Adaptive Scenario Selection in Serious Games Using Finite State Machines and Agent-Based Models

Melissa Rêgo Rodrigues
Ivaldir Honório de Farias Júnior
Universidade de Pernambuco
R. Cap. Pedro Rodrigues
São José, Garanhuns - PE, 55294-902
melissa.rego@upe.br

ivaldir.farias@upe.br

ABSTRACT

This study presents a hybrid approach to scenario adaptation in serious games, using Finite State Machines (FSMs) and Agent-Based Models (ABMs). Focusing on the educational RPG genre of software development, the proposed model aims to automatically adjust the behavior of non-playable characters (NPCs) and the game's progression based on the player's actions and preferences. The methodology included a literature review, followed by the development of a simulation in the JFLAP software, integrating FSMs to manage states and ABMs to promote dynamic and realistic reactions. The results highlight the viability of this integration, which offers an immersive experience by combining predictability and flexibility in the behavior of NPCs. As a limitation, we identified the need for more advanced tools to integrate the technologies. For future work, it is proposed that frameworks be developed to facilitate this approach's implementation.

Keywords

Serious games, Finite State Machines (FSM), Agent-Based Models (ABM), Scenario adaptation, Gamification

INTRODUCTION

There has long been a quest to facilitate the learning process, and to this end, various pedagogical tools and technologies have been adopted to create a bridge between the individual and knowledge. In this context, digital games are increasingly gaining ground as a facilitating technology for teaching concepts and ideas, as discussed in the theory of gamified learning. This theory proposes that gamification positively affects learning outcomes by influencing behaviors and attitudes relevant to learning and, over time, its applications have been shown to have small but significant positive effects on cognitive, motivational, and behavioral learning outcomes [12].

Proceedings of DiGRA 2025

It is within this context that research into the use of gamification in professional training—through simulations of real scenarios for teaching in the workplace—has been growing steadily. However, the idea of games with purposes beyond entertainment is not new. Serious games, a term first introduced by Clark C. Abt in 1970 in his book *Serious Games*, are games that aim to instruct and inform as well as entertain [11]. Serious games combine elements of entertainment with educational purposes and are useful tools to aid learning in complex areas such as computer theory and software development. These games offer an interactive environment where players can apply theoretical concepts in practical and simulated situations, facilitating the understanding and retention of technical content while offering an error-tolerant and experimental environment.

In this context, a wide variety of techniques, methodologies, and technologies are employed to enhance the learning experience, from the application of basic operational algorithms, such as truth tables, to deep learning approaches. Despite the numerous applications, a common challenge is to ensure that the non-playable characters (NPCs) and scenarios of serious games offer a balance between predictability and realism, which is essential to engage the player without compromising the educational objective.

Among all these technologies, this article aims to explore how finite-state automata, a basic concept in the theory of computation, remains relevant for use in the field of game computing, especially in serious games [16]. To this end, we analyzed the possibility of integrating Finite State Machines (FSM) and Agent-Based Models (ABM), computational concepts that, although different in complexity and operation, can together form a well-balanced decision-making basis for the learning environment proposed by the serious game.

Agent-Based Models (ABMs) are a fundamental computational technique for modeling complex dynamic systems composed of autonomous agents that represent individuals, groups, or entities. These agents make independent decisions based on their internal rules and interactions with other agents and the environment. Such models are particularly valuable for simulating phenomena in which macro-level patterns emerge from the aggregation of individual agent behaviors, offering insights into system dynamics that are not easily captured by traditional top-down models [17]. In the context of serious games, ABMs enable the simulation of realistic human behaviors, such as the decisions of a fictitious client in a software development project, allowing the game to adapt in real-time to the player's actions in a more organic and immersive way.

To structure the behavior of agents, Finite State Machines (FSMs) are often used. In computation theory, a finite state machine is a mathematical model consisting of a finite set of states, transitions between these states, and associated actions [18]. Each agent can be represented by an FSM, where its behavior is defined by a current state and the transitions that occur based on specific conditions or inputs. For example, an agent can switch between states such as "waiting", "deciding", and "executing" according to defined rules, allowing dynamic and predictable responses to the environment.

This study integrates the capabilities of ABM with FSMs to create an adaptive system that automatically adjusts the behavior of characters who are not under the player's

control—the NPCs—and the progression of the scenario as the player interacts with the game. This results in a more immersive and realistic environment, where the NPCs, who in the game scenario envisioned in this article represent the clients of a software development company, exhibit dynamic and reactive behaviors modeled by an FSM that controls state transitions based on the player's preferences and actions.

THEORETICAL REFERENCE

This section focuses on the theoretical concepts behind the experiment, including FSMs in serious and educational games, highlighting their ability to model NPC behavior and adjust game difficulty based on player progress and actions [3]. Literature showcases various FSM applications, such as creating intelligent agents that respond dynamically to player choices, adapting game content to user profiles, and optimizing educational game performance. Games like "Flora the Explorer" [2] and BARA [5] illustrate how FSMs simulate complex interactions and offer immersive learning experiences. The section emphasizes FSMs' importance in creating adaptive and realistic games for effective and engaging learning, especially in educational and professional settings.

Finite State Machines (FSM)

Finite State Machines (FSMs) are computational models used to describe systems that can exist in various states and transition between them based on inputs or events [18]. These models are widely used across domains such as control systems, formal language theory, and digital games. An FSM consists of three main components: states, representing the system’s conditions; events, which trigger transitions; and actions, which may occur during or after transitions between states [19].

Within this framework, FSMs can be categorized as deterministic (DFA) or non-deterministic (NFA) [20]. A Deterministic Finite Automaton (DFA) has exactly one transition for each state and input pair, resulting in predictable behavior. In contrast, a Non-deterministic Finite Automaton (NFA) allows multiple possible transitions for the same input, including spontaneous ones. While NFAs offer greater theoretical flexibility, both models are equally expressive in recognizing regular languages.

These foundational concepts are explored throughout this article to demonstrate how FSMs can support the teaching of software development and computational theory. In particular, we examine how FSMs enable the gamification of learning by making complex technical content more engaging and interactive [12].

FSMs are widely applied in serious and educational games to model non-player character (NPC) behavior and dynamically adjust game difficulty. For example, Arif et al. [1] proposed a neural network-enhanced FSM to adapt game scenarios based on user profiles. In this approach, the FSM defines game states, while transitions are informed by a multilayer perceptron that predicts user preferences based on five factors: work, hobbies/interests, origin, group composition, and repetition. The system achieved a scenario-selection accuracy of 67.5%, highlighting the effectiveness of combining FSMs with artificial neural networks.

Another common strategy involves Dynamic Difficulty Adjustment (DDA), which tailors the challenge level to a player's experience. Suaza et al. [3] implemented FSMs to control enemy behavior in combat, with states such as long-range attacks, melee attacks, movement, and retreat. Heuristics, such as the ratio of player-to-enemy health points (HP), guide transitions and maintain balanced encounters. For instance, battles are adjusted so both characters reach the end with comparable HP, indicating a well-calibrated challenge.

Adeniyi et al. [4] highlight another advantage of FSMs: their ability to optimize performance by reducing the computational load of real-time AI decisions. This is especially valuable in educational games, where high frame rates (FPS) are essential for a responsive and fluid user experience.

Liu et al. [5] introduced the serious game BARA, which simulates realistic software requirement elicitation using FSMs. In BARA, stakeholders are represented by NPCs whose responses change based on player actions and timing, mimicking real-world professional scenarios. Its accessible editing interface allows educators to customize scenarios without programming, making it adaptable to diverse learning contexts. The FSM used in BARA integrates both emotional and technical feedback, creating a more immersive and practical learning experience.

In the educational game *Ulun Smart-Kid* [21], developed by Andrea and Nurhuda (2020), the FSM model plays a central role in defining the behavior of a game agent that acts as a virtual learning companion. This intelligent agent reacts to player input with emotional expressions—such as happiness, disappointment, or urgency—based on the correctness of word-assembly tasks in the Banjar language. Responses are triggered by specific in-game events (e.g., success or failure in assembling letters, or time running out), creating a pedagogically meaningful and interactive experience. The FSM enables the agent to function as a virtual tutor, offering feedback in context, thereby enhancing engagement and promoting personalized learning.

The authors also emphasize that FSMs support the scalability of *Ulun Smart-Kid*, allowing for the easy addition of new game scenarios and agent behaviors. This design aligns with the Open/Closed Principle from object-oriented programming, promoting extensibility without modifying existing code. Furthermore, FSMs contribute to efficient game performance, enabling consistent animations and responsive interactions. Andrea's and Nurhuda's work [21] illustrates how FSMs can significantly enhance educational game design by ensuring interactivity, adaptability, and maintainability.

While previous studies use different mechanisms—such as heuristics or structured protocols—to manage state transitions, the model proposed in this article adopts a probabilistic approach inspired by Thibaud L'Yvonnet's (2022) work [7]: Discrete-Time Markov Chains (DTMCs). A DTMC augments state-transition systems with probabilities, where the next state depends solely on the current one (i.e., the system is memoryless).

Definition 1: *A discrete-time Markov chain over a set of atomic propositions AP is a tuple (S, Sinit, P, L), where S is a set of states (state space), Sinit ⊆ S is the set of initial states, P: S × S → [0, 1] is the transition probability function (where* ∑□′∈□

P(s, s') = 1 for all s ∈ S), and L: S → 2^AP is a function that labels the states with atomic propositions about AP.

Markov chains provide a discrete notion of time, considering only significant transitions rather than the continuous passage of time. This simplifies analysis and reduces computational overhead—an important consideration in modern games, where high-fidelity graphics and AI can impose demanding hardware requirements. As noted by Adeniyi et al. [4], FSMs already contribute to smoother computational performance by maintaining finite, manageable state spaces with predictable transitions.

In summary, FSMs are part of a class of models with clearly defined rules and predictable behavior. Their popularity in digital games—from early classics like *Pac-Man* (1980), where they managed ghost behaviors, to modern titles like *The Legend of Zelda: Breath of the Wild* (2017)—is largely due to their ease of implementation and debugging. However, as expectations for realism and interactivity increase, FSMs reveal limitations in representing emergent behaviors and dynamic environments. They struggle to handle unforeseen variables or multiple simultaneous interactions, which can limit immersion and adaptability.

To address these challenges, we propose integrating FSMs with Agent-Based Models (ABMs). While FSMs provide strict control over local NPC behaviors, ABMs enable more complex interactions by modeling autonomous agents that respond to external variables and to each other. This hybrid approach is especially useful in serious games, where educational goals require predictability but real-world complexity must also be simulated.

In a training game for software development, for example, FSMs can define predictable NPC states such as "satisfied and calm", "satisfied and impatient", "dissatisfied and calm", and "dissatisfied and impatient", with transitions based on player response time and solution quality. Integrating an ABM layer would allow the NPCs to evolve dynamically—such as clients becoming more demanding as deadlines approach or reprioritizing features—thus creating a more realistic and immersive experience that better reflects actual project dynamics.

Recent advances in computing and modern game engines like Unity and Unreal Engine make such hybrid implementations increasingly feasible. These tools support the integration of deterministic FSM control and emergent ABM behaviors in a unified framework. The result is a powerful and flexible approach that combines the strengths of both models—simplicity and structure from FSMs, adaptability and realism from ABMs—pushing the boundaries of what serious games can achieve in terms of educational impact and user engagement.

Agent-Based Models

Intelligent agents are autonomous entities capable of independently perceiving and acting upon their environment to achieve specific objectives [2]. Agent-Based Models (ABMs) enable the simulation of complex systems through three core components: agents, the environment, and interaction rules. Each agent follows a defined set of rules to assess its situation and decide on future actions, determining how it interacts both with other agents and the environment [6].

In serious games—particularly those that simulate real-world processes such as software development—ABMs can model dynamic and responsive behaviors. For instance, fictitious clients or regional scenarios can influence evolving software requirements, reflecting the ever-changing nature of real-world projects. This modeling approach aligns the training environment of serious games with one of the few constants in software development: change.

Unlike deterministic models such as FSMs, ABMs are well-suited to simulating more complex and decentralized behaviors. Each agent in an ABM can make decisions independently, based on its own perceptions and logic, resulting in flexible and often unpredictable system dynamics. For example, in a simulation by Hassanpour et al. (2022), agents modeled human behavior during an earthquake evacuation [13]. Their autonomy allowed them to respond with a high degree of realism, closely mimicking human reactions under stress. In such contexts—where unpredictability is desirable—ABMs provide immense value.

However, in educational contexts, particularly serious games with learning objectives, unpredictability must be carefully controlled. A serious game must remain coherent and structured to guide the player effectively through a learning process. While unpredictability enhances realism, it can hinder comprehension when it interferes with instructional consistency. For instance, in a simulation-based game about software development, NPCs representing clients may need to react dynamically, but not so unpredictably that they disrupt the game's pedagogical structure. The repetition of core procedures—such as requirement gathering, implementation, and testing—is critical for reinforcing learning. As such, agents representing clients should exhibit controlled variability, enabling different scenarios without undermining educational consistency.

To reconcile these needs, this article proposes an integrated model that combines FSMs and ABMs. In this approach, FSMs manage predefined behavioral states and transitions, ensuring that NPCs remain understandable and structured in their reactions to player actions. ABMs, meanwhile, govern broader agent behaviors and interactions, enabling adaptive and dynamic reactions to changes in the environment or among other agents. This layered model achieves a balance between predictability and realism, enhancing immersion without compromising instructional clarity.

A concrete example of FSM usage in educational games is Flora the Explorer [2]. In this game, FSMs manage the different conditions of the game, such as "waiting," "correct answer," "wrong answer," "game won," and "game lost." Transitions between these states are determined by the player's actions, and an intelligent agent provides real-time feedback—such as cheerful animations for correct answers or explanations for incorrect ones. This combination of FSM and intelligent agents creates responsive and personalized learning experiences that are both engaging and educational.

Beyond this, ABMs are widely employed in other game types. In RPGs, they are used to model opposing characters; in simulations, they can represent entire populations—such as during the COVID-19 pandemic [14]. Because agents in ABMs operate independently and interact stochastically, these models are particularly

effective in replicating social phenomena and collective behavior. This makes them well-suited to modeling realistic NPC populations and dynamic environments.

In the case of the serious game proposed in this article—a simulation of a software development company—every client, developer, and even environmental factors can be modeled as intelligent agents with their own variables and objectives. Specifically for clients, an FSM would define the set of behavioral states and transitions—such as shifting demands, approval cycles, or feedback reactions—allowing the game to simulate the uncertainty of client interactions while preserving the instructional structure necessary for effective learning.

RELATED WORK

Several researchers have explored the use of Finite State Machines (FSMs) and Agent-Based Models (ABMs) in games to enhance scenario adaptation and non-playable character (NPC) behavior. These studies provide valuable insights into the application of computational models in game design, although they differ in scope, implementation, and pedagogical objectives. Most research tends to implement either FSMs or ABMs in isolation, with few integrating both within the same adaptive system—particularly in the domain of software development education.

One particularly relevant study that integrates both models is *"An Intelligent Agent of Finite State Machine in Educational Game 'Flora the Explorer'"* by Pukeng et al. (2019) [2]. In this work, FSMs are used to manage game states and transitions based on player input, while an intelligent agent observes and reacts to the FSM's current state. Although conceptually simple, this model provides effective real-time feedback, which is critical in serious educational games.

Similarly, Adegun et al. (2020) [15], in *"Design and Implementation of an Intelligent Gaming Agent Using A* Algorithm and Finite State Machines,"* propose a hybrid system where agents make decisions through FSMs. Their model defines NPC states such as idle, attack, or search, with transitions based on environmental observations. This approach allows agents to perceive their surroundings and respond through FSM-driven behaviors, demonstrating the synergy between agent-based modeling and state machines in complex environments.

From a pedagogical standpoint, the study *"BARA: A Dynamic State-Based Serious Game for Teaching Requirements Elicitation"* by Liu et al. (2023) presents a concept closely aligned with the game proposed in this research. While BARA focuses on teaching requirements elicitation, our model addresses broader aspects of software engineering processes, including product development and continuous validation. Both games aim to replicate real-life scenarios to reinforce procedural understanding through guided interactivity.

Based on this brief analysis, we conclude that although prior research has highlighted the individual strengths of FSMs and ABMs in serious games, few studies have proposed a truly hybrid system—one that balances structured, rule-based transitions with emergent, adaptive agent behavior. Our proposed integration model aims to fill this gap by creating a serious educational game where NPCs respond

dynamically to player decisions, enhancing realism while preserving the structure needed for effective skill development.

METHODOLOGY

The methodology employed in this research began with an exploratory literature review conducted on the Google Scholar platform, covering the period from 2019 to 2024. The search string used was:("serious game" OR "serious games" OR "serious gaming" OR "educational games") AND ("finite state machine"). This search yielded 665 scientific papers, from which 15 articles were selected using the following inclusion criteria: Peer-reviewed article format, focus on the use or analysis of FSMs in serious games and Emphasis on NPC behavior modeling.

Subsequently, a secondary search was conducted by adding the term "agent-based model" to the query. This yielded 12 additional articles, which were filtered using the same selection criteria, with a preference for more recent publications to better reflect the current state of research in FSM and ABM integration. Below is an image displaying the methodology followed.

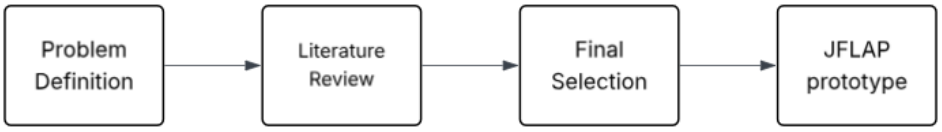

Figure 1: Methodology

Following the literature review, a preliminary FSM model was developed using the JFLAP tool to simulate NPC behaviors. This model included states such as “satisfied and calm” and “dissatisfied and impatient”, with transitions based on gameplay variables like player response time and solution adequacy. The JFLAP simulation enabled a visual and practical understanding of how FSMs can drive NPC behavior in serious games, based on player interaction and game events.

However, due to JFLAP’s limitations, it was not possible to directly implement ABM features or simulate autonomous agent interactions. As such, while the FSM portion of the system was successfully modeled, full integration with ABM remains a subject for future work. Qualitative analysis suggests that combining FSMs and ABMs would substantially enhance immersion and complexity, without compromising the structured nature needed for educational experiences.

The next stage of this research involves implementing a custom software prototype that combines FSM and ABM frameworks to create a fully dynamic and adaptive NPC system. This will allow us to validate, in practice, the proposed model’s effectiveness in delivering realistic, engaging, and pedagogically sound game experiences.

RESULTS AND DISCUSSIONS

This section will detail the development procedure for the JFLAP project, covering the stages involved in building the model and the results obtained from its

implementation. The focus will be on explaining the technical processes, decisions made during development, and the methodologies applied to create the game and design the intelligent agents. After this, the intelligent agent model adopted for the project will be discussed, with an emphasis on defining the rules and behaviors that guide the interactions of the NPCs within the game. It will be presented how the agents should be configured to react adaptively to the player's actions, based on changes in the environment and decisions made throughout the simulation.

FSM implementation

This section details the development process of the Finite State Machine (FSM) model created using JFLAP, focusing on the design decisions, technical steps, and methodologies employed to construct the simulation. The objective was to create a structured FSM capable of managing the behavior of non-playable characters (NPCs)—specifically, simulated clients—in a serious educational game designed to teach computer theory and software development.

The FSM was implemented to simulate two key behavioral dimensions: client satisfaction and mood, which reflect how NPCs (clients) respond to the player's decisions and performance throughout a development cycle. This design emulates real-world client-developer interactions, where technical success must often be balanced with interpersonal and time-management factors.

Customer Sets And Preference Criteria

Each client may occupy one of four combined states, representing varying levels of satisfaction and urgency. These states are defined as follows:

- Satisfied and calm: The customer is happy with the progress of the software and maintains a positive attitude.
- Satisfied and Impatient: The customer likes what has been delivered, but expects quick responses from the player to finalize the order.
- Dissatisfied and Calm: The customer is dissatisfied with the proposal, but still receptive to changes.
- Dissatisfied and Impatient: The customer is dissatisfied and their patience wanes with each interaction, requiring the player to adapt quickly and precisely.

These states are determined by two key gameplay variables:

- Response Time (TR): The time taken by the player to fulfill a client’s request.

- Quality of Solution (QS): A score that reflects how well the delivered solution meets client expectations, ranging from 0% to 100%.

The FSM governs transitions between these states based on changes in TR and QS. The preference criteria include variable states that influence the customer's mood and satisfaction. The table below shows each of them in better detail.

	States	What they symbolize
Response time	Impatient (TA - High response time)	Delayed customer service and late delivery of demands
Response time	Calm (TB - Low Response Time)	Punctual customer service and on-time delivery of demands
Quality of the solution	Dissatisfied (SI - Inadequate Solution)	The solution was not delivered according to the defined requirements
Quality of the solution	Satisfied (SA - Adequate Solution)	Solution delivered according to defined requirements

Table 1: Finite state machine states

For example, a customer who was initially satisfied and calm may become impatient if the player is slow to respond or makes an inadequate implementation. Similarly, if the customer is dissatisfied and impatient, but the player quickly makes adjustments, they can move into a calm state again. Transitions are limited to changes in one attribute at a time, ensuring realistic emotional dynamics (e.g., a client cannot instantly jump from “Dissatisfied and Impatient” to “Satisfied and Calm” without an intermediate step).

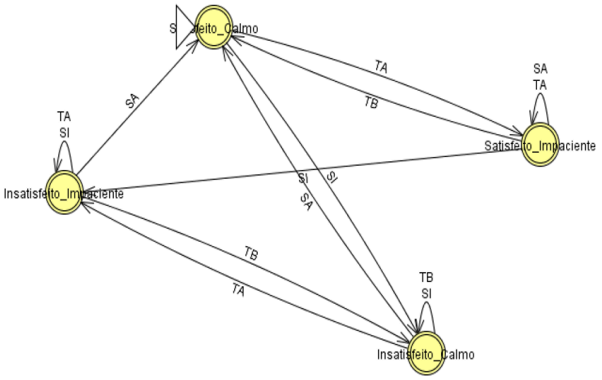

Figure 2: Customer state machine, representing the transition between states

In the JFLAP software, the states were represented as follows: TA (High response time), TB (Low response time), SA (Adequate solution, i.e. in line with customer requirements), and SI (Inadequate solution, i.e. not in line with customer requirements). In this state machine, the possible transitions are from satisfied_calm to satisfied_impatient, dissatisfied_cal,m or remaining in the state, maintaining a pattern in which one of the attributes must always remain the same in the transition, and it is not possible to change state instantaneously to its complete opposite.

Implementation of the Intelligent Agent Model in the FSM

The integration of an Intelligent Agent Model with the Finite State Machine (FSM) aims to enrich the simulation by replicating, in a more realistic way, the dynamic and multifaceted behavior of clients in software development. This model is designed to capture the nuances of a programmer's day-to-day life, such as the need to balance deadlines, technical requirements, and relationships with stakeholders. The intelligent agent, in this context, is represented by each client in the educational game, and its main function is to react adaptively to the player's actions, transitioning between states of mood and satisfaction previously defined in the state machine.

In the game, the client would act as an intelligent agent that constantly evaluates the player's performance and adjusts its behavior, simulating common situations faced by software developers. This behavior is managed by the state machine, which organizes the client's possible states and defines transitions based on environmental variables. The aim is to create a system that, like that of A. F. Pukeng et al [2], in their work on the same type of application, an intelligent finite state machine agent in educational games, integrates the two technologies in favor of the greatest possible similarity to the situation that the game seeks to prepare the player to face.

The intelligent agent model has three main functions in the context of FSM: dynamic state management, controlling the transition between customer mood and satisfaction states based on variables such as response time and solution quality; realistic behavior simulation, creating an immersive experience by

replicating human behaviors, such as the gradual loss of patience with delays or the increase in satisfaction with fast and accurate solutions; influence on the flow of the game, modulating the difficulty and feedback provided to the player, forcing them to deal with challenges such as tight deadlines and demanding customers. For example, if the player takes too long to fulfill an order (high TR) or delivers an inadequate solution (low QS), the customer can move from "Satisfied and Calm" to "Satisfied and Impatient" and eventually to "Dissatisfied and Impatient", demanding quick and effective action to reverse the situation.

In summary, the FSM functions as the core of the client's behavior, while the intelligent agent model acts as the mechanism that feeds this FSM with contextual data and processes the transitions. Integration takes place in such a way that the intelligent agent receives data from variables in the game environment:

- Response Time (TR): Measured in seconds. High values (>5s) increase impatience.

- Quality of Solution (QS): Percentage scale. Values <50% reduce satisfaction.

The agent uses these variables to determine the state transition of the same name (Table 1). If the TR exceeds a threshold (e.g. 5 seconds), the customer becomes more impatient. Similarly, if the QS is low (e.g. below 50%), customer satisfaction decreases. Based on the analysis, the agent adjusts the customer's current state, moving through the states of the finite state machine. For example, a customer who was initially "Satisfied and Calm" (SC) can change to "Satisfied and Impatient" (SI) if the TR is high. If the player doesn't correct the situation quickly, the customer can become "Dissatisfied and Impatient" (II). In addition to the state transitions, the agent adjusts internal variables, two of which are:

- Patience: Progressively decreases with increasing TR.
- Satisfaction: Increases or decreases based on QS.

The updated state of the client is reflected in the game (OUTPUT), providing visual and textual feedback to the player, such as NPC facial expressions or comments on performance. This feedback is of paramount importance, as it has been one of the main factors in the positive impact of gamification, being a central learning mechanism triggered by game design elements [11].

CONCLUSION

This work presented the development of a project using the integration of Finite State Machines (FSM) with Agent-Based Models (ABM), intending to improve the simulation and behavior of NPCs (non-playable characters) in serious games. By combining these two approaches, we explored the possibility of creating a system that offers predictability in the behavior of NPCs without losing flexibility in their reactions to player actions. The use of FSMs helps to maintain control over game states and ensure a predictable experience, while ABMs allow NPCs to adapt more autonomously and realistically, simulating complex behaviors. Predictability helps ensure that the intended learning or skills are effectively achieved, as well as keeping the player focused on the game's objectives. On the other hand, flexibility in NPC

reactions adds to the player's immersion in the proposed universe, promising a more satisfying experience with the system.

Despite the contributions made, some limitations were noted, such as the complexity of integrating the two technologies and the need for more advanced tools to fully implement intelligent agents. The lack of a specific environment that only combines FSMs and ABMs was a challenge, but it also highlighted the importance of future innovations in this area, since JFLAP itself is not enough to truly simulate this integration. To do this, it would be necessary to program both the state machine and the agent-based models, possibly developing a framework to make it easier to implement the integration of these technologies in other projects.

As a suggestion for future work, we recommend creating software or frameworks that integrate these two technologies more efficiently, as well as expanding the model to include other types of interaction and more complex scenarios. The use of real data and the implementation of machine learning techniques could also be explored to further increase the adaptability and accuracy of agents in serious games. To sum up, this study has sought to contribute to instigating development interest in the subject and to propose the advancement of the application of intelligent agent models in serious games, offering a basis for the development of more sophisticated and effective games for teaching and training in various areas of knowledge.

REFERENCES

[1] Y. M. Arif et al, 2023. "An artificial neural network-based finite state machine for adaptive scenario selection in serious game." Int. J. Intell. Eng. Syst., vol. 16, no. 5, pp. 488-500.

[2] A. F. Pukeng et al, 2019. "An intelligent agent of finite state machine in educational game 'Flora the Explorer'." J. Phys. Conf. Ser. pp. 042006.

[3] J. Suaza, E. Gamboa and M. Trujillo, 2019. "A health point-based dynamic difficulty adjustment strategy for video games." In Joint Int. Conf. Ent. Comp. Ser. Games. pp. 436-440.

[4] A. E. Adeniyi et al, 2024. "Development of Two Dimension (2D) Game Engine with FSM-based AI." Procedia Comp. Sci., vol. 235, pp. 2996-3006.

[5] Y. Liu, T. Li, Z. Huang and Z. Yang, "BARA: A Dynamic State-based Serious Game for Teaching Requirements Elicitation", 2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET), Melbourne, Australia, 2023, pp. 141-152, doi: 10.1109/ICSE-SEET58685.2023.00020.

[6] Babichenko, Dmitriy, et al. "The Use of Agent-Based Models As Non-Player Characters in Serious Games." 2020 IEEE 8th International Conference on Serious Games and Applications for Health (SeGAH). IEEE, 2020.

[7] Thibaud L.. Relationships between human activity models and brain models : application to clinical serious games. Modeling and Simulation. Université Côte d'Azur, 2022. English.

[8] GRIGOLETTI, Pablo Souza. Markov chains. Retrieved from, v. 19, n. 10, p. 2014, 2011.

[9] Dengel, A. (2019). Seeking the treasures of theoretical computer science education: towards educational virtual reality for the visualization of finite state machines. In: Proceedings of 2018 IEEE international conference on teaching, assessment, and learning for engineering, TALE 2018. https://doi.org/10.1109/TALE.2018.8615288

[10] Silverman, B. G., Bharathy, G., & Weyer, N. (2019). What is a good pattern of life model? Guidance for simulations. SIMULATION, 003754971879504. doi:10.1177/0037549718795040

[11] Abt, C. (1970). Serious Games. New York: The Viking Press.

[12] Hamari, J., Xi, N., Legaki, Z., & Morschheuser, B. (2023). Gamification. In Hawaii International Conference on System Sciences (p. 1105).

[13] Hassanpour, S., Gonzalez, V., Liu, J., Zou, Y., & Cabrera-Guerrero, G. (2022). A hybrid hierarchical agent-based simulation approach for buildings indoor layout evaluation based on the post-earthquake evacuation. Advanced Engineering Informatics, 51, 101531.

[14] M. Mohmmadnejad, M. Dorrigiv and F. Yaghmaee, "A Serious Game Designed to Simulate Coronavirus Transmission," 2020 International Serious Games Symposium (ISGS), Tehran, Iran, 2020, pp. 88-93, doi: 10.1109/ISGS51981.2020.9375463.

[15] Adegun, A., Ogundokun, R. O., Ogbonyomi, S., & Sadiku, P. O. (2020). Design and implementation of an intelligent gaming agent using A* algorithm and finite state machines. Int J Eng Res Technol. ISSN, 0974-3154.

[16] Jagdale, D. (2021). "Finite state machine in game development." International Journal of Advanced Research in Science, Communication and Technology 10.1.

[17] Gilbert, N. (2019). Agent-based models. Sage Publications.

[18] Wang, J. (2019). Formal Methods in Computer Science. CRC Press. p. 34. ISBN 978-1-4987-7532-8.

[19] Vieira, P., & Corchado, J. (2015). A formal machines as a player of a game. In Distributed Computing and Artificial Intelligence, 12th International Conference (pp. 137-147). Springer International Publishing.

[20] Belzer, J.; Holzman, Albert G.; Kent, A. (1975). Encyclopedia of Computer Science and Technology. Vol. 25. USA: CRC Press. p. 73. ISBN 978-0-8247-2275-3.

[21] Andrea, R., & Nurhuda, A. (2020). Developing Edu-Game “Ulun Smart-Kid” Learning Media of Banjar Language and Game Agent with Finite State Machine Model. International Journal of Education and Management Engineering, 10(6), 10-16.