MEMBANGUN *EDUGAME " BABY ZOO PUZZLE"* BERBASIS ANDROID DENGAN *GAME AGENT* IMPLEMENTASI *FINITE STATE MACHINE*

Ahmad Rofiq Hakim[1)], Reza Andrea[2)], Devian Antoni[3)]

[1]Sistem Informasi, STMIK Widya Cipta Dharma
[2,3]Teknik Informatika, STMIK Widya Cipta Dharma
[1,2,3]Jl. M. Yamin No.25, Samarinda, 75123
E-mail : rofiq_93@yahoo.com[1)], reza@bbirdesign.com[2)], devian.anthony@gmail.com[3)]

ABSTRAK

Dalam pembuatan *edugame "baby zoo puzzle"* berbasis android dengan *game agent* implementasi *finite state machine*. *Edugame "baby zoo puzzle"* menggunakan algoritma pengacakan posisi pada potongan-potongan *puzzle* penggunaan algoritma pengacakan posisi agar potongan *puzzle* tidak mudah dihapal dan pada *game agent* menggunakan metode *finite state machine* (FSM), *game agent* akan memberikan notifikasi atau pemberitahuan pada para pemain. Metode *Finite State Machine* perancangan sistem *control* yang menggambarkan prinsip kerja sistem dengan menggunakan tiga hal : *state* (keadaan), *event* (kejadian), *action* (aksi) yang digunakan kedalam *game agent* dapat memberikan aksi dan reaksi kepada pemain pada saat permainan dimainkan. Hasil dari pembuatan *edugame"baby zoo puzzle"* berupa .Apk dan .Exe yang dapat di jalan *platform* android dan PC, *edugame "baby zoo puzzle"* berbasis android dapat dimainkan untuk semua umur dari anak-anak hingga dewasa.

Kata Kunci: *Edugame, Puzzle, Android, Game agent, Finite State Machine*

1. PENDAHULUAN

Permainan merupakan salah satu yang menarik untuk dikembangkan. Pemanfaatan dan penggunaan *game* edukasi *puzzle* dapat menunjang proses pembelajaran. Pembelajaran dengan menggunakan *game* edukasi *puzzle* dapat mempermudah untuk berpikir, serta anak pun merasa memiliki kesenangan tersendiri, sehingga membutuhkan pemikiran yang lebih besar dapat diasah.

Game edukasi sangat menarik untuk dikembangkan. Ada beberapa kelebihan dari *game* edukasi dibandingkan dengan metode edukasi konvensional. Salah satu kelebihan utama *game* edukasi adalah pada visualisasi dari permasalahan nyata. *Game* edukasi berbasis simulasi didesain untuk mensimulasikan permasalahan yang ada sehingga diperoleh ilmu yang dapat digunakan untuk menyelesaikan permasalahan tersebut. *Game* simulasi dengan tujuan edukasi ini dapat digunakan sebagai salah satu media edukasi yang memiliki pola pembelajaran *learning by doing.* Berdasarkan pola yang dimiliki oleh *game* tersebut, pemain dituntut untuk belajar sehingga dapat menyelesaikan permasalahan yang ada.

Dalam pemaparan di atas maka akan menerapkan *Edugame "Baby Zoo Puzzle"* untuk pengenalan *puzzle* dan mengenal binatang-binatang dalam *game puzzle*. Dimana pemain harus menyusun *puzzle* yang telah di acar posisinya dan harus menyusunnya dengan benar.

Pengunaan algoritma pengacakan posisi agar *game* ini lebih menarik dan tidak membosankan untuk dimainkan oleh anak, memberikan tantangan dalam menyusun potongan-potongan *puzzle. Puzzle* yang sudah diacak tidak dapat mudah diingat oleh pemain.

Game agent akan memberikan aksi, reaksi, mengamati, bertindak pada suatu kondisi. Sistem *game agent* pada permainan ini memberikan notifikasi pemberitahuan yang dibutuhkan pemain terhadap penyusunan potongan *puzzle*. *Game agent* digambarkan dengan bentuk-bentuk binatang yang memberikan pemberitahuan. *Edugame* merupakan salah satu model yang dirasa efektif karena tanpa sadar membuat pemain sedang bermain dan sambil belajar.

2. RUANG LINGKUP PENELITIAN

Dalam penelitian ini permasalahan mencakup:

1. Cakupan Permasalahan
   Dari Berdasarkan latar belakang diatas, maka rumusan masalah dalam penelitian ini yaitu "Bagaimana membangun *Edugame "Baby Zoo Puzzle"* Berbasis Android dengan *Game Agent* Implementasi *Finite State Machine*?".
2. Batasan-batasan penelitian
   Adapun batasan masalah berdasarkan penelitian diatas adalah sebagai berikut:
   1) *Game* diperuntukkan untuk semua umur.
   2) *Game* hanya 10 *level* berupa *chapter* atau *stage* yang harus dilalui oleh pemain
   3) *Game* hanya dijalankan pada *platform Android (Jellybean)* dan PC
   4) Untuk membuat *game* menggunakan *SwishMax4*.
   5) Sistem pergerakan *puzzle* adalah *drag and drop*.
   6) *Content* di dalam *game* ini tidak dapat di *update* secara *online* maupun *offline*
   7) *Game* ini tidak memiliki inputan nama pemain, data pemain terakhir (*history*) yang bermain *game* ini dan tidak memiliki data tertinggi (skor akhir)
   8) *Game agent* menerapkan metode *Finite State Machine* (FSM).

9) Algoritma yang digunakan pada *game* ini yaitu algoritma logika pengacakan posisi yang berfungsi untuk mengacak potongan *puzzle*.

3. BAHAN DAN METODE

Adapun bahan dan metode algoritma yang digunakan dalam membangun aplikasi ini yaitu:

3.1 *Game*

Dalam kamus bahasa Indonesia *game* adalah permainan. Menurut Benny (2013). *Game* merupakan salah satu media hiburan yang popular untuk semua kalangan usia. Sejak pertama kali ditemukan sampai saat sekarang, teknologi *game* telah mengalami kemajuan yang terbilnag pesat. Hal ini di tandai dengan berkembangannya jenis *game*, produk, alat dan jenis interaksi *game* dengan penggunaan yang semakin beragam berbentuknya.

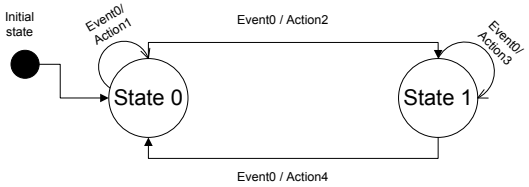
Menurut Zulfadli Fahrul Rozi (2010). *Game* atau permainan adalah sesuatu yang dapat dimainkan dengan aturan tertentu sehingga ada yang menang da nada yang kalah, niasamya dalam konteks tidak serius dengan tujuan *refreshing* atau hiburan.tur.

3.2 *Finite State Machine*

Pengambilan Bahasa formal dapat dipandang sebagai entitas abstrak, yaitu sekumpulan *string-string* simbol alphabet tertentu. Namun bahasa juga dapat dipandang sebagai entitas-entitas abstrak yang dapat dikenali atau dibangkitkan melalui suatu mesin komputasi. Mesin yang dapat mengenali bahasa kelas ini adalah *Finite State Machine*.

Ada beberapa definisi mengenai *Finite State Machine* (FSM) atau sering juga disebut dengan *Finite State Automata* (FSA).

1. FSM didefenisikan sebagai perangkat komputasi yang memiliki *input* berupa *string* dan *output* yang merupakan satu dari dua nilai yang dapat di-*accept* dan *reject*.
2. *Finite Automata* adalah model matematika sistem dengan masukan dan keluaran diskrit. Sistem dapat berada di salah satu dari sejumlah berhingga konfigurasi internal disebut *state*.
3. FSM adalah sebuah metodologi perancangan sistem kontrol yang menggambarkan tingkah laku atau prinsip kerja sistem dengan menggunakan tiga hal berikut: *State* (Keadaan), *Event* (kejadian) dan *action* (aksi). Sistem dapat beralih atau bertransisi menuju *state* lain jika mendapatkan masukan atau *event* tertentu, baik yang berasal dari perangkat luar atau komponen dalam sistemnya itu sendiri. Transisi keadaan ini umumnya juga disertai oleh aksi yang dilakukan oleh sistem ketika menanggapi masukan yang terjadi. Aksi yang dilakukan tersebut dapat berupa aksi yang sederhana atau melibatkan rangkaian proses yang relatif kompleks seperti pada gambar 1.



Gambar 1. Contoh diagram *state* sederhana.

Diagram tersebut memperlihatkan FSM dengan dua buah *state* dan dua buah *input* serta empat buah aksi *output* yang berbeda : seperti terlihat pada gambar, ketika sistem mulai dihidupkan, sistem akan bertransisi menuju *state*0, pada keadaan ini sistem akan menghasilkan *Action*1 jika terjadi masukan *Event*0, sedangkan jika terjadi *Event*1 maka *Action*2 akan dieksekusi kemudian sistem selanjutnya bertransisi ke keadaan *State*1 dan seterusnya.

FSM terdiri dari dua jenis, yaitu FSM ber-*output* dan FSM tidak ber-*output*. FSM tidak ber-*output* digunakan untuk pengenalan bahasa dalam komputer, dengan *input* yang dimasukkan akan diperoleh apakah *input* tersebut dikenal oleh bahasa komputer atau tidak. Salah satu penggunaan FSM tidak ber-*output* adalah program *compiler*, yaitu program untuk memeriksa apakah perintah yang digunakan pengguna benar atau salah. Sementara untuk FSM ber-*output* digunakan untuk merancang mesin atau sistem. Dan FSM yang akan digunakan dalam penelitian ini adalah FSM ber-*output*, dan untuk selanjutnya akan dituliskan dengan FSM saja.

3.3 *Agentt Game*

Menurut Kristanto (2005), dalam kecerdasan buatan, *intelligent agent*t adalah sebuah entitas otonom yang mengamati dan bertindak atas lingkungan, yaitu membutuhkan agent dan mengarahkan aktivitasnya untuk mencapai tujuan yaitu rasional. *Intelligent agent* juga dapat belajar atau menggunakan pengetahuan untuk mereka. Menurut nikila kasabov, agentt intelegent harus menunjukan karakteristik berikut:

1. Mengakomodasi pemecahan masalah baru aturan bertahap.
2. Beradaptasi *online* dan *real time*
3. Mampu menganalisis sendiri dalam hal perilaku, kesalahan dan kesuksesan.
4. Belajar dan meningkatkan melalui interaksi dengan lingkungan (perwujudan).
5. Belajar dengan cepat dari sejumlah besar data.

Dari karakteristik ini dapat di simpulkan bahwa agent adalah suatu fungsi terlihat seperti sendiri dalam menentukan suatu tindakan atau memberi keputusan dalam menjelankan fungsinya dalam seuah entitas otonom yang mengamati dan bertindak atas lingkungannya.

3.4 *Puzzle*

Menurut Hidayat (2013), kata *puzzle* berasal dari bahasa inggris yang berarti teka-teki atau bongkar pasang, media *puzzle* merupkana media sederhana yang dimainkan dengan bongkar pasang.

Puzzle adalah suatu masalah dimana dapat membuat orang yang mencoba memmecahkannya berpikir. *Puzzle* digunakan untuk hiburan dan dapat membantu kemampuaan logika. Pemain mungkin perlu untuk mengenali pola-pola untuk mengatasinya,sehingga

mereka yang memiliki logika yang baik dengan *puzzle*. Bermain *puzzle* dapat mengasah otak kanan para pemainnya. *Puzzle* terdiri atas *cube puzzle, Sudoku puzzle, jigsaw puzzle*, dan banyak lagi.

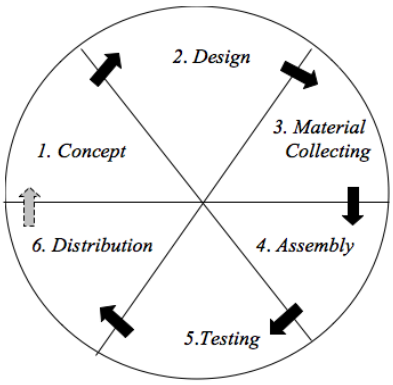
3.5 Android

Menurut Safaat (2012), *Android* merupakan subset perangkat lunak untuk perangkat *mobile* yang meliputi sistem operasi, *middleware* dan aplikasi inti yang di-*release* oleh *Google*. Sedangkan *Android SDK (Software Development Kit)* menyediakan *Tools* dan API (*Aplication Programing Interface*) yang diperlukan untuk mengembangkan aplikasi pada *platform Android* dengan menggunakan bahasa pemrograman *Java*.

Dikembangkan bersama antara *Google*, *HTC*, *Intel*, *Motorola*, *Qualcomm*, *T-Mobile*, *NVIDIA* yang tergabung dalam *OHA* (*Open Handset Alliance*) dengan tujuan membuat sebuah standar terbuka untuk perangkat bergerak (*mobile device*).

3.6 Tahapan Pengembangan Multimedia

Menurut Menurut Binanto (2010), metodologi pengembangan multimedia terdiri dari enam tahap, yaitu *concept* (pengonsepan), *design* (pendesainan), *meterial collecting* (pengumpulan materi), *assembly* (pembuatan), *testing* (pengujian), dan *distribution* (pendistribusian). Keenam tahap ini tidak dapat bertukar posisi. Meskipun begitu, tahap *concept* memang harus menjadi hal yang pertama kali dikerjakan.



Gambar 2. Tahapan Pengembangan Multimedia

1. *Concept*

   Tahapan *concept* (pengonsepan) adalah tahap untuk menentukan tujuan dan siapa pengguna program (indentifikasi audiens). Tujuan dan penguna akhir program berpengaruh pada nuansa multimedia sebagai pencerminan dari indentitas organisasi yang menginginkan informasi sampai pada pengguna akhir. Karakteristik pengguna termasuk kemampuan pengguna juga perlu dipertimbangkan karena dapat memengaruhi pembuatan desain.

   Selain itu, tahap ini juga akan menentukan jenis aplikasi (presentasi, interaktif, dan lain-lain) dan tujuan aplikasi (hiburan, pelatihan, pembelajaran dan lain-lain). Dasar aturan untuk perancangan juga ditentukan pada tahap ini, misalnya ukuran aplikasi, target, dan lain-lain. Output dari tahap ini biasanya berupa dokumen yang bersifat naratif untuk mengungkapkan tujuan projek yang ingin dicapai.

2. *Design*

   Design (perancangan) adalah tahap pembuatan spesifikasi mengenai arsitektur program, gaya, tampilan, dan kebutuhan material/bahan untuk program. Spesifikasi dibuat serinci mungkin sehingga pada tahap berikutnya, yaitu material *collecting* dan *assembly,* pengambil keputusan baru tidak diperlukan lagi, cukup menggunakan keputusan yang sudah ditentukan pada tahap ini. Meskipun demikian, pada prakteknya, pekerjaan proyek pada tahap awal masih akan sering mengalami penambahan bahan atau pengurangan bagian aplikasi, atau perubahan-perubahan lain.

3. *Material Collecting*

   Material Collecting adalah tahap pengumpulan bahan yang sesaui dengan kebutuhan yang dikerjakan. Bahan-bahan tersebut, antara lain gambar *clip art*, foto, animasi, *video, audio*, dan lain-lain yang dapat diperoleh secara geratis atau dengan pemesanan kepada pihak lain sesuai dengan rancangannya. Tahap ini dapat dikerjakan secara paralel dengan tahap *assembly*. Namun, pada beberapa kasus, tahap material *collecting* dan tahap *assembly* akan dikerjakan secara linear dan tidak paralel.

4. *Assembly*

   Tahap *Assembly* adalah tahap pembuatan semua objek atau bahan multimedia. Pembuatan aplikasi didasarkan pada tahap *design,* seperti *storyboard,* bagan alir, dan /atau struktur navigasi.

5. *Testing*

   Tahap *Testing* (pengujian) dilakukan setelah menyelesaikan tahap pembuatan (*assembly*) dengan menjalankan aplikasi/program dan melihatnya apakah ada kesalahan atau tidak. Tahap pertama pada tahap ini disebut tahap pengujian *alpha* (*alpha test*) yang pengujiannya dilakukan oleh pembuat atau lingkungan pembuatnya sendiri. Setelah lolos dari pengujian *alpha,* pengujian *beta* yang melibatkan penggunaan akhir akan di lakukan.

6. *Distribution*

   Pada tahap ini, aplikasi akan disimpan dalam suatu media penyimpanan. Jika media penyimpanan tidak cukup untuk menampung aplikasinya, komprensi terhadap aplikasi tersebut akan dilakukan. Tahap ini juga dapat disebut tahap evaluasi untuk pengembangan produk yang sudah jadi supaya menjadi lebih baik. Hasil evaluasi ini dapat

digunakan sebagai masukkan untuk tahap *concept* pada produk selajutnya informasi.

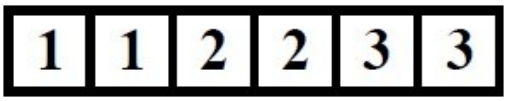
3.7 Algoritma Pengacakan Posisi

Menurut Reza Andrea, (2015), *Shuffle random* adalah pengacakan urutan *indeks* dari sebuah *record* atau *array*. Pengacakan ini diibaratkan pengocokan pada dek kartu, dimana semua kartu dikocok sehingga susunannya teracak [4]. Contoh lain misalkan A adalah *array* 5 x 1, A = [1 2 3 4 5] maka proses *shuffle random* akan mengacak susunan indek dari *array* A menjadi A1 = [5 1 3 2 4] ataupun menjadi susunan *array* yang lain. Dalam bahasa pemrograman fungsi *shuffle random* tidak hanya dapat mengacak angka, tetapi juga dapat mengacak *array string* ataupun campuran *string* dan angka.

Untuk menerapkan pengacakan posisi gambar dengan sistem *shuffle random* dilakukan dengan beberapa tahap berikut ini:

1. Menentukan jumlah pasangan gambar

Awal dari penerapan yaitu menentukan jumlah *puzzle* pasangan gambar dan menyusunnya layaknya sebuah matrik seperti pada gambar 3.



Gambar 3. Matrik 6 x 1 Permainan *Match-up* angka yang belum teracak

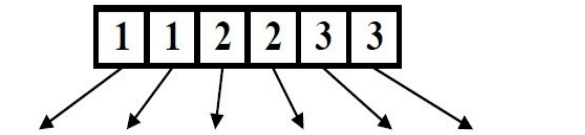
Pada tahap ini dilakukan deklarasi nilai *array* seperti pada contoh *script* dibawah ini:

```
A = new Array(0,1,2,3,4,5)
```

Dimana nilai *indeks array* yang pertama (*indeks* ke-0) adalah 0, dan *indeks* terakhir adalah 5

2. Mencatat setiap koordinat x dan y dari setiap *puzzle* gambar

Setiap objek gambar atau *shape* dalam *project board* permainan pasti memiliki koordinat *x* dan *y* seperti pada gambar 4.



Gambar 4. Koordinat *x* dan *y* dari 6 kotak gambar

Pada tahap ini keenam koordinat kotak gambar di catat dalam sebuah prosedur

```
Procedure daftar_posisi()

if (posisi = 0) then
    x ← 100
    y ← 150
Else if (posisi = 1) then
    x ← 100
    y ← 200
Else if (posisi = 2) then
    x ← 100
    y ← 250
Else if (posisi = 3) then
    x ← 100
    y ← 300
Else if (posisi = 4) then
    x ← 100
    y ← 350
Else
    x ← 100
    y ← 400
  End If
End Procedure
```

Dapat dilihat pada prosedur di atas, variabel posisi dimulai pada kondisi pada saat nilai posisi adalah 0, nilai 0 menunjukkan nilai indek pertama dari *array*

3. Pengkodean pengacakan posisi

Tahap terakhir adalah penggunaan fungsi *shuffle random*, serta pengacakan posisi koordinat dari setiap kotak *puzzle* gambar, dimana setiap pasangan kotak diberi nama kotak_1, kotak_2, dan kotak_3

```
A ← random.shuffle(A)

posisi ← A[0]
daftar_posisi()
kotak_1a._x ← x
kotak_1a._y ← y

posisi ← A[1]
daftar_posisi()
kotak_1b._x ← x
kotak_1b._y ← y

posisi ← A[2]
daftar_posisi()
kotak_2a._x ← x
kotak_2a._y ← y

posisi ← A[3]
daftar_posisi()
kotak_2b._x ← x
kotak_2b._y ← y
posisi ← A[4]
daftar_posisi()
kotak_3a._x ← x
kotak_3a._y ← y

posisi ← A[5]
daftar_posisi()
kotak_3b._x ← x
kotak_3b._y ← y
```

Dapat dipahami dari algoritma teks di atas, terdapat 2 kotak_1 (kotak_1a dan kotak_1b), hal ini menjelaskan

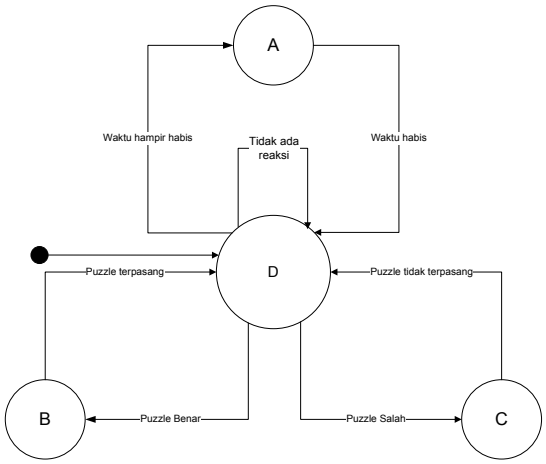
bahwa ada 2 kotak yang memiliki gambar atau angka yang sama (ada 2 kotak bernomer 1), begitu pula pada kotak_2 dan 3.

4 RANCANGAN SISTEM/APLIKASI

Perancangan *game agent Baby Zoo Puzzle* menggunakan *Unified Modeling Language* (*UML*)*:*

4.1 Perancangan FSM (*Finite State Machine*)

FSM adalah sebuah perancangan sistem control yang menggunakan tiga hal yaitu *state, event, action.* FSM ini di gunakan untuk menggambarkan tingkah laku atau prinsip kerja *game agent.*



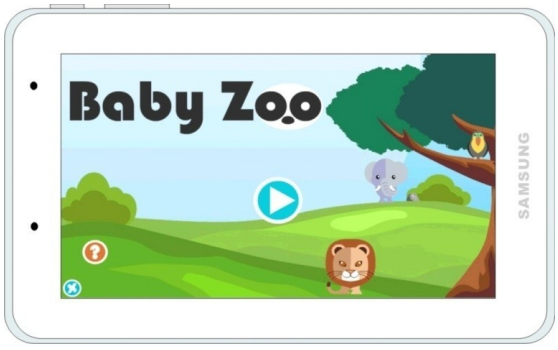
Gambar 5. Rancangan FSM (*Finite State Machine*).

5 IMPLEMENTASI

Hasil implementasi berdasarkan analisis dan perancangan adalah sebagai berikut:

5.1 Scene Menu Utama

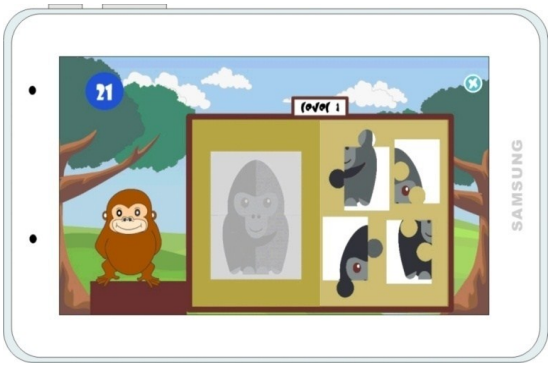
Pada gambar 6 *scene* menu utama merupakan tampilan utama dari *edugame* disaat pemain baru memasuki sistem. Terdapat nama atau judul dari *Edugame "Baby Zoo Puzzle*". Setiap tombol pada *scene* menu utama memiliki fungsi masing-masing yang menghubungkan satu *scene* dengan *scene* lainnya. Tampilan *scene* menu utama.



Gambar 6. *Scene* Menu Utama

5.2 *Scene* Bermain *level* 1

Pada gambar 7 *Scene* menu Bermain *level* 1 adalah *Scene* dimana pemain harus menyusun dan menyelsaikan permainan yang terdiri dari 4 *Puzzle* dengan batasan waktu bermain 25 detik.



Gambar 7. *Scene* Bermain *level* 1

5.3 *Scene* Menu Bermain *Level* 9

Pada gambar 8 *Scene* menu Bermain *level* 9 adalah *Scene* dimana pemain harus menyusun gambar dan menyelsaikan permainan yang terdiri dari 16 *puzzle* dengan batasan waktu bermain 30 detik.

Gambar 8. *Scene* menu bermain *level* 9

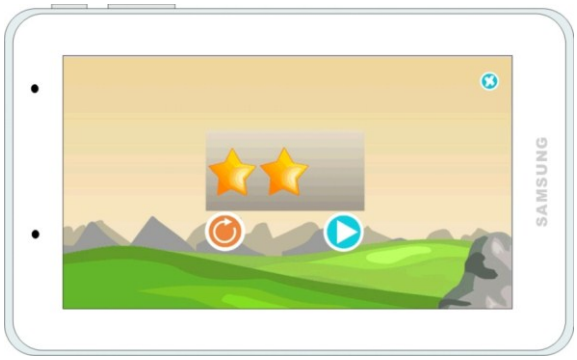
5.4. *Scene* Menang (Menampilkan Bintang)

Scene mendapatkan bintang merupakan *scene* yang tampil disaat pemain berhasil memenangkan suatu *level*, pada *scene* ini ditampilkan skor yang diperoleh oleh pemain saat ia menyelesaikan *level* tersebut, jika pemain dapat menyelesaikan permainan dengan waktu yang cepat makan pemainakan memperoleh skor yang lebih tinggi. Terdapat tiga bintang pada *scene* untuk menggambarkan seberapa cepat pemain dapat meneyelesaikan permainan ini, pemain bisa mendapatkan 3 bintang dan sebanyak 100 skor apabila pemain dapat menyelesaikan dalam waktu 10 detik pertama dari 25 – 15 detik, pemain bisa mendapatkan 2 bintang dan skor sebanyak 75 skor apabila pemain dapat menyelesaikan permainan dalam 10 detik kedua dari 14 – 5 detik dan minimal 1 bintang banyak skor 50 apabila pemain dapat menyelesaikan dalam waktu 4 detik terakhir. Dengan kondisi yang dijelaskan sebagai berikut:

```
if waktu ≤ 25 Then bintang = 3 and skor = 100
```

```
Else if waktu > 26 and waktu ≤ 35 Then
bintang = 2 and skor = 75
Else Then bintang = 1 and skor = 50
```

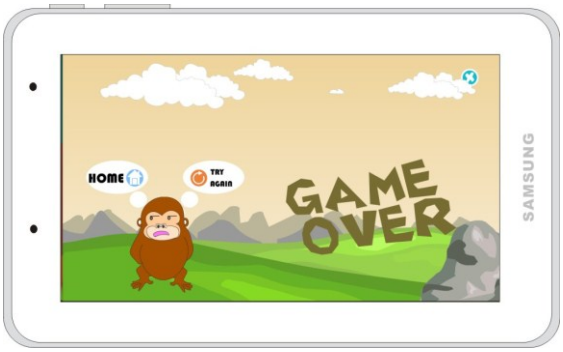
Pada *Scene* ini terdapat 2 tombol utama yang memiliki fungsi masing-masing. Tombol "Lanjut" digunakan pemain untuk lanjut ke *level* berikutnya "Main Lagi" digunakan pemain untuk mengulang di *level* yang sama.Tampilan *scene* menang dapat dilihat pada gambar 9.



Gambar 9. *Scene* Menang (menampilkan bintang)

5.5 *Scene* Kalah

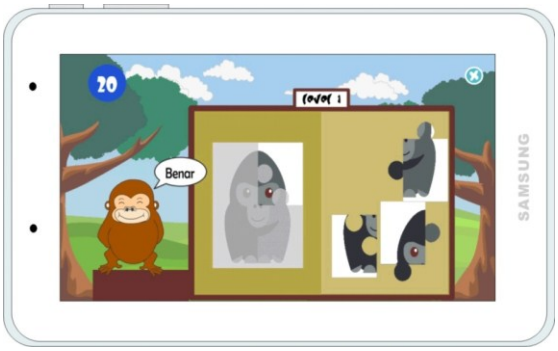
Pada *scene* kalah akan terdapat gambar karakter telah gagal menyelesaikan suatu *level* permainan. Pada *scene* ini terdapat 2 tombol utama. Tombol "*try again*" digunakan untuk mengulang dan mencoba kembali di *level* yang sama, dan tombol "*home*" digunakan untuk kembali ke *Scene* Menu .Tampilan *scene* kalah dapat dilihat pada gambar 10.



Gambar 10. *Scene* Kalah

5.6 *Game Agent* Susunan *Puzzle* Benar

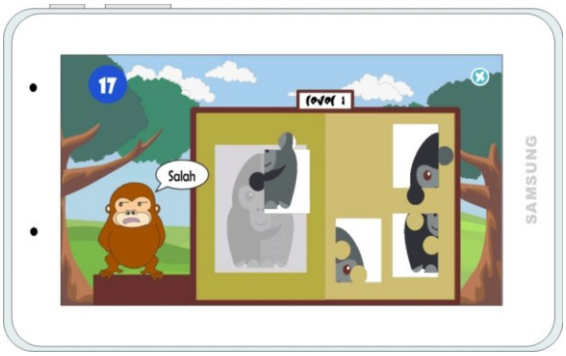
Tampilan dari *game agent* memberikan petunjuk untuk pemain bahwa *puzzle* yang di susun benar.



Gambar 11. Tampilan dari *game agent* susunan *puzzle* benar

5.7 *Form* Hasil Perhitungan

tampilan dari *game agent* meberikan petunjuk untuk pemain bahwa *puzzle* yang di susun salah.



Gambar 12. Tampilan dari *game agent* susunan *puzzle* salah

6. KESIMPULAN

Dengan adanya hasil penelitian yang dilakukan dan Berdasarkan uraian dari masing-masing bab dan hasil pembahasan maka dapat disimpulkan bahwa Membangun *Edugame "Baby Zoo Puzzle"* berbasis *android* dengan *game agent* implementasi *Finite State Machine* sebagai berikut :

1. Pembuatan *edugame "baby zoo puzzle"* berbasis *android* melalui beberapa proses menggunakan program *swishmax4* dan *eclipse*, setelah itu menjadi sebuah *file* yang dapat dimainkan di *platform android* dan PC.
2. *Edugame "baby zoo puzzle"* berbasis *android* dapat dimainkan untuk semua umur dari anak-anak hingga dewasa.

7. SARAN

Berdasarkan hasil dari penelitian ini ada beberapa saran, yaitu sebagai berikut:

1. Diperlukan perbaikan dan penambahan fitur baru agar lebih menarik.
2. Permainan ini berjalan di *platform android jellybean*, diharapkan *game* ini dapat berjalan di *android kitkat* dan *lollipop*.
3. Memperbanyak ekspresi pada *game agent* dan agar dapat lebih berinteraksi pada *user*.

8. DAFTAR PUSTAKA

Adnyana, MA. 2011. *Modul SwishMAx* .http://ilmukomputer.org/2008/11/25/animasi-flash-dengan-swishmax-2/. (Diakses pada tanggal 23 januari 2015)

Alamsyah, R.Yadi Rakhman. Nurhakim, Bhilly. *Game Simulasi Pengelolaan Tanaman Di Indonesia Berbasis Desktop. Bandung : STMIK LPKIA.*

Andrea, Reza, 2012, *Teknik Pengacakan Posisi – Find Me The Game Prosiding Senaik 2012*, Samarinda: Unmul Press.

Anung Rachman, Vincent Suhartono, Yuliman Purwanto, 2010, *Agent cerdas animasi wajah untuk game tebak kata, Semarang: Universitas Dian Nuswantoro.*

Binanto, Iwan, 2010, *Multimedia Digital Dasar Teoridan Pengembangan,* Yogyakarta: Andi.

Bambang, Warsita, 2008, *Teknologi Pembelajaran dan Aplikasinya*, Jakarta: Rineka Cipta.

Brownlee, J. 2011. Finite State Machine in Game. <url,http://ai-depot.com/Finite State Machine (FSM) html>. (Diakses 25 maret 2015)

Elliani. 2014. *Membangun edugame Smart & Fun Hijayah berbasis android*. Skripsi S1 Teknik Informatika, Samarinda : STMIK WICIDA.

Ladjamudin, Al-bahra, 2005, *Analisis dan Desain Sistem Informasi.* Yogyakarta: Garaha Ilmu.

Hidayat, Febri. 2013. *Perancangan game edukasi puzzle sebagai media pembelajaran untuk anaku siadini (studi kasus di tk. Pertiwi gebang-cirebon*. http://elib.unikom.ac.id/. (Diakses tanggal 23 januari 2014)

Madcoms. 2008. *Adobe Flash CS3 Professional* , Madiun: CV ANDI OFFSET.

Madcoms, 2008, *Mahir dalam 7 hari CorelDRAW X4,* Yogyakarta: Andi.

Madcoms, 2005, *Mahir Dalam 7 Hari Adobe Photoshop CS*, Yogyakarta : Andi offset.

Nedya Matahari Bhakti. 2012. *Pemetaan Perilaku Non-Playble Character Pada Permainan Berbasisi Role Playning Game Menggunakan Metode Finite State Machine,* Yokyakarta.

Nugroho, Adi, 2005, *Fokus Bangun Dasar Perancangan Sistem Dengan UML*, Yogyakarta : Gramedia Pustaka Utama.

Pressman, Roger, 2010, *Rekayasa Perangkat Lunak*. Yogyakarta : Andi Offset.

Suyanto, 2005, *Multimedia Alat Untuk Meningkatkan Keunggulan Multimedia,* Yogyakarta: Andi.

Safaat, Nasrudin H., 2012, *Pemrograman Aplikasi Mobile Smartphone dan Tablet PC berbasis Android,* Yogyakarta: Andi.