FSM, yakni daftar-state , daftar event , dan content-FSM nya sendiri yang mendeskripsikan alur kendali perilaku cerdas NPC. Entitas content-FSM mengambil *primary key* dari entitas *state* dan *event-action*.

b. Rancangan wireframe Simulasi

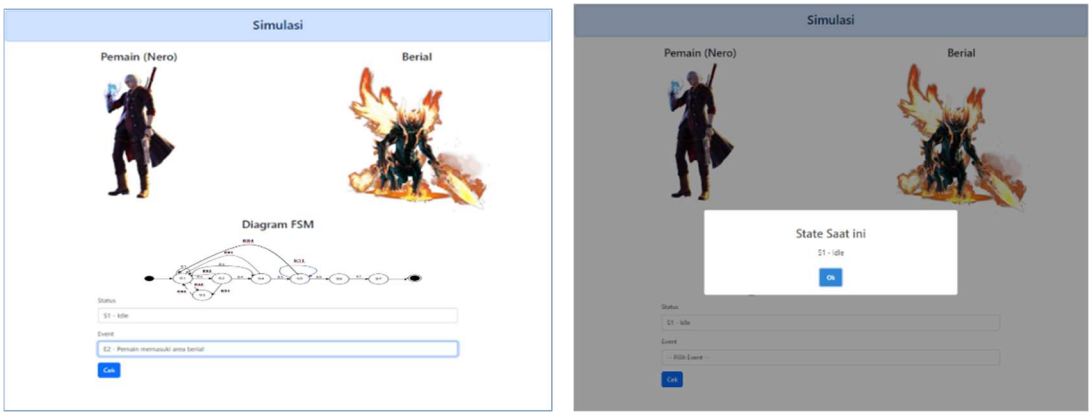
Wireframe adalah gambaran kasar dari prototype yang akan dibuat. Modul simulasi memiliki sebuah form utama yang dibagi menjadi bagian informasi dan kendali simulasi sebagaimana dicantumkan di **Error! Reference source not found.**. Pada bagian informasi ditampilkan gambar pemain, *boss berial*, dan diagram FSM-nya. Pada bagian kendali simulasi ditampilkan *current state* dan pilihan *event* yang terkait dengan *current state*. Terdapat fitur PopUP untuk mendetilkan penjelasan tentang *current* state.

c. Konstruksi dan Uji Coba Model
Modul simulasi dikonstruksi dalam platform aplikasi berbasis web menggunakan Laravel. FSM disimpan dalam basis data relasional mysql. Terdapat satu kelas Form , dan lima fungsi dalam Controler, dan tiga kelas model. Gambar 8 menampilkan

cuplikan dari source code pengendali ekseskusi FSM dan tabel dalam database yang menyimpan *incidency matrix* dari FSM.



Gambar 7 Capture sebagian Source Code dan Tabel Database penyimpan State



Gambar 8 Tampilan Awal State berada pada S_1

Berikutnya pada Gambar *8* ditampilkan form pada saat awal aplikasi diaktifkan. Dimana modul simulasi berada pada state awal S_1- dengan aksinya *Iddle* dan eventnya masih kosong. Ketika diklik Status, maka akan muncul *Popup State* saat ini. Ditampilkan ketika dipilih E_2 diantara empat pilihan event, yakni pemain masuk ke area pemantauan Berial, maka ditampilkan *State* berikutnya adalah S_2 dimana aksi yang dilakukan adalah Berial akan bergerak mendekati pemain. Berikutnya pada **Error! Reference source not found.** Current State adalah E_2. Sistem Modul Simulasi akan menampilkan 3 pilihan E_8, E_3, dan E_4 , dipilih E_3. Pada proses pengujian telah ditelusuri semua alternatif event dari setiap *state* dan hasilnya sudah sesuai dengan alur pada FSM. Jika dianalisis kesesuaiannya dengan perilaku Boss Berial, model FSM yang dihasilkan

IV. KESIMPULAN

Beberapa hal yang bisa disimpulkan dari kajian ini adalah sebagai berikut

1. Perilaku cerdas Berial Boss telah direpresentasikan ke dalam FSM (*finite state automata*) dengan tujuh macam *state* dan delapan *edge*. Terdapat *edge* yang berupa *self loop*, enam *state* diantaranya memiliki derajat lebih dari satu.
2. Untuk persiapan ke tahap konstruksi perangkat lunak yang mengkomputasi FSM, bisa digunakakan pendekatan *adjacency matriks* maupun *incidency matriks*, namun diantara keduanya tahapannya lebih sederhana adalah *adjacency matriks*

3. Dengan memperhatikan optimasi penyimpanan sparse matrix , hasil transformasi dari *adjacency matriks* maupun *incidency matriks* menghasilkan baris yang sama. Karena secara substansi tabel hasil transformasi dari kedua macam representasi adalah sama, maka bisa diwakili dengan satu algoritma kendali perilaku cerdas NPC.

Ucapan Terima Kasih

Peneliti mengucapkan banyak terimakasih kepada Fakultas Teknik dan Program Studi Teknik Informatika Universitas Pasundan, Ketua Program Studi, para dosen dan pihak lain yang telah mendukung berjalannya kegiatan penelitian ini.

Referensi

[1] R. D. Agustin, "KERANGKA ANALISIS KOMPONEN KONSEP DAN DESAIN GAME," *Jurnal Ilmiah Teknologi Informasi Terapan (Jitter) - ISSN : 2407 - 3911,* vol. III, no. No 2, 15 April 2017.

[2] R. I. Rouse, Game Design: Theory & Practise, Wordware Publishing, Inc, 2005.

[3] M. e. A. Nadeem, "AR4FSM: Mobile Augmented Reality Application in Engineering Education for Finite-State Machine Understanding," *Education Science,* vol. 12, no. 8, pp. 555-561, 2022.

[4] D. Jagdale, "Finite State Machine in Game Development," *International Journal of Advanced Research in Science, Communication and Technology (IJARSCT),* Vols. 10,, no. 1, pp. 386-390, Oktober 2021.

[5] W. E. ,. e. A. Hidayat, "Penerapan Finite State Machine pada Battle GameBerbasis Augmented Reality," *JEPIN(Jurnal Edukasi dan Penelitian Informatika),* vol. 5, no. 1, pp. 48 -61, 2019.

[6] P. Fred E. Szabo, "The Linear Algebra Survival Guide," 20 07 2024. [Online]. Available: https://www.sciencedirect.com/topics/mathematics/adjacency-matrix.

[7] T. D. K. Siregar, "https://informatika.stei.itb.ac.id," ITB, 2013. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2012-2013/Makalah2012/Makalah-IF2091-2012-094.pdf. [Accessed 7 2024].

[8] M. G. Khairi, "Penerapan Finite State Machine dalam Game Pada Model Karakter Pemain Menggunakan PADAMODELKARAKTERPEMAIN MENGGUNAKAN GODOT ENGINE," *Scientica Jurnal Ilmiah Sainsdan Teknologi,* vol. 2, no. 7, p. 45, 2024.

[9] E. a. E. H. Yulsilviana, "Penerapan Metode FInite State Machine (FSM) pada Game Agent Legenda Anak Borneo," *Sebatik,* vol. 23, no. 1, pp. 116-123, 2019.

[10] "Fandom of Devil May Cry," fandom.com, 23 10 2023. [Online]. Available: https://devilmaycry.fandom.com/. [Accessed 20 7 2024].

[11] Chris, Director, *Berial Boss Fight - Devil May Cry 4 Remaster (4K Ultra HD) Boss #1.* [Film]. Zanar Aesthetics, 2017.

[12] M. Roni, R. D. Agustin and H. Supeno, "Analisis Model Kecerdasan Non Player Character Sebagai Pembentuk Challeng ANALISIS MODEL KECERDASAN NON PLAYER CHARACTER SEBAGAI PADA DOTA 2," Repository Unpas, Bandung, 2012.

[13] M. A. Akbar, "Koordinasi pasukan NPC menggunakan agen cerdas berbasis fuzzy coordinator dan distribusi Gaussian," Repository ITS, Surabaya, 2015.

[14] S. a. N. P. Russel, Artificial Intelligence A Modern Approach, New Jersey: Pearson Education, Inc, 2010.

[15] M. e. a. Urubkin, "Representation of graphs for storing in," in *E3S Web of Conferences*, Moscow, Russia, 2019.