BAB II

LANDASAN TEORI

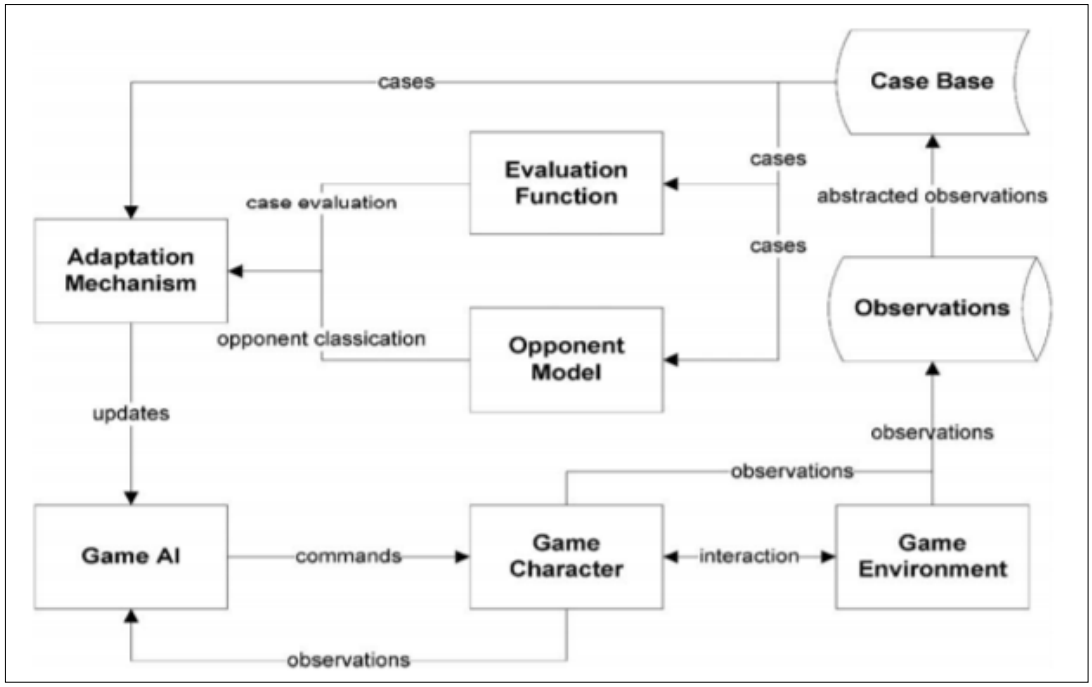
2.1 Adaptive AI

Adaptive AI mengacu kepada *non-player character* (NPC) dinamis di mana komputer mampu mengubah *game behavior*-nya dalam merespon musuhnya, baik ketika sesi bermain *game*, atau diantara sesi tersebut. Dalam kasus tertentu, penskalaan tingkat kesulitan pada game dinamis menggunakan *adaptive AI* untuk secara otomatis mengadaptasi parameter game dan *behavior* pada kondisi real time menurut tingkat keahlian pemain pada game (Tan, et al., 2011).

Menurut **(Satrio, et al., 2014)** efek dari implementasi *adaptive AI* pada *game* adalah:

1. Meningkatkan pengalaman bermain, karena *adaptive AI* dapat beradaptasi ke tiap individu untuk menyesuaikan cara bermain, dan tujuam pemain memainkan *game* tersebut.
2. Mengurangi waktu dan biaya pengembangan jika sebuah *game* mampu beradaptasi sendiri, pihak pengembang *game* membutuhkan usaha yang lebih sedikit untuk memprediksi segala kemungkinan yang terjadi.

Salah satu contoh pendekatan dari *adaptive AI* adalah *case-based adaptive game AI*, yang merupakan pendekatan pada *game AI* di mana *domain knowledge* dikumpulkan secara otomatis oleh *game AI*, dan dieksploitasi secara langsung untuk menghasilkan tingkah laku yang efektif (Bakkes, et al., 2009). *Case-based adaptive game AI* biasanya bagus digunakan pada *game* yang memiliki akses internet untuk menyimpan dan mengambil sampel dari *gamplay experiences*. Pendekatan ini dapat dilihat dari gambar 2.1.



Gambar 2.1 Case-Based Adaptive Game AI

2.2 Game Design

Game design merupakan deskripsi berbasis fitur dari produk akhir yang dapat digunakan sebagai visi kreatif bersama dalam semua *game development team*. Seiring dengan progress *development* dan perubahan menjadi wajib. Hal ini merupakan tugas *game designer* untuk mengembangkan *game design* (Rollings & Morris, 2004).

Dalam membuat *game design* ada beberapa hal penting yang harus diperhatikan. Hal-hal penting tersebut dapat dilihat dalam beberapa pertanyaan berikut (Rollings & Morris, 2004).

1. Is it original?

   Maksud dari *original* disini maksudnya bukan berarti apakah *game* yang di *design* merupakan *game* yang benar-benar baru, karena hanya sedikit *game* yang benar-benar merupakan *game* baru. Maksud dari *original* disini adalah apakah *game* yang di *design* memiliki fitur yang membedakannya dengan *game* lain.

2. Is it coherent?

   Maksudnya adalah apakah fitur yang akan diberikan pada game dapat mendukung *core vision* dari *game*. Kare *game design* yang baik harus dibangun berdasarkan *core vision* dari *game* tersebut.

3. Is it interactive?

   Kebanyakan dari *computer game* yang dibuat sekarang menggunakan *high interactivity*, karena pemain memiliki peran yang *proactive*. Cerita yang ada dalam *game* harus terungkap langsung dari apa yang dilihat dan dilakukan

oleh pemain, karena pemain memiliki ekspektasi bahwa mereka memiliki peran yang *proactive*.

4. Is it interesting?

   Tidak semua hal yang terjadi dalam game harus merupakan bagian dari *gameplay*. Akan tetapi semua hal yang terjadi dalam game haruslah *interesting*.

5. Is it fun?

   Maksudnya adalah apakah semua yang ditambahkan di dalam game dapat meningkatkan *enjoyment* dari pemain

2.3 Game Architecture

Game architecture adalah titik pertemuan yang menggabungkan sisih artistik dalam *game development* dengan kebutuhan teknisnya. Membayangkan desain sebagai pandangan ideal bagaimana game saharusnya dibuat. Sisih arsitekturlah yang mencari cara untuk mencapai pandangan ideal tersebut (Rollings & Morris, 2004).

Menurut (Rollings & Morris, 2004) *game architecture* yang sempurna harus harus mencapai semua yang berikut ini:

1. Modularity

   Memisahkan projek ke dalam modul yang terenkapsulasi seluruhnya sehingga memungkinkan setiap modul di test, dimodifikasi, dan diganti secara independent tanpa merusak sistem

2. Reusability

   Modul harus di desain menjadi *extensible* dan *portable* sehingga dapat diimplementasikan ke dalam projek lain

4. Robustness

   Reliabilitas yang tinggi dapat paling baik tercapai dengan membangun arsitektur terbebas dari *module interdependencies* (saling ketergantungan).

5. Trackability

   Rancangan projek berasal secara langsung dari arsitektur, memiliki jadwal yang memungkinkan pengukuran kemajuan yang realistis

2.4 Game Design Document

Game design document (GDD) merupakan dokumen yang merinci semua karakter, level, *game mechanics*, *views*, *menus*, dan sebagainya dalam sebuah *game* (Bethke, 2003). Tujuan utama dari GDD adalah sebagai *instruction manual* untuk membangun *game*, dan bukan sebagai petunjuk cara memainkan *game*. Hal ini dikarenakan GDD ditujukan untuk tim pengembang dan bukan untuk pemain (Oxland, 2004).

Menurut (Gonzalez, 2016), GDD terdiri dari beberapa bagian, yaitu:

1. Characters

   Bagian ini menjelaskan karakter-karakter yang ada dalam *game*, seperti karakter utama, karakter lawan, karakter pendukung, dan sebagainya.

2. Story

   Bagian ini menjelaskan cerita dan *event* yang terjadi di dalam *game*

3. Story Progression

Bagian ini menjelaskan perkembangan dari alur cerita seiring dengan berjalannya *game*

4. Gameplay

Bagian ini merupakan salah satu bagian yang terpenting di dalam GDD. Pada bagian ini, akan dijelaskan *gameplay* dari game yang dibuat. Karena bagian ini besar, bagian ini dapat terbagi menjadi beberapa *subsection*, sebagai berikut:

a. Goals

Bagian ini menjelaskan objektif yang ingin dicapai oleh pemain dalam *game*.

b. User Skills

Bagian ini menjelasakan kemampuan yang harus dikuasai pemain untuk memainkan *game.*

c. Game Mechanics

Bagian ini menjelaskan bagaimana seharusnya *game* bekerja dan berbagai mekanisme yang ada di dalam *game.*

d. Progression and challenge

Bagian ini merupakan bagian yang opsonal. Pada bagian ini, akan dijelaskan bagaimana tingkat kesulitan dari permainan akan bertambah selama berjalannya *game*.

e. Losing

Bagian ini menjelaskan kondisi-kondisi yang akan menyebabkan pemain kalah

5. Art Style

   Bagian ini menjelaskan tampilan *art* yang digunakan dalam *game*, seperti *asset sprites*, tampilan *button*, dan sebagainya

6. Music and Sounds

   Bagian ini menjelaskan music dan *sound effect* yang digunakan dalam game

7. Technical Description

   Bagian ini mendeskripsikan *platform* yang dapat digunakan untuk menjalankan *game*. Selain itu, pada bagiani ini akan dijelaskan *tools* apa saja yang digunakan atau ingin digunakan selama pengembangan *game*.

8. Marketing & Funding

   Bagian ini merupakan bagian yang sangat opsional. Pada bagian ini akan dijelaskan mengenai cara mendaptakan dana untuk mengembangkan game. Selain itu, pada bagian ini juga akan dijelaskan mengenai teknkik pemasaran yang akan digunakan untuk menjual game.

9. Other Ideas

   Bagian ini merupakan bagian yang sangat opsional. Bagian ini berfungsi untuk menyimpan ide-ide yang tidak diyakinkan harus dimasukan ke dalam *game* atau tidak.

2.5 Fighting Game

Fighting game adalah tipe dari *action game* di mana 2 (kadang lebih) karakter bertarung satu sama lain. Game jenis ini biasanya menyediakan Gerakan khusus yang dapat diaktifkan dengan penekanan kombinasi tombol dan pergerakan *joystick* tertentu. *Fighting game* biasanya memperlihatkan petarung dari sisih samping, walaupun genre game ini telah berkembang dari grafik 2 dimensi menjadi 3 dimensi (Rollings & Adams, 2006).

2.6 Unity

Unity adalah *game engine cross-platform* yang dikembangkan oleh Unity Technologies, pertama diumumkan dan diluncurkan Juni 2005 pada Apple Inc.'s Worldwide Developers Conference sebagai *game engine* eksklusif untuk Mac OS X. Unity dapat digunakan untuk membuat game 3 dimensi, 2 dimensi, *virtual reality*, *augmented reality*, simulasi dan sebagainya (Axon, 2016).

2.7 Online Machine Learning

Unsupervised online Machine learning pada *game AI* merupakan Teknik yang mengharuskan proses *automatic learning* teraplikasikan pada *game* AI saat game sedang dimainkan (Spronck, et al., 2004). pengaplikasian teknik *online machine learning* pada *game* akan membuat game *AI* dapat beradaptasi dengan bagaimana *game* dimainkan oleh pemain. Oleh karena itu, dengan digunakannya *online machine learning*, *AI* pada *game* dapat menjadi *adaptive* (Spronck, et al., 2006).

Menurut (Spronck, et al., 2006), ada dua cara yang dapat digunakan dalam mengaplikasikan teknik *online machine learning* pada *game AI*, yaitu:

1. *Human-controlled*

   Human-controlled online learning mengimplementasikan perubahan pada *game AI* dengan memproses *feedback* langsung yang diberikan oleh pemain dari semua keputusan yang diambil oleh *AI*. *Feedback* tersebut mengindikasikan apakah keputusan yang diambil *AI* tersebut diinginkan atau tidak.

2. *Computer-controlled*

   Computer-controlled online learning mengimplementasikan perubahan otomatis pada game dengan memproses observasi terhadap efek dari *game AI* selama game berjalan secara langsung.

2.8 Noedify

Noedify merupakan *framework neural network* untuk *game engine* unity yang memiliki *interface* tingkat tinggi dan digunakan untuk membangun dan melatih model *neural network* (TINYANGLELABS, 2020). Fitur yang disediakan oleh *plugin* noedify adalah sebagai berikut:

1. Membangun *deep neural network* dalam projek unity saat *runtime*
2. Melatih *full-connected* atau 2D *convolutional neural network*
3. Menggunakan *job system* unity untuk melakukan *training* di *background*.
4. Mengimport model dari TensorFlow keras untuk digunakan dalam projek unity.

2.9 Game User Experience Satisfaction Scale

Game user experience satisfaction scale (GUESS) merupakan sebuah metode yang digunakan untuk mengukur tingkat kepuasan pemain terhadap sebuah *game*. Metode GUESS memiliki 55 pertanyaan yang terbagi ke dalam 9 *subclas*, yaitu *usability* / *playability*, *narratives*, *play engrossment*, *enjoyment*, *creative freedom*, *audio aesthetics*, *personal gratification*, *social connectivity*, and *visual aesthetic* (Phan, et al., 2016).

Jika game yang dievaluasi menggunakan GUESS tidak memiliki komponen *narratives* atau *social connectivity*, maka *subscales narratives* atau *social connectivity* dapat dihilangkan dari kuesioner (Phan, et al., 2016). Dalam penghitungan skor menggunakan GUESS, direkomendasikan untuk merata-ratakan *rating* dari setiap *item* agar dapat memperoleh nilai rata-rata dari setiap *subscales*. Selanjutnya, nilai rata-rata dari setiap *subscales* dapat dijumlahkan untuk mendapatkan skor gabungan dari *video game satisfaction* (Phan, et al., 2016).

Pertanyaan dari GUESS akan diukur menggunakan skala *likert* 7 poin. Nilai terendah adalah 1 yang berarti *strongly disagree* dan nilai tertinggi adalah 7 yang berarti *strongly agree*. Untuk mengukur nilai interval dari skala *likert*, akan digunakan rumus sebagai berikut.

Interval = 100 / *total scale* (*likert*)

Berdasarkan rumus interval skala *likert*, akan diperoleh skala interval yang dapat dilihat dari tabel dibawah.

Tabel 2.1 Interval Skala Likert

Interval	**Keterangan**
0% – 14.285%	Sangat Tidak Puas
14.286% - 28.570%	Tidak Puas
28.571% - 42.855%	Cukup Tidak Puas
42.856% - 57.140%	Netral
57.141% - 71.425%	Cukup Puas
71.426% - 85.710%	Puas
85.711% - 100%	Sangat Puas

Tabel 2.1 merupakan tabel yang menunjukan interval tingkat kepuasan pemain dalam skala *likert.* Untuk mendapatkan skor dari *video game satisfaction,* akan digunakan rumus sebagai berikut.

1. Rumus Pertanyaan Positif

Result = (*strongly disagree* x 1 + *disagree* x 2 + *somewhat disagree* x 3

+ *neutral* x 4 + *somewhat agree* x 5 + *agree* x 6 + *strongly agree* x 7)

÷ (*total sample* x *total scale*) x 100

2. Rumus Pertanyaan Negatif

Result = (*strongly disagree* x 7 + *disagree* x 6 + *somewhat disagree* x 5

+ *neutral* x 4 + *somewhat agree* x 3 + *agree* x 2 + *strongly agree* x 1)

÷ (*total sample* x *total scale*) x 100