MJI

Media Jurnal Informatika
Vol. 16, no.2, Desember 2024
http://jurnal.unsur.ac.id/mjinformatika

p-issn : 2088-2114
e-issn : 2477-2542

Implementasi Finite State Automata pada Desain Sistem Dialog Interaktif Non-Player Character dalam Game "Legenda Guardian Kuno"

Firas Atqiya[1], Muhammad Rizqi Sholahuddin[2], Ririn Suharsih[3]
[1,3]Universitas Muhammadiyah Bandung, [2]Politeknik Negeri Bandung
firasatqiya@umbandung.ac.id[1], muhammad.rizqi@polban.ac.id[2], ririnsuharsih@umbandung.ac.id[3]

Abstract

Interaction between players and Non-Player Characters (NPCs) is a crucial element in gaming experience, especially in the role-playing game (RPG) genre. An adaptive and responsive dialogue system can enhance immersion and player engagement. This research aims to implement the concept of Finite State Automata (FSA) in the NPC dialogue system of the game "Legenda Guardian Kuno" to improve interaction quality and conversation flexibility. An FSA model was developed and integrated with dialogue data in JSON format, which facilitates the management and maintenance of the dialogue system. The research method included designing an FSA model that represents the conversation flow, implementing program code using Python with dialogue data stored in JSON format, as well as conducting functionality testing and system validation. The results showed that the use of FSA is effective in managing branching conversations, enabling NPCs to provide dynamic and contextual responses based on player choices, thus increasing immersion and player engagement. The developed system is able to accommodate various complex dialogue paths, ensuring that the storyline is logical and consistent. This research shows that FSA is an effective approach in developing interactive dialogue systems.
Keywords: Finite State Automata, Dialogue System, Non-Player Character, Game Development

Abstrak

Interaksi antara pemain dan Non-Player Character (NPC) merupakan elemen kunci dalam pengalaman bermain game, terutama dalam genre role-playing game (RPG). Sistem dialog yang adaptif dan responsif dapat meningkatkan imersi dan keterlibatan pemain. Penelitian ini bertujuan untuk mengimplementasikan konsep Finite State Automata (FSA) dalam sistem dialog NPC pada game "Legenda Guardian Kuno" guna meningkatkan kualitas interaksi dan fleksibilitas percakapan. Model FSA dikembangkan dan diintegrasikan dengan data dialog dalam format JSON, yang memudahkan pengelolaan dan pemeliharaan sistem dialog. Metode penelitian meliputi perancangan model FSA yang merepresentasikan alur percakapan, implementasi kode program menggunakan bahasa Python dengan data dialog disimpan dalam format JSON, serta pengujian fungsionalitas dan validasi sistem. Hasil penelitian menunjukkan bahwa penggunaan FSA efektif dalam mengelola percakapan bercabang, memungkinkan NPC memberikan respons dinamis dan kontekstual berdasarkan pilihan pemain, sehingga meningkatkan imersi dan keterlibatan pemain. Sistem yang dikembangkan mampu mengakomodasi berbagai jalur percakapan yang kompleks, memastikan alur cerita berjalan logis dan konsisten. Penelitian ini menunjukkan bahwa FSA merupakan pendekatan yang efektif dalam pengembangan sistem dialog interaktif.
Kata kunci: Finite State Automata, Sistem Dialog, Non-Player Character, Game Development

I. PENDAHULUAN

Interaksi antara pemain dan Non-Player Character (NPC) merupakan elemen krusial dalam pengembangan permainan video modern. NPC berfungsi sebagai media untuk menyampaikan cerita, memberikan misi, dan memperkaya pengalaman bermain. Dialog yang dinamis dan responsif antara NPC dan pemain dapat meningkatkan imersi, dan mendorong keterlibatan emosional pemain terhadap alur cerita yang disajikan [1], [2].

Perancangan desain sistem dialog NPC yang fleksibel dan adaptif merupakan hal yang kompleks di mana memerlukan pendekatan terstruktur dan efisien. Dalam konteks ini, Finite State Automata (FSA) dapat digunakan untuk memodelkan dan mengelola sistem dialog dalam game.

FSA adalah model matematika yang merepresentasikan mesin dengan jumlah state terbatas, di mana transisi antar state ditentukan oleh input yang diterima. Dalam sistem dialog NPC, setiap state merepresentasikan titik tertentu dalam percakapan, sementara transisi terjadi berdasarkan pilihan atau input pemain. Struktur ini memungkinkan pengelolaan pohon dialog yang kompleks dengan cara yang jelas dan terkontrol, memastikan bahwa NPC dapat mempertahankan percakapan yang koheren dan relevan dengan konteks interaksi.[3]

Penerapan FSA dalam berbagai sistem telah dibahas dalam beberapa penelitian sebelumnya. Suharsih dan Atqiya [4] dalam penelitiannya mengimplementasikan konsep FSA pada aplikasi simulasi vending machine yoghurt Walagri. Penelitian tersebut menunjukkan bahwa FSA dapat digunakan sebagai logika dasar dalam pembuatan simulasi vending machine, yang memiliki kesamaan dengan sistem dialog NPC dalam hal

pengelolaan state dan transisi berdasarkan input pengguna.

Selain itu, Kaunang dan Waworundeng [5] mendemonstrasikan implementasi FSA dalam mesin penjualan tiket otomatis di taman hiburan, yang konsepnya dapat diadaptasi untuk sistem dialog NPC. Ernawati et al. [6] menerapkan konsep FSA dalam desain game edukasi, menyoroti fleksibilitas FSA dalam berbagai genre game. Ouedraogo et al. [7] memperkenalkan Extended Finite Automata (EFA), yang memungkinkan integrasi variabel dalam transisi state, sehingga respons NPC dapat lebih kontekstual dan bergantung pada interaksi sebelumnya.

Penggunaan FSA tidak hanya terbatas pada manajemen dialog satu arah. Barbanera et al. [8] memperkenalkan konsep Choreography Automata, yang memungkinkan pemodelan pola komunikasi asinkron antara beberapa agen atau NPC. Hal ini membuka peluang untuk menciptakan interaksi yang lebih kompleks dan dinamis, di mana NPC dapat berinteraksi tidak hanya dengan pemain tetapi juga dengan NPC lain, menciptakan ekosistem percakapan yang hidup dan realistis.[9]

Implementasi FSA dalam sistem dialog telah diaplikasikan dalam berbagai platform dan proyek. Alam [10] menunjukkan penggunaan FSA dalam chatbot Telegram bernama e-Knows, yang mampu mengelola dialog secara efisien dan responsif terhadap input pengguna. Pramadya [11] menerapkan Non-Deterministic Finite Automata (NFA) dan algoritma Monte Carlo Tree Search (MCTS) untuk menentukan perilaku NPC dalam game "The Last Hope", menekankan pentingnya automata dalam pengambilan keputusan NPC.

Penelitian ini berfokus pada implementasi FSA dalam desain sistem dialog interaktif NPC pada game "Legenda Guardian Kuno". Dengan menggunakan FSA, diharapkan dapat menciptakan sistem dialog yang lebih fleksibel dan adaptif, di mana NPC dapat memberikan respons yang sesuai berdasarkan pilihan pemain. Hal ini sejalan dengan penelitian yang menunjukkan bahwa penerapan model formal seperti FSA dapat meningkatkan interaksi dalam sistem dialog, terutama dalam konteks game yang kompleks [12]. Selain itu, penelitian ini juga bertujuan untuk menambah literatur mengenai penerapan FSA dalam pengembangan game dan sistem dialog interaktif. Melalui penelitian ini, diharapkan dapat memberikan contoh konkret dan terperinci tentang bagaimana FSA dapat digunakan untuk mengelola percakapan bercabang serta memengaruhi alur cerita dan ending game.[13] Penelitian ini juga berusaha mengidentifikasi manfaat dan keterbatasan penggunaan FSA dalam konteks pengembangan game modern.

Dengan mencapai tujuan-tujuan tersebut, diharapkan penelitian ini dapat memberikan kontribusi signifikan baik dalam praktik pengembangan game maupun dalam kajian akademik mengenai sistem automata dan interaksi manusia-komputer.

II. METODE PENELITIAN

Penelitian ini menggunakan pendekatan pengembangan sistem dengan menerapkan konsep Finite State Automata (FSA) untuk mendesain sistem dialog interaktif Non-Player Character (NPC) dalam game "Legenda Guardian Kuno". Metodologi ini dirancang secara sistematis untuk memastikan bahwa implementasi FSA dapat memenuhi tujuan penelitian.

2.1 Finite State Automata (FSA)

Finite State Automata (FSA) adalah model matematika yang digunakan untuk merepresentasikan sistem yang memiliki sejumlah state terbatas. Dalam konteks penelitian ini, FSA digunakan untuk memodelkan alur percakapan antara pemain dan NPC dalam game "Legenda Guardian Kuno". FSA terdiri dari beberapa komponen utama yang saling berinteraksi untuk mengelola transisi antar state berdasarkan input yang diterima.[14]

FSA dapat didefinisikan melalui lima komponen utama, yaitu State (Q), Input Symbol (Σ), Transition Function (δ), Initial State (S), dan Output (λ).

a. State merepresentasikan tahap dialog atau situasi tertentu dalam game. Setiap state memiliki nama unik yang menggambarkan kondisi atau titik percakapan tersebut. dan berfungsi sebagai titik referensi untuk mengelola percakapan yang sedang berlangsung.
b. Input Symbol merupakan pilihan dialog atau aksi yang dapat dipilih oleh pemain. Setiap pilihan ini akan mempengaruhi transisi state dalam FSA. Pemain memilih salah satu opsi ini, yang kemudian menentukan arah percakapan selanjutnya.
c. Transition Function adalah hubungan antar state berdasarkan input pemain. Fungsi ini mendefinisikan bagaimana sistem berpindah dari satu state ke state lain.
d. Initial State adalah state awal percakapan, yaitu state di mana percakapan dimulai ketika pemain pertama kali berinteraksi dengan NPC.
e. Output adalah respons NPC atau perubahan state yang terjadi setelah pemain memberikan input. Output ini dapat berupa dialog yang ditampilkan kepada pemain atau tindakan lain yang mempengaruhi perkembangan cerita.

2.2 Diagram State

Diagram state merupakan visualisasi percakapan yang menggambarkan alur dialog antara pemain dan NPC. Diagram ini membantu dalam perancangan dan pemahaman struktur dialog dengan cara yang lebih intuitif dan terstruktur.[11] Dengan menggunakan diagram state, pengembang dapat dengan mudah melihat bagaimana percakapan berkembang berdasarkan pilihan yang dibuat oleh pemain.[15]

Diagram state digunakan untuk memetakan setiap state dan transisi yang terjadi dalam percakapan. Diagram ini mencakup

- State Awal (Start State), Titik awal percakapan ketika pemain pertama kali berinteraksi dengan NPC.
- State Transisi, State perantara yang dicapai setelah pemain membuat pilihan tertentu.
- State Akhir (End State), State yang menandakan akhir percakapan atau menentukan arah cerita selanjutnya.
- Transisi, Garis penghubung antar state yang menunjukkan pilihan pemain dan perpindahan state.

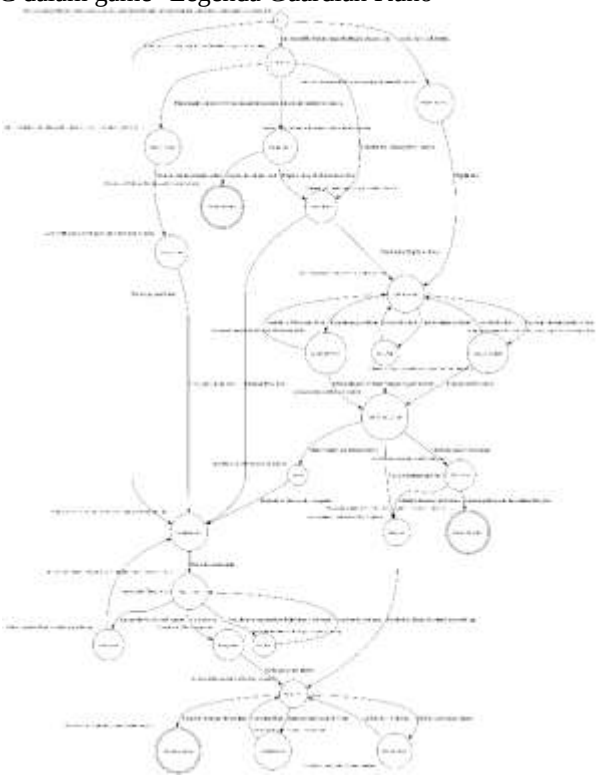
Sebagai contoh, berikut adalah gambaran diagram state untuk percakapan awal dalam game

- State "Start" NPC menyapa pemain dengan kalimat, "Selamat datang di desa kami, pengembara..."
  - Pilihan 1 "Saya akan membantu." → Transisi ke state "Accept_Mission".

- Pilihan 2 "Maaf, saya tidak tertarik." → Transisi ke state "Decline_Mission".
- Pilihan 3 "Saya memiliki urusan dengan kelompok peneliti lain." → Transisi ke state "Join_Rivals".

Diagram ini membantu pengembang untuk memastikan bahwa setiap pilihan pemain memiliki jalur percakapan yang jelas dan terdefinisi dengan baik.

Gambar 1 berikut menunjukkan diagram state FSA yang telah dirancang untuk desain sistem dialog interaktif NPC dalam game "Legenda Guardian Kuno"



Gambar 1: Diagram State FSA Desain Sistem Dialog

2.3 Tupel FSA

Finite State Automata (FSA) dapat secara formal didefinisikan sebagai sebuah tupel M = (Q, Σ, δ, S, λ), yang terdiri dari lima komponen utama

a. Q adalah himpunan semua state yang ada dalam FSA. Dalam game ini, Q dapat mencakup state seperti {Start, Accept_Mission, Decline_Mission, Join_Rivals, ...}. Setiap state dalam Q merepresentasikan titik tertentu dalam percakapan yang sedang berlangsung.
b. Σ adalah himpunan simbol input yang dapat diterima oleh FSA. Dalam konteks ini, Σ terdiri dari pilihan dialog yang diberikan kepada pemain, seperti {"Saya akan membantu.", "Maaf, saya tidak tertarik.", "Saya memiliki urusan dengan kelompok peneliti lain."}.
c. δ adalah fungsi yang mendefinisikan transisi antar state berdasarkan input yang diterima. Secara matematis, δ: Q × Σ → Q. Fungsi ini menentukan state mana yang akan dicapai setelah state saat ini menerima input tertentu.
d. S adalah state awal di mana FSA memulai operasinya. Dalam game ini, S ditetapkan sebagai state "Start", yaitu state di mana NPC menyapa pemain dan menawarkan misi.
e. λ adalah fungsi output yang menghasilkan respons NPC atau perubahan state setelah menerima input pemain. Output ini berupa dialog atau tindakan yang diberikan NPC sebagai respons terhadap pilihan pemain.

Berikut adalah definisi tupel FSA untuk percakapan awal dalam game "Legenda Guardian Kuno"

- Q = {Start, Accept_Mission, Decline_Mission, Ask_Benefit}
- Σ = {"Saya akan membantu.", "Maaf, saya tidak tertarik.", "Apa manfaat bagi saya?"}
- δ =
  - δ(Start, "Saya akan membantu.") = Accept_Mission
  - δ(Start, "Maaf, saya tidak tertarik.") = Decline_Mission
  - δ(Start, "Apa manfaat bagi saya?") = Ask_Benefit
  - δ(Ask_Benefit, "Baiklah, saya akan membantu.") = Accept_Mission
  - δ(Ask_Benefit, "Saya masih tidak tertarik.") = Decline_Mission
- S = Start
- λ =
  - λ(Start) = "Elder Lyra: Selamat datang di desa kami, petualang. Apakah Anda bersedia membantu kami membangkitkan Guardian Kuno?"
  - λ(Accept_Mission) = "Elder Lyra: Terima kasih! Kami sangat menghargai bantuan Anda. Temui Pustakawan Alaric untuk informasi lebih lanjut."
  - λ(Decline_Mission) = "Elder Lyra: Saya mengerti. Jika Anda berubah pikiran, kami akan menyambut bantuan Anda."
  - λ(Ask_Benefit) = "Elder Lyra: Anda akan mendapatkan penghargaan besar dan nama Anda akan tercatat dalam sejarah."

Dengan mendefinisikan tupel FSA secara formal, pengembang dapat memastikan bahwa setiap elemen dalam sistem dialog telah terstruktur dengan baik dan dapat dikelola secara efisien.

2.4 Implementasi Sistem

Pada tahap implementasi, model FSA yang telah dirancang diubah menjadi kode program yang dapat dijalankan. Implementasi dilakukan menggunakan bahasa pemrograman Python, dipilih karena kemampuannya dalam pemrosesan data dan kemudahan integrasi dengan format data seperti JSON.

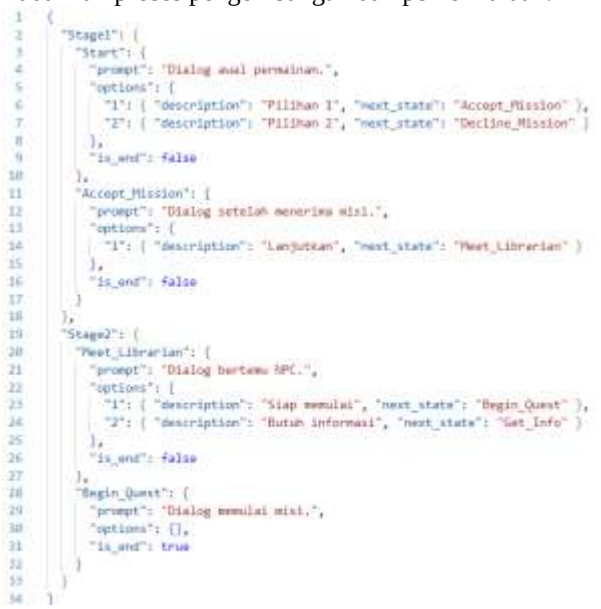
2.4.1 Pengembangan Kode Program

Pengembangan kode program dilakukan dengan membuat kelas-kelas yang merepresentasikan state dan automata. Kelas State digunakan untuk merepresentasikan setiap state dalam FSA, sementara kelas FiniteStateAutomaton mengelola transisi antar state berdasarkan input pemain.

Gambar 2: Code FSA

2.4.2 Penyimpanan Data Dialog dalam File JSON

Data dialog dan struktur FSA disimpan dalam file dialog.json dengan format JSON. Pendekatan ini memungkinkan pengelolaan dan pembaruan konten dialog tanpa perlu mengubah kode program, sehingga memudahkan proses pengembangan dan pemeliharaan.



Gambar3: Kerangka Dialog dalam JSON

2.4.3 Integrasi dengan Antarmuka Pengguna

Sistem dialog diintegrasikan dengan antarmuka pengguna berbasis teks, memungkinkan pemain berinteraksi dengan NPC melalui terminal atau command line interface (CLI). Antarmuka ini menampilkan prompt dari NPC, pilihan yang tersedia, dan menerima input dari pemain.

- Sistem menampilkan pesan dari NPC dan daftar pilihan yang dapat dipilih oleh pemain.
- Sistem menerima input dari pemain, memvalidasi input tersebut, dan melakukan transisi ke state berikutnya sesuai dengan pilihan yang dipilih.
- Jika pemain memasukkan input yang tidak valid, sistem memberikan pesan kesalahan dan meminta pemain untuk memasukkan pilihan yang benar.

Dengan integrasi ini, pemain dapat berinteraksi secara langsung dengan NPC.

2.5 Pengujian Sistem

Setelah implementasi selesai, dilakukan pengujian untuk memastikan bahwa sistem berfungsi sesuai dengan yang diharapkan dan memenuhi kebutuhan yang telah ditetapkan. Pengujian melibatkan beberapa jenis, yaitu pengujian fungsionalitas dan pengujian jalur percabangan.

2.5.1 Pengujian Fungsionalitas

Pengujian fungsionalitas bertujuan untuk memastikan bahwa setiap komponen sistem berfungsi dengan benar sesuai dengan spesifikasi. Dalam konteks ini, pengujian dilakukan pada setiap state dan transisi dalam FSA.

- Memastikan sistem dapat menerima input yang valid dan melakukan transisi ke state yang benar.
- Memastikan sistem dapat menangani input yang tidak valid dengan memberikan pesan kesalahan dan meminta input ulang dari pemain.

2.5.2 Pengujian Jalur Percabangan

Pengujian jalur percabangan dilakukan untuk memastikan bahwa semua kemungkinan alur percakapan telah terakomodasi dan berjalan sesuai dengan yang diharapkan.

- Mensimulasikan percakapan dengan berbagai pilihan untuk menjelajahi semua jalur yang mungkin.
- Dari state "Start" memilih opsi 1 → "Accept_Mission".
- Dari state "Start" memilih opsi 2 → "Decline_Mission".
- Dari "Start" pilih opsi 3, lalu di "Ask_Benefit" pilih opsi 1 → "Accept_Mission".
- Dari "Start" pilih opsi 3, lalu di "Ask_Benefit" pilih opsi 2 → "Decline_Mission".

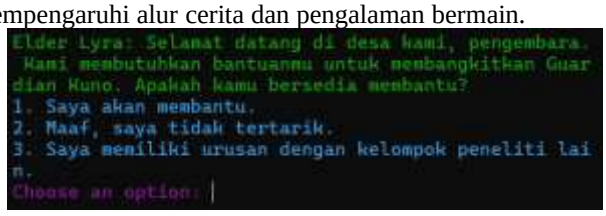
III. HASIL PENELITIAN

Penelitian ini menghasilkan desain sistem dialog interaktif Non-Player Character (NPC) dalam game "Legenda Guardian Kuno" dengan menerapkan Finite State Automata (FSA). Hasil penelitian mencakup perancangan model FSA, pengembangan kode program, pengujian sistem, dan evaluasi interaksi antara pemain dan NPC. Berikut adalah paparan hasil penelitian secara terperinci.

3.1. Hasil Implementasi Sistem Dialog NPC Berbasis FSA

3.1.1. Integrasi Sistem Dialog dengan Game "Legenda Guardian Kuno"

Implementasi sistem dialog telah berhasil diintegrasikan ke dalam game "Legenda Guardian Kuno". NPC utama, Elder Lyra, berfungsi sebagai titik awal interaksi pemain dengan dunia game. Sistem dialog memungkinkan pemain untuk berinteraksi dengan NPC melalui pilihan-pilihan yang disajikan, yang akan mempengaruhi alur cerita dan pengalaman bermain.



Gambar 4: Contoh Tampilan Dialog

Pemain dapat memilih salah satu opsi, dan sistem akan menampilkan respons sesuai dengan pilihan tersebut, sesuai dengan fungsi transisi (δ) yang telah didefinisikan dalam FSA

3.1.2. Penggunaan Struktur Data JSON dalam Implementasi

Sebagaimana dijelaskan pada Metode Penelitian, struktur dialog disimpan dalam format JSON, yang memudahkan pengelolaan dan pemeliharaan sistem dialog. Implementasi ini memungkinkan pengembang untuk menambahkan, mengubah, atau menghapus dialog tanpa perlu memodifikasi kode program secara langsung.

```
{
  "Stage1": {
    "Start": {
      "prompt": "Elder Lyra: Selamat datang di desa kami, pengembara...",
      "options": {
        "1": { "description": "Saya akan membantu.", "next_state": "Accept_Mission" },
        "2": { "description": "Maaf, saya tidak tertarik.", "next_state": "Decline_Mission" },
        "3": { "description": "Saya memiliki urusan dengan kelompok peneliti lain.", "next_state": "Join_Rivals" }
      },
      "is_end": false},
    // ... state lainnya ...
  },
  // ... stage lainnya ...
}
```

3.1.3. Implementasi Kode Program

Kode program yang dikembangkan menggunakan bahasa pemrograman Python dengan memanfaatkan kelas State dan FiniteStateAutomaton yang telah dijelaskan pada Bab 2. Library yang digunakan dalam implementasi program adalah json yang digunakan untuk bekerja dengan data dalam format JSON (JavaScript Object Notation)..

Berikut adalah cuplikan implementasi yang menunjukkan bagaimana sistem membaca struktur JSON dan mengelola alur dialog

Gambar 5: Code Load JSON dan Start State

Kode di atas menunjukkan bagaimana sistem membaca data dialog dari file JSON dan menginisialisasi automata dengan state awal yang telah ditetapkan.

3.2. Pengujian dan Analisis Sistem

Setelah implementasi selesai, dilakukan pengujian untuk memastikan bahwa sistem berfungsi sesuai dengan yang diharapkan. Pengujian ini meliputi pengujian fungsionalitas dan pengujian jalur percabangan.

3.2.1. Pengujian Fungsionalitas

Pengujian fungsionalitas bertujuan untuk memastikan bahwa setiap state dan transisi dalam FSA berfungsi dengan benar. Pengujian dilakukan dengan mensimulasikan berbagai skenario interaksi antara pemain dengan NPC.

Tabel 1: Hasil Pengujian Fungsionalitas

State Saat Ini	Input Pemain	State Berikutnya yang harus dicapai	Hasil Pengujian
Start	1 ("Saya akan membantu.")	Accept_Mission	Berhasil
Start	2 ("Maaf, saya tidak tertarik.")	Decline_Mission	Berhasil
Start	3 ("Saya memiliki urusan dengan kelompok peneliti lain.")	Join_Rivals	Berhasil
Start	4 (Tidak Valid)	Tetap di Start	Berhasil
Accept_Mission	1 ("Temui Librarian Alaric.")	Meet_Librarian	Berhasil
Accept_Mission	2 (Tidak Valid)	Tetap di Accept_Mission	Berhasil
Decline_Mission	1 ("Jelajahi desa.")	Explore_Village	Berhasil
Decline_Mission	2 (Tidak Valid)	Tetap di Decline_Mission	Berhasil
Join_Rivals	1 ("Setuju dan membantu mereka.")	Corrupt_Quest	Berhasil
Join_Rivals	2 ("Berpura-pura setuju sambil merencanakan sesuatu.")	Deceive_Rivals	Berhasil
Join_Rivals	3 ("Menolak dan meninggalkan mereka.")	Leave_Rivals	Berhasil
Join_Rivals	4 (Tidak Valid)	Tetap di Join_Rivals	Berhasil
Meet_Librarian	1 ("Ya, saya siap.")	Begin_Quest	Berhasil
Meet_Librarian	2 ("Bisakah saya mendapatkan lebih banyak informasi?")	Get_Info	Berhasil
Meet_Librarian	3 ("Saya perlu waktu untuk bersiap.")	Delay_Quest	Berhasil
Meet_Librarian	4 (Tidak Valid)	Tetap di Meet_Librarian	Berhasil
Begin_Quest	1 ("Mulai pencarian artefak.")	Ritual_Site	Berhasil
Begin_Quest	2 (Tidak Valid)	Tetap di Begin_Quest	Berhasil
Get_Info	1 ("Kembali ke Librarian.")	Meet_Librarian	Berhasil
Delay_Quest	1 ("Kembali ke Tahap Sebelumnya.")	Accept_Mission	Berhasil
Explore_Village	1 ("Kunjungi perpustakaan.")	Library_Discovery	Berhasil
Explore_Village	2 ("Pergi ke hutan sendirian.")	Find_Map	Berhasil
Explore_Village	3 ("Berbicara dengan penduduk desa.")	Talk_To_Villagers	Berhasil
Explore_Village	4 (Tidak Valid)	Tetap di Explore_Village	Berhasil
Library_Discovery	1 ("Kembali ke Eksplorasi Desa.")	Explore_Village	Berhasil
Find_Map	1 ("Mengikuti peta tersebut.")	Confrontation_Alone	Berhasil
Talk_To_Villagers	1 ("Mencari artefak sendiri.")	Confrontation_Alone	Berhasil
Corrupt_Quest	1 ("Kembali ke Tahap Sebelumnya.")	Join_Rivals	Berhasil
Deceive_Rivals	1 ("Jalankan rencana pengkhianatan.")	Betray_Rivals	Berhasil
Leave_Rivals	1 ("Kembali ke Eksplorasi Desa.")	Explore_Village	Berhasil
Ritual_Site	1 ("Lanjutkan ritual pembangkitan.")	Summon_Guardian	Berhasil
Ritual_Site	2 ("Serahkan tugas pada NPC lain.")	Delegate_Ritual	Berhasil
Ritual_Site	3 ("Periksa kembali persiapan.")	Double_Check	Berhasil

Ritual_Site	4 (Tidak Valid)	Tetap di Ritual_Site	Berhasil
Double_Check	1 ("Kembali ke Ritual Site.")	Ritual_Site	Berhasil
Confrontation_Alone	1 ("Melawan dengan keberanian.")	Fight_Alone	Berhasil
Confrontation_Alone	2 ("Mencari bantuan terakhir.")	Seek_Help	Berhasil
Confrontation_Alone	3 ("Mundur untuk menyusun strategi.")	Retreat	Berhasil
Confrontation_Alone	4 (Tidak Valid)	Tetap di Confrontation_Alone	Berhasil

Berdasarkan tabel tersebut, desain sistem telah diuji dengan berbagai input dan skenario, dan berhasil berfungsi sesuai dengan yang diharapkan dalam setiap pengujian. Semua transisi "State" sesuai dengan input yang diberikan, dan sistem dapat menangani input yang tidak valid dengan tetap berada di "State" sebelumnya.

3.2.2. Pengujian Jalur Percabangan

Pengujian jalur percabangan dilakukan untuk memastikan bahwa semua kemungkinan alur percakapan dapat diakses dan berjalan dengan benar. Beberapa jalur percakapan yang diuji antara lain

- Jalur Menerima Misi Langsung, Dari state "Start" memilih opsi 1, kemudian melanjutkan hingga state akhir.
- Jalur Menolak Misi, Dari state "Start" memilih opsi 2, dan mengeksplorasi konsekuensi dari penolakan misi.
- Jalur Bergabung dengan Rival, Dari state "Start" memilih opsi 3, dan melihat bagaimana alur cerita berkembang dalam jalur ini.

Analisis Hasil Pengujian

- Semua jalur percakapan berfungsi dengan baik dan sesuai dengan desain yang telah dibuat.
- Sistem dapat menangani input yang tidak valid dengan memberikan pesan kesalahan dan meminta pemain untuk memasukkan pilihan yang benar.

IV. PEMBAHASAN

Hasil penelitian menunjukkan bahwa penerapan Finite State Automata (FSA) dalam desain sistem dialog interaktif Non-Player Character (NPC) pada game "Legenda Guardian Kuno" mampu mengelola percakapan bercabang dengan struktur yang jelas dan memberikan respons adaptif berdasarkan pilihan pemain. Pembahasan ini akan menganalisis keberhasilan implementasi FSA, serta tantangan dan potensi pengembangan lebih lanjut yang diidentifikasi selama penelitian.

4.1 Keberhasilan Implementasi FSA

Implementasi FSA dalam sistem dialog NPC berhasil dilakukan sesuai dengan perancangan yang telah ditetapkan. Struktur FSA memungkinkan pengelolaan state dan transisi dengan efisien, memastikan bahwa setiap percakapan memiliki alur yang logis dan konsisten. Beberapa keberhasilan utama yang dicapai adalah sebagai berikut,

- Dengan menggunakan FSA, seluruh alur percakapan dapat divisualisasikan dalam bentuk diagram state, memudahkan pengembang dalam memahami dan mengelola setiap percabangan dialog. Hal ini juga mempermudah proses debugging dan pemeliharaan sistem dialog.
- NPC dapat memberikan respons yang relevan dan sesuai dengan pilihan pemain, menciptakan interaksi yang terasa lebih alami dan personal. Sistem ini mampu menangani berbagai pilihan pemain tanpa mengorbankan keutuhan alur cerita.
- Struktur FSA yang modular memungkinkan penambahan state dan transisi baru dengan mudah. Pengembang dapat memperluas dialog NPC tanpa harus merombak keseluruhan sistem, menjadikan proses pengembangan lebih efisien.

4.3 Tantangan dan Keterbatasan

Meskipun implementasi FSA menunjukkan hasil yang positif, terdapat beberapa tantangan dan keterbatasan yang dihadapi selama penelitian.

4.3.1 Kompleksitas Percakapan dan Pilihan Terbatas

Untuk menjaga sistem tetap sederhana dan mudah dikelola, jumlah pilihan dan percabangan dialog dibatasi. Hal ini dapat mengurangi kedalaman interaksi dan membuat percakapan terasa monoton.

- Menambah lebih banyak pilihan dan jalur percabangan dapat memperkaya interaksi dan memberikan lebih banyak variasi dalam alur cerita.
- Menggunakan EFA memungkinkan penambahan variabel dan kondisi yang lebih kompleks dalam transisi state, sehingga dialog NPC dapat lebih dinamis dan kontekstual.

4.3.2 Pengelolaan Data Dialog

Pengelolaan data dialog dalam format JSON memiliki keterbatasan dalam hal skalabilitas dan kompleksitas dialog yang dapat diakomodasi. Untuk percakapan yang lebih panjang dan bercabang, struktur data JSON dapat menjadi sulit dikelola.

- Menggunakan database terstruktur seperti SQLite atau MongoDB dapat mempermudah pengelolaan data dialog yang lebih kompleks dan besar.
- Mengembangkan tool atau script yang dapat membantu dalam pembuatan dan pengelolaan data dialog secara otomatis dapat meningkatkan efisiensi dan mengurangi potensi kesalahan.

4.4 Implikasi dan Manfaat Penelitian

Penelitian ini memberikan beberapa implikasi dan manfaat penting dalam pengembangan game dan sistem dialog interaktif sebagai berikut

- Memberikan contoh konkret bagaimana FSA dapat diimplementasikan dalam desain sistem dialog interaktif NPC, yang dapat dijadikan referensi bagi pengembang game lain yang ingin mengadopsi metode serupa.
- Menjadi dasar bagi penelitian lebih lanjut dalam penerapan automata dan teknologi interaktif lainnya dalam pengembangan game dan aplikasi berbasis interaksi manusia-komputer.

V. KESIMPULAN

Hasil penelitian menunjukkan bahwa penggunaan FSA efektif dalam mengelola percakapan bercabang, memberikan respons adaptif, dan meningkatkan kualitas interaksi antara pemain dan NPC. Sistem yang

dikembangkan mampu memenuhi tujuan utama penelitian, yaitu menciptakan sistem dialog yang fleksibel dan responsif terhadap pilihan pemain.

Beberapa poin utama yang dapat disimpulkan dari penelitian ini adalah

a. FSA terbukti mampu mengelola dialog NPC dengan struktur yang terorganisir dan mudah dipahami. Dengan model FSA, alur percakapan dapat divisualisasikan secara jelas, memudahkan proses perancangan dan implementasi sistem dialog yang kompleks.
b. Implementasi FSA dalam sistem dialog NPC memungkinkan NPC memberikan respons yang adaptif sesuai dengan pilihan pemain. Hal ini meningkatkan imersi dan keterlibatan pemain dalam alur cerita, serta membuat pengalaman bermain menjadi lebih dinamis dan personal.
c. Struktur FSA yang modular memungkinkan penambahan state dan transisi baru dengan mudah, menjadikan sistem dialog yang dikembangkan fleksibel dan mudah diadaptasi untuk pengembangan game lebih lanjut. Sistem ini dapat dikembangkan lebih lanjut untuk mengakomodasi alur cerita yang lebih kompleks dan beragam.
d. Meskipun implementasi berhasil, terdapat beberapa tantangan yang diidentifikasi, seperti kompleksitas pengelolaan data dialog. Solusi yang diusulkan, seperti optimisasi pengelolaan data, membuka peluang untuk pengembangan lebih lanjut.

Untuk penelitian selanjutnya, disarankan untuk melibatkan pengguna dalam pengujian sistem secara langsung untuk mendapatkan umpan balik yang lebih komprehensif mengenai pengalaman pengguna dan efektivitas sistem dalam konteks yang lebih luas. Hal ini dapat membantu dalam mengidentifikasi area yang perlu ditingkatkan dan memastikan bahwa sistem dialog yang dikembangkan benar-benar memenuhi kebutuhan dan ekspektasi pemain.

VI. REFERENSI

[1] D. Abdurrohman, "Perancangan Game Lost Forest 3d Menggunakan Metode Finite State Machine Berbasis Desktop," *Jati (Jurnal Mahasiswa Teknik Informatika)*, vol. 7, no. 4, pp. 2280–2287, 2023, doi: 10.36040/jati.v7i4.7483.

[2] M. A. Darmawan, H. Haryanto, and Y. Rahayu, "Perilaku Penyerangan NPC Berbasis Fuzzy Sugeno Pada Game Action-RPG Bertema Sejarah Geger Pacinan," *Creative Information Technology Journal*, vol. 4, no. 3, p. 195, 2018, doi: 10.24076/citec.2017v4i3.110.

[3] F. Said, D. Andriyanto, R. Sari, and W. Gata, "Perancangan Validasi Permohonan Narasumber Pada Sistem Informasi Permohonan Narasumber Menggunakan Finite State Automata," *Paradigma - Jurnal Komputer Dan Informatika*, vol. 22, no. 2, pp. 189–196, 2020, doi: 10.31294/p.v22i2.8157.

[4] R. Suharsih and F. Atqiya, "Penerapan Konsep Finite State Automata (FSA) pada Aplikasi Simulasi Vending Machine Yoghurt Walagri," *Edsence*, vol. 1, no. 2, pp. 71–78, Dec. 2019, doi: 10.17509/edsence.v1i2.21778.

[5] F. J. Kaunang and J. Waworundeng, "Implementation of Finite State Automata in an Amusement Park Automatic Ticket Selling Machine," *Abstract Proceedings International Scholars Conference*, vol. 7, no. 1, pp. 1801–1810, 2019, doi: 10.35974/isc.v7i1.1979.

[6] M. Ernawati, W. Gata, E. H. Hermaliani, L. Kurniawati, and S. Rahayu, "Implementasi Konsep Finite State Automata Pada Desain Game Edukasi Jenis Hewan," *Technologia Jurnal Ilmiah*, vol. 13, no. 1, p. 65, 2022, doi: 10.31602/tji.v13i1.6268.

[7] L. Ouedraogo, R. Kumar, R. Malik, and K. Åkesson, "Nonblocking and Safe Control of Discrete-Event Systems Modeled as Extended Finite Automata," *Ieee Transactions on Automation Science and Engineering*, vol. 8, no. 3, pp. 560–569, 2011, doi: 10.1109/tase.2011.2124457.

[8] F. Barbanera, I. Lanese, and E. Tuosto, "Choreography Automata," pp. 86–106, 2020, doi: 10.1007/978-3-030-50029-0_6.

[9] D. S. H. Tobing, "Transformasi Desain Video Game Dari Media Bermain Menjadi Media Ekspresif," *Jommit Jurnal Multi Media Dan It*, vol. 6, no. 2, pp. 62–69, 2023, doi: 10.46961/jommit.v6i2.629.

[10] C. N. Alam, "Implementation of Finite State Automata on E-Knows Telegram Chatbot," *Coreid*, vol. 1, no. 1, pp. 33–41, 2023, doi: 10.60005/coreid.v1i1.3.

[11] R. D. Pramadya, "Penerapan Non-Deterministic Finite Automata (NFA) Dan Decision Making Menggunakan Algoritma Monte Carlo Tree Search (MCTS) Menentukan Perilaku Non-Player Character (NPC) Pada Game the Last Hope," *Jurnal Coscitech (Computer Science and Information Technology)*, vol. 4, no. 2, pp. 500–509, 2023, doi: 10.37859/coscitech.v4i2.5419.

[12] H. Brabra, M. Báez, B. Benatallah, W. Gaaloul, S. Bouguelia, and S. Zamanirad, "Dialogue Management in Conversational Systems: A Review of Approaches, Challenges, and Opportunities," *Ieee Transactions on Cognitive and Developmental Systems*, vol. 14, no. 3, pp. 783–798, 2022, doi: 10.1109/tcds.2021.3086565.

[13] U. Nurhasan, "Analisis Perilaku Non Playable Character (Npc) Pada Game Menggunakan Fuzzy Sugeno," *Techno Com*, vol. 19, no. 3, pp. 308–320, 2020, doi: 10.33633/tc.v19i3.3477.

[14] C. Slaviero and E. H. Haeusler, "A-Games: Using Game-Like Representation for Representing Finite Automata," 2021, doi: 10.5753/weit.2021.18918.

[15] Y. Yanto, D. Ismunandar, E. Erni, S. Setiawan, and M. I. R. Ihsan, "Desain Game Edukasi Ilmu Tajwid Bagi Anak Usia Dini Menggunakan Pemodelan Finite State Automata," *Edumatic Jurnal Pendidikan Informatika*, vol. 5, no. 1, pp. 80–88, 2021, doi: 10.29408/edumatic.v5i1.3317.