CHAPTER 2

LITERATURE REVIEW

2.1 Related Research

Table 2. 1 Literature Studies

No.	Heading	Author/Year	Research Results
1	DESKTOP-BASED RPG GAME "THE ROYAL SWORD" USING THE FINITE STATE MACHINE METHOD (FSM)	Fadel Marzian, Mukti Qamal, 2017	The Finite State Machine (FSM) method in the RPG game The Royal Sword serves to limit the transfer of missions or levels so that the game becomes more organized and organized with gradual and sequential completion
2	Implementation of Finite State Automata in Process Study Plan Card Filling Student	Ridwan Ahmad Ma'arif, Fauziah, 2018	The placement of courses is carried out using NFA which has been sorted based on the provisions of each course.
3	GAME APPLICATION DEVELOPMENT "SORT OUT GARBAGE" USING FINITE STATE MACHINE MODELING	Sahrul, Fitri Karimah, Alzahid Muhazabah, Aries Dwi, Prasetyo, Ariana Yunita, Nurulbaiti Listyendah Zahra, 2018	After conducting experiments for the game "Sort the Garbage", the study resulted that the application of the game can be built from modeling ε-NFA
4	Fruit Rojak Vending Machine Design with Finite State Automata	Ranu Agastya Nugraha, Yanto, Astriana Mulyani, Windu Gata, 2020	This research has resulted in the design of a fruit rojak automatic peddling machine that can accept input and selection of combinations of several types of fruit as fruit rojak ingredients as well as the selection of seasonings chili sauce according to the degree of spiciness that produces expected output

2.2 Game

In the big Indonesian dictionary, "*Game*" means game. *Games* are interactive games that require a person or computer to play them. With various categories of *games*, action (*shooting*), *fighting, adventure, simulation, role playing, strategy, puzzle, sport, education* (Ardi & Sutabri, 2014).

As a result of the visual representation of the game, it can be divided into two categories, 2D and 3D. (3D). A 2D game is a game that, mathematically speaking, uses only two basic coordinate axes, x and y, so the camera in this kind of game can only give the player an "image" about what happens in the game. As a result, the concept of the camera in 3D games is indistinguishable from the concept of the camera in 2D games, namely that the camera can be used other than shifted (as in 2D games) but can also be used with the axis in question. (Marzian & Qamal, 2017)

2.3 PES (Pro Evolution Soccer)

PES (Pro Evolution Soccer) has been developed by Konami in 2001, the pes (Pro Evolution Soccer) game has continued to grow every year since it followed the development of the “Playstation" game console. This PES (Pro Evolution Soccer) game sold more than 106.8 million copies and was released on various platforms including playstation, playstation 2, playstation 3, playstation 4, playstation 5, Xbox, Android, iOS, Windows and playstation portable (PSP). Pes (Pro Evolution Soccer) is a game that imitates real football games. (Aliyu, 2020).

Pes (Pro Evolution Soccer) has NPCs (Non-Player Characters) who have AI artificial intelligence (Artificial Intelligence) to make the game more interesting and do not make the browsers who play feeling bored playing the PES (Pro Evolution Soccer) game. NPC (Non-Player Character) is a supporting character in a game in the form of human characters, robot animals and others, NPCs (Non-Player Character) in pes (Pro Evolution Soccer) games cannot be controlled by players, but NPC (Non-Player Character) can act and carry out activities that seem to be controlled by the player and move based on computational results or AI provided by the game developer so that it can affect the gameplay.

2.4 FSM (Finite state machine)

Finite State Machines (FSMs) are methodological approaches to system control that use the concepts of *state*, *event*, and action to describe different levels of activity or principles operation system action (action).

In *the state machine the* system recognizes only one condition (state). The system will be stopped or switched to another state if the specified event is received. The system will continue to perform the same actions under the same circumstances until the system receives the desired event, either coming from outside the system or is a component from the system itself. The system will then stop. The transition connects all the states, and each of them leads to the next one. In addition, system actions during the input-input transition are usually considered the source of this change. Such actions can be categorized as long-term or short-term actions, depending on the context in which they are performed. (Yulsilviana & Ekawati, 2019).

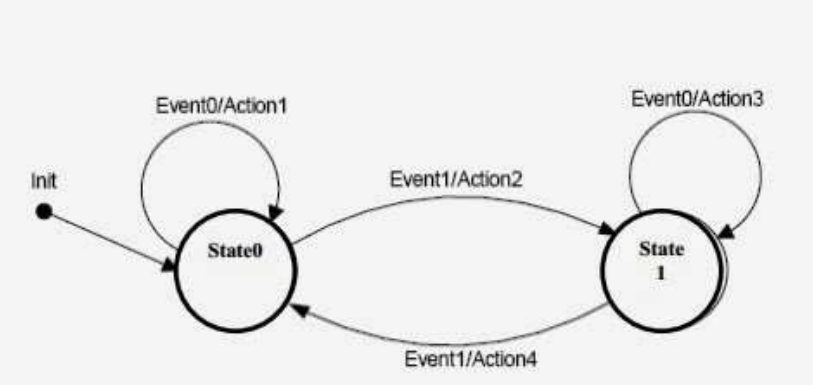

Figure 2.1 Simple state diagram

Source: (Yulsilviana & Ekawati, 2019)

The following diagram illustrates the logic with two different states, two different inputs, and eight different output actions that are all the same: When the system starts, the system will transition to state 0, where it will generate Action1 if Event0 is detected, whereas Action2 will be executed if Event1 is detected. Then, the system will switch to state 1 and continue. (Pamilih, 2016)

OOP, or Object-Oriented Programming, is one option for implementing FSM rather than a more traditional procedural approach. On the other hand, the

advantages of using OOP in FSM include high flexibility and ease of use, whether for simple, medium, or more complex systems. However, as a side maNDFAat, OOP also provides maNDFAat such as the ability to reuse code that has been written (code reusability), thus reducing the amount of code that must be written. (Pamilih, 2016)

Formally FSM is expressed by 5 tuples or M= (Q, Σ, δ, S, F):

Q = state set /position

Σ = set of input/input/alphabetic symbols

δ = transition function

S = initial state

F = set of final states

State-to-state communication is powered by finite state machines, which are used to provide interactivity between states determined by the decisions of each state in question. As a result, NPCs (Non-Player Characters) will be able to provide actions to players based on the player's assessment of them.

2.5 NDFA (Non-Deterministic Finite Automata)

Deterministic Finite Automata (DFA) and Non-deterministic Finite Automata (NDFA) are two types of FSA. The main difference between the two is that DFA has only one transition state direction, while NDFA can have more than one direction. Non-deterministic Finite Automaton (NFA) is a type of finite state machine (FSM) where one of the next states is not fully determined by the current state or input. The set of possible next states that automata can move from a certain state (qa) to another state (qb) in response to an input (α). (Sahrul et al., 2018)

To describe the logic model flow used Non-deterministic Finite Automata (NDFA). Non-deterministic Finite Automata (NDFA) is one of the automata machine languages, this machine language was chosen because it is easy to apply and in accordance with human logic. (Wirasbawa et al., 2019).

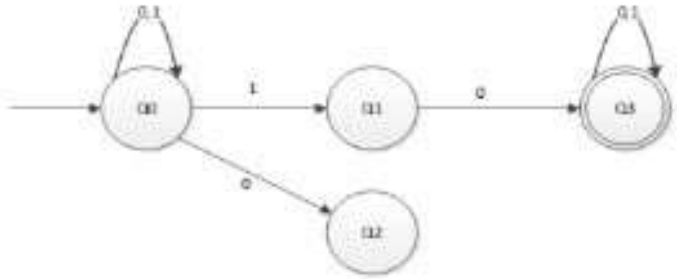

Figure 2.2 Simple NDFA state diagram
Source: (Nugraha et al., 2020)

Figure 2.2 shows a simple NDFA state diagram with 4 states and 2 different inputs and outputs. If it is entered in Qo, then the state moves to Q1 or Q2 with the output being a different action. (Nugraha et al., 2020).

Non-deterministic Finite Automata (NDFA) allows for a single input to give rise to a transition to more than one condition and allows it to provide multiple movements so that the output cannot be ascertained. (Yanto et al., 2021).