International Journal of Computer Applications Technology and Research
Volume 4– Issue 12, 916 - 922, 2015, ISSN: 2319–8656

A Proposed Simulation Model of Automatic Machine For House Paint Selection Using Finite State Automata

Emigawaty
Departement of Informatics
STMIK AMIKOM
Yogyakarta, Indonesia

Abstract: One important criterion considered by the public at the time of house selection or property is the aspect of house paint colors. Principally, the selection of house paint is not exessively complicated, but if the homeowner does not have enough knowledge about how to make a good combination with the paint color, the consequence is the color of choice is not in accordance with their expected. This study aims to provide a simulation of how to do a combination of paint color selection using automated machines. The method used in the design of this model is the Prototype method and blackbox testing to ensure the system is made valid and reliable. In the mean while, the approach automata models used are Finite State Automata (FSA) has the characteristics due to work by identifying and capturing the pattern in the compilation process of determining the color of wall paint the house. This paper also resulted in the identification of input symbols, FSA-NFA diagrams, rules production, NFA, equivalence transition table of FSA-DFA, and FSA-DFA transition diagram. At the same time, in the design of a prototype, also produced mockup simulation applications in order to generate automatic machine house paint selection. Thus, problems such as confusion and lack of knowledge of homeowners against a combination of paint colors can be overcome properly.

Keywords: automatic machine, prototype, finite state automata

1. INTRODUCTION

Along with the high expectations of today's human needs would be a challenge for developers of information technology to create models or equipment that can support daily activities. This is evidenced by the increasing number of variants of information technology innovation both based hardware and software which proved quite effective in helping humans for teaching and learning activities, work, sports, and other business aspects. One of the areas that affect the development of the information technology tools is the property sector. The rise of property business in almost all regions of the country is certainly a phenomenon as demand for property is also one of the primary needs of human. The developers of housing today competing to offer properties ranging from design models, thematic; to the color selection is also an important consideration to attract public attention.

The selection of paint color for some people may not be a priority compared to some other criteria. But not a few people have a high level of sensitivity to choose the appropriate color combinations or desired according to the taste or thematic expected by the owner of the house later. Determine the combination of colors of paint it takes precision and a good knowledge before deciding the color of choice. Because if we are incorrect in choosing and combining these colors is not likely to result from the combination does not conform to what is expected.

Automatic machine is one variant of the technology innovations created to change an activity or process that is the business of a conventional or to automatic with the goal of accelerating time activity process itself [1]. One of the automated machines that began to develop at this time is a compilation engine automatically selecting paint the walls of the house. The presence of the house paint compilation engine is expected to be one solution to simplify the electoral process and the determination of the color of paint with different variations. Finite State Automata (FSA) is an abstract machine in the form of mathematical models of systems with discrete inputs and outputs that can recognize the simplest language also known as regular language and can be implemented in practice. FSA also serves as a model that is very useful to help recognizing and capturing a pattern in the data. This model can also receive inputs and produce outputs that have a finite number of states and can transform from one state to another based on input and transition functions in it [2]. FSA on a compilation machine house paint color selection is the right choice to model the process in this case. After simulating paint color selection is expected to house the homeowner will get a selection of color illustrations and recommendations in accordance with the needs and desires.

In this study, using a model of the application of the concept of FSA on the implementation of the automatic machine compilation paint of the house where the automatic machine has the characteristic assignment by identifying and capturing the pattern in the compilation process of determining the color of house paint. Therefore, problems such as confusion and lack of knowledge of homeowners against a combination of paint colors can be overcome with good. Moreover, this study also used some of the conditions that limit to do simulation and compiling election house paint colors using 3 size capacities gallon of paint (5kg, 10kg, and 15kg) with color inputs or raw are: yellow, blue, and red as primary colors, black and white as a neutral color, green (mixture of blue and yellow) as the secondary color, purple (mix of blue and red) as the secondary colors, orange (mix of red and yellow as secondary colors, while the composition used is 1 to 1.

2. LITERATURE REVIEW

2.1 Language Theory

In the process, a clump of computer science has two elements, namely: First, the model and the fundamental idea of the theory of computation. Secondly, the so-called engineering techniques for the system design of the computational model, both in terms of software and devices severity. Meanwhile, language and automata theory included in the first element

that makes this concept as a model and the fundamental idea of the theory of computation [3].

One theory being unique or singular when it may be good alternative solution of a phenomenon that exists in a science discipline. Computer science itself has a very broad scope ranging from concept, design theory, computer networks, programming, until the business processes within an information system. For that was created or constructed abstract models of the computer system or computing. Model theory of language has an important function, and generally how the computing system may be invoked on a software development and hardware [4].

In studying the theory of language, is identical with an understanding of formal language, primarily for the benefit of the design compiler and processing a script or text processor. By definition, a formal language is a set of sentences in a language that is raised by a grammar the same [3]. In its characteristics, that a formal language can be generated by two or more different grammar. Background that the theory of language can be called as well as the formal language for grammar created precedes the generation of each sentence. In contrast to human language, which is the reverse, while, grammar was created to formalize the words that live in the community.

Utdirartatmo [3] wrote that the automata is a system that consists of several finite state, where the state represents information regarding the input, and can also be translated as a reminder mesing. In this case, the input is meant automata machine is considered as the language that must be recognized by a machine. Then, the automata machines can make decisions that recognize or analyze whether the input is acceptable or not. In line with that rose by Linz [2], that for modeling hardware is a needed models automaton, which can also serve as a temporary storage medium and has the capability to transform an input to an output.

Furthermore, Juarna [5] provide some basic understanding of the language and automata theory as the basis of a common understanding of the definition and the main terminology as follows:

a) Symbol is an abstract entity (such as the definition of a point in geometry). A letter or a figure is an example of the symbol.
b) The string is a row of a limited (finite) symbols. For example, if a, b, and c are three symbols then abcb is a string that is constructed from the three symbols.
c) If w is a string, the string length is expressed as $|w|$ and defined for the count (number) symbols that make up the string. For example, if w = abcb then $|w| = 4$.
d) The empty string is a string with zero fruit symbols. Empty string ε is expressed by a symbol (or ^) that $|\varepsilon| = 0$. The empty string can be viewed as a symbol of a vacuum because both are composed of zero fruit symbols.
e) Alphabet is hinpunan up (finite set) symbols.

2.2 Grammar and Language

Grammar by Ullman [6] is a collection of the sets of variables, symbols terminal, the initial symbol, which is governed formally by the rules of production. Historically, teradapat a mathematician named Noam Chomsky in 1959 performed clustering or classification level language into four categories, which became known as Chomsky Hierarcy. The clustering can be seen in Table 1:

Table 1. Classification Level of Language

Language	Automatic Machine	Production Rules Limitation
Regular / Type 3	Finite State Automata (FSA) covers Deterministic Finite Automata (DFA) & Non Deterministic Finite Automata (NFA)	α is a variabel symbol. Maximum β has a variable symbol that when there is in the rightmost position
Context Free / Type 2	Push Down Automata (PDA)	α is a variable symbol
Context Sensitive / Type 1	Linier Bounded Automata	$\|\alpha\| \leq \|\beta\|$
Unrestricted / Phase Structure / Natural Language / Type 0	Turing Machine	No Limitation

2.3 Finite State Automata (FSA)

Finite State Automata (FSA) or often also referred to as finite automata, in principle, not a physical machine but a mathematical model of a notation or phrase that can receive input and output is discrete. FSA is part of the machine automata regular languages, which have a finite number of states, and can move from one state to another. One shortcoming of these automata models is that the FSA does not have the storage allocation so that the ability to remember is very limited [3].

Furthermore, as proposed by Ulman [7] that the FSA is a tool that can be used to design a system, whereby the mechanism can be applied or implemented on lexical analysis, text-editor, the network communication protocol, and checks parity, elevator system, traffic light system, or also the automatic machine. FSA formally expressed by 5 tuple or M = (Q, Σ, δ, S, F), where:

Q = set of state / status
Σ = set of symbols input / input / alphabet
δ = transition function
S = the initial state / initial position (initial state)
F = set the final state

Finite State Automata is a machine that can recognize regular language classes and has the following characteristics:

a) Ribbons inputs contains a series of symbols (string) from the set of symbols / alphabet,
b) Every time after reading one character, posisir read head will be in the next symbol,
c) At any time, the FSA is at a certain status,
d) The number of valid status for the FSA is finite.

Based on the function and definition of Finite State Automata (FSA) and also in terms of its ability to change its state then this model can dikaslifikasikan into 2 groups: Deterministic Finite Automata and Non-Deterministic Finite Automata.

2.4 Deterministic Finite Automata (DFA)

At Deterministic Finite Automata (DFA), from a state there is exactly one next state for each symbol of the input received, cannot be empty and more than one state. DFA formally expressed by 5 tuple or M = (Q, Σ, δ, S, F), where Ulman [7]:

Q = set of state / status
Σ = set of symbols
δ = transition function
S = the initial state (S ε Q)
F = set the final state (can be more than 1)

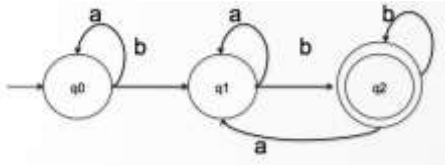


Figure 1. Deterministics Finite Automata (DFA) Machine

DFA configuration in Figure 1 is expressed formally as follows:

Q = {q0, q1, q2}
Σ = {a, b}
S = q0
F = q2

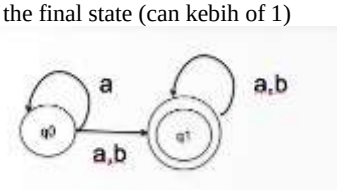
While the transition of existing functions in Figure 1 are as follows:

d (q0,a) = q0
d (q0,b) = q1
d (q1,a) = q1
d (q1,b) = q2
d (q2,a) = q1
d (q2,b) = q2

2.5 Non-Deterministic Finite Automata (NDFA)

In the non-deterministic finite automata of a state there can be 0,1, or more bows out (transition) labeled with the same input symbol. FSA formally expressed by 5 tuple or M = (Q, Σ, δ, S, F), where:

Q = set of state / status
Σ = set of symbols
δ = transition function
S = the initial state (S ε Q)
F = set the final state (can kebih of 1)



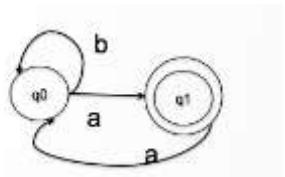


Figure 2. Non-Deterministics Finite Automata (DFA) Machine

2.6 Related Works

1. Melly [8] with the title of paper "Application of Concept Finite State Automata at Coffee Beverage Maker Machine Auto". In this study focused on an application of the concept of FSA in an automatic coffee machine beverage maker. The method used in this study is the implementation of formal methods that can be used to bridge (the formal specification) the manufacture, development and verification of hardware and software, which can be used from the initial design to the test results. The study produced a simulation applications beverage automatic coffee maker is an application simulating a machine that can make the process of making coffee drinks and variations automatically where the completion of the process used concept of FSA. Disadvantages of this study are not yet modeled Equivalence of NFA to DFA thus cannot be recommended to be developed or translated into the programming language.
2. Suprihatin [9], titled paper "Finite State Automata for Parsing Beheading Syllabary In Indonesian". In this study how decapitated syllables in Indonesian is good and right, while the FSA itself functions as an abstract machine, which is used to help design, the parsing (decapitation) syllable. FSA in its implementation is used to describe the structure of vowels, consonants, or the structure of words in Indonesian accordance with Spelling Enhanced (EYD). The downside of this research has not been able to show in detail the transition diagram for each of the components or variables measured.
3. Dihya [10], with paper title "Implementation of the Transitional Graf Defining Formal Languages". This study on how to model a formal language with transition graph and proof when modeled with other modeling. This is done to determine how the machine answers the problem decisions based on the input string to provide the output of 'yes' or 'no'. So that it can be seen how the issues will be seen run private computing simple graphical representation of the transition. The downside of this research is only at the stage of analysis and produces a transition table, not to the implementation of an object or case studies, so the validity of this study cannot be measured.

3. METHODOLOGY

3.1 Prototype Model

Methodology is the unity of methods or rules used by the work of a science. While the method is a systematic method or technique to grind. The general objective of this development is to provide ease in conveying information, reduce costs and save time, improve control, drive growth, improve the productivity and profitability of the organization.

System development can mean the preparation of a new system to replace the old system as a whole or improve existing systems. System development method used is the method Prototype. Prototype is a method used in a systems approach to make something quick and gradual program that can immediately be evaluated by the user. The stages contained in this prototype method is as follows [11]:

1. Identification of User Needs, at this stage developers and users meet. User describes the system requirements.
2. Create Prototype; developers began to create a prototype of the system.
3. Test Prototype, after prototype form testing prototype user and give criticism or suggestions.
4. Improving Prototype, In This stage developers make modifications in accordance with the input from the user.
5. Develop Prototype, after evaluation and perfect the system in accordance with the wishes of the user. Then the developer completed the final system in accordance with the insert of the wearer.

3.2 FSA Formal Specification

Formal specification by definition is a specification or statement, which is represented in the form of a formal language with the intention to describe what, should be done or decided by the software [12]. In the implementation, this study will use a formal specification in order to make a model of an object or system behavior using mathematical approaches such as sets and sequences.

After having a specification, either in the form of formal or informal, the implementation can be made. Implementation of the model can be made of conventional or automatic. Making the manual implementation is only suitable for use in the design of the level of complexity is not too high. For complex systems, easy error occurred. The process of making the implementation is automatically called automatic synthesis. To guarantee the truth of the synthesis process, the synthesis process must be substantiated. If a tool for automatic synthesis and synthesis processes that are used can be proved right then the results can be said to be "correct by construction".

Usually this is done to a system that departs from a system specified formally, then performed "refining" in order to obtain results in more detail, and done repeatedly until it becomes implementations in the level of abstraction desired (eg up to level layout transistor). Each step in the refining process is guarded by a mechanism (for example, by a theorem prover) so it cannot perform the process of refining the wrong [12].

4. RESULTS AND DISCUSSION

4.1 Input Symbol Identification

In the design stage and construction of the simulation modeling of automated machinery house paint color selection using the FSA, the necessary step towards the identification of input symbols in advance. This is necessary in order to provide the initial terms of the initiation of the formal specification by way of input variables, process variables, and variable selection. In practice, this study will use a formal specification that is used to create a model of an object or system behavior using arithmetic approach such as sets and sequences.

The following is a formal specification through initiation illustration of input and output variables:

Tabel 2. Formal Specification of Input Identification

Process Initiate		Capacity Selection		Compile Process	
0	Back to start state	j	Gallon 5 kg	m	Red
a	Choose green	k	Gallon 10 kg	n	Add Red
b	Choose light green	l	Gallon 15 kg	o	Yellow
c	Choose dark green			p	Add Yellow
d	Choose purple			q	Blue
e	Choose light purple			r	Add Blue
f	Choose dark purple			s	Black
g	Choose orange			t	Add Black
h	Choose light orange			u	White
i	Choose orange tua			v	Add White
				w	Compile

In Table 2 describes the formal specification in the initiation input variables in the form of a variable to i, and the process variables, namely m up to w. As for the election process capacity is represented by the variable j, k, and l. In the implementation, application built will have an interface to allow the user or the user selects multiple colors and desired gallon capacity according to their needs.

4.2 FSA – Non DFA

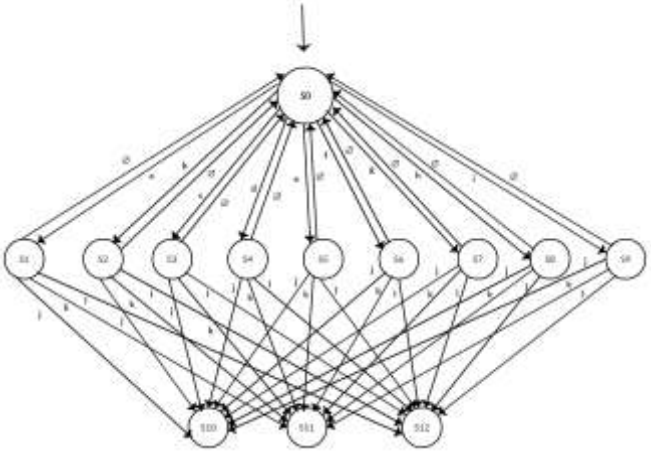


Figure 3. Diagram of Finite State Automata - NFA

In Figure 3 it can be seen that the Finite State Automata start work on State S0, in this diagram, there are 9 colors that can be, namely: green [a] to State S1, light green color [b] to State S2, dark green color [c] to State S3, purple [d] to the State S4, light purple [e] to the State S5, and petunia [f] to State S6. Then, from State to State S1 to S9 choice for gallon capacity, ie paint 5kg [j] to State S10, paint 10kg [k] to State S11, and paint 15kg [l] to State S12. Mathematical model on FSA-NFA diagram is then translated into the design of the interface by providing drop down option or combo box for each option.

Transition specification diagram illustrates the process that is automated election house paint using the concept or theory of FSA. In principle, the machine is designed to mengikuki pattern of the flow of paint color selection process is based on several primary colors or base is then processed according to color choices finally as an output. Is because each state has had towards both the source state and state clear objectives, then it is highly unlikely to happen mistakes or errors in processing or even have an end result that is not in accordance with a predetermined scheme. In the mean time, in order to be able to read and interpret symbols input the formal specification designed, the FSA granted terminology in the form of a dictionary of symbols is fixed and redundancy with other processes.

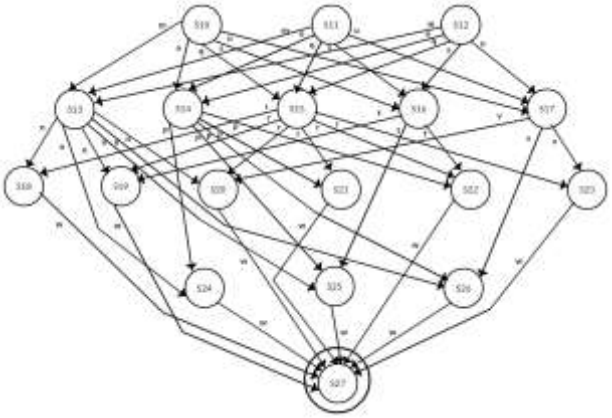


Figure 4. Diagram of Finite State Automata - NFA (continued)

Figure 4 is an advanced process flow previously, where the State S1 to S9 State 5kg choice (j) to State S10, 10kg (k) to State S11 and 15kg (l) to S12. Furthermore, from State S10, S11, and S12, if the choice of red (m) to State S13, yellow (o) to State S14, blue (q) to State S15, black (s) to State S16, and

white (u) to State S17. After that, from State S13 add red (n) to State S18, S19, S20, S24, S25, and S26. Of State S14 add yellow (p) to State S21, S22, S23, S24, S25, and S26. Of State S15 add blue (r) to S18, S19, S20, S21, S22, and S23. Of State S16 add black color (s) to State S19, S22, and S25. Of State S17 add white (v) to State S20, S23, and S26. Furthermore, from State S18 to State S26 stirring (w) to State end S27. From State to State S1 to S27 option ∅ (cancellation) back to S0.

4.3 Rule Production of NDFA

In the domain of language and automata theory, grammar can also be referred to as Grammar, which is a collection of the set of variables, symbols terminal (start and an end state), the initial symbols are constrained by the rules of production. Furthermore, the production rules in a State Non-Deterministic Finite Automata (NFA) have produced some string variables.

NFA Production Rules:

S0 = S
S1 = A S2 = B S3 = C S4 = D
S5 = E S6 = F S7 = G S8 = H
S9 = I S10 = J S11 = K S12 = L
S13 = M S14 = N S15 = O S16 = P
S17 = Q S18 = R S19 = S S20 = T
S21 = U S22 = V S23 = W S24 = X
S25 = Y S26 = Z
S27 = F

Thus,

G = {VT, VN, S, P}
VT = {a,b,c,d,e,f,g,h,I,j,k,l,m,n,o,p,q,r,s,t,u,v,w,∅ }
VN = {A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z}
S = S

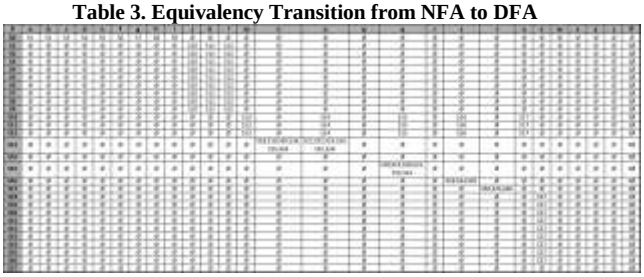
P = { S → aA | bB | cC | dD | eE | fF | gG | hH | iI | ∅,
A → jJ | kK | lL | ∅
B → jJ | kK | lL | ∅
C → jJ | kK | lL | ∅
D → jJ | kK | lL | ∅
E → jJ | kK | lL | ∅
F → jJ | kK | lL | ∅
G → jJ | kK | lL | ∅
H → jJ | kK | lL | ∅
I → jJ | kK | lL | ∅
J → mM | oN | qO | sP | uQ | ∅
K → mM | oN | qO | sP | uQ | ∅
L → mM | oN | qO | sP | uQ | ∅
M → nR | nS | nT | nX | nY | nZ | ∅
N → pU | pV | pW | pX | pY | pZ | ∅
O → rR | rS | rT | rU | rV | rW | ∅
P → tS | tV | tY | ∅
Q → nT | vW | vZ | ∅
R → w | S
S → w | S
T → w | S
U → w | S
V → w | S
W → w | S
X → w | S
Y → w | S
Z → w | S

4.4 Table of Transition Equivalences

Basic concept of Finite State Automata (FSA) provide insight into fundamental that when a system is in a state of a state that moves between state-state that no principle can be produced depends on the input to the system. It is confirmed that the input (input) is true and valid is crucial for conditioned value without error. Implementation of this concept can be seen in modeling compilation or translation of high-level programming language (high level language) into machine language equivalent.

From Non-Deterministic Finite Automata (NFA) machine, which had previously modeled, then the machine can be made Deterministic Finite Automata (DFA) equivalent. Equivalent is meant accords or able to accept the same language. In the case of this study, the transition equivalence can be seen in Table 4.2. The table contains a description of all State with input option (input) respectively to the state-state is written.

Table 3. Equivalency Transition from NFA to DFA



4.5 FSA- DFA Transition Diagram

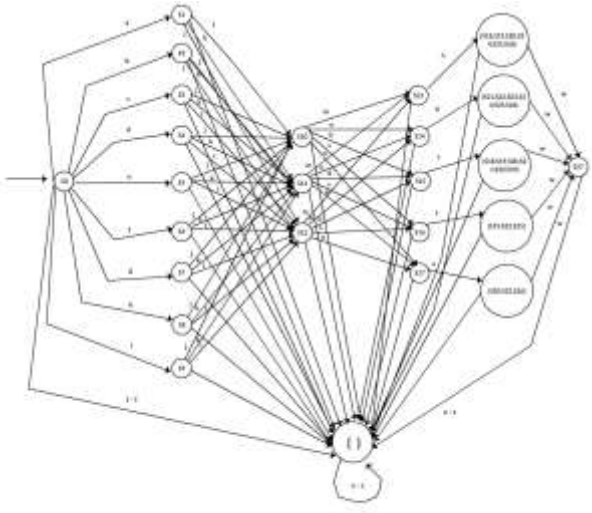


Figure 5. ***Finite State Automata – NFA*** **Transition Diagram**

In Figure 5 can be explained that the FSA began work on the State S0, from S0 option A to S1, b to S2, c to S3, d to S4, e to S5, f to S6, g to S7, h to S8, i to S9, j - z State to an empty ({}). Then, from the State S1 to S9 to S10 option j, k and l to S11 to S12, to a - i and m - z State to an empty ({}).

Furthermore, from State S10, S11, and S12 option m to S13, o to S14, q to S15, s to S16 and u to S17, for a - l, n, p, r, t, v - z to State empty ,

Afterwards, from State to State S13 option n new ({SA18, S19, S20, S24, S25, S26}). For a - v, x - z to State empty. From State S14 selection to p ({S21, S22, S23, S24, S25, S26}). For a - o, q - z to State empty. From State S15 selection to r ({S19, S20, S21, S22, S23}). For a - q, s - z to State empty. State S16 selection of t to ({S19, S22, S25}). For a - s, v - z to State empty. From State S17 to choice v ({S20, S23, S26}). For a - v, w - z to State empty. Of the newly formed State 5 w option to S27. For a - v, x - z to State empty. And for all the existing state S0 option other than 0 to S0.

4.6 Application Prototype

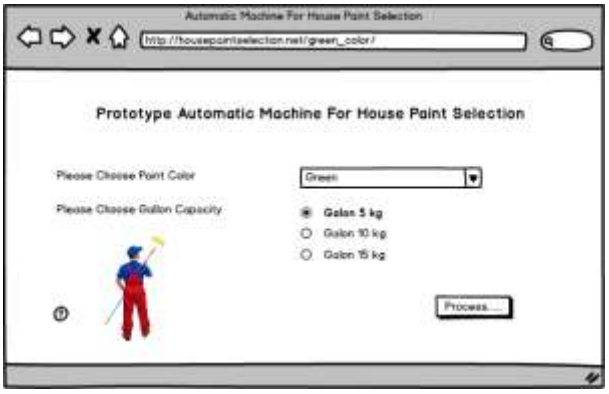


Figure 6. Prototype of Paint Color Selection: Green

Figure 6 illustrates the main interface where the user can immediately make paint color selection, and also the capacity of gallons of paint in accordance with the wishes or needs. In this example, the user make the selection with the green color gallon capacity is 5kg, and a button to execute the process of mixing paint.

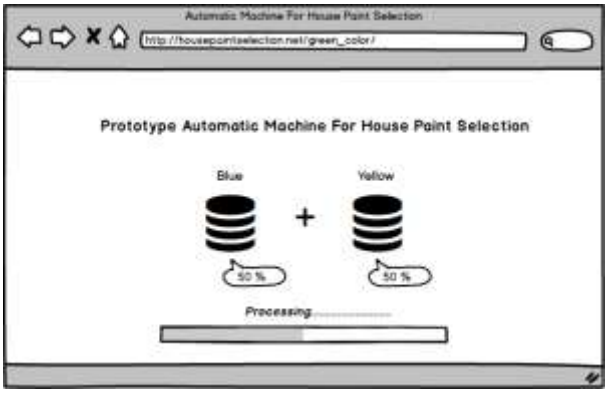


Figure 7. Prototype of Automated Paint Mixing Process: Green

Meanwhile, in Figure 7 can be seen that the application displays the information about the process through simulation where the system will demonstrate the process of stirring or mixing paint, the color blue (the portion of the composition 50%) plus Yellow (the portion of the composition of 50%) to produce a green color.

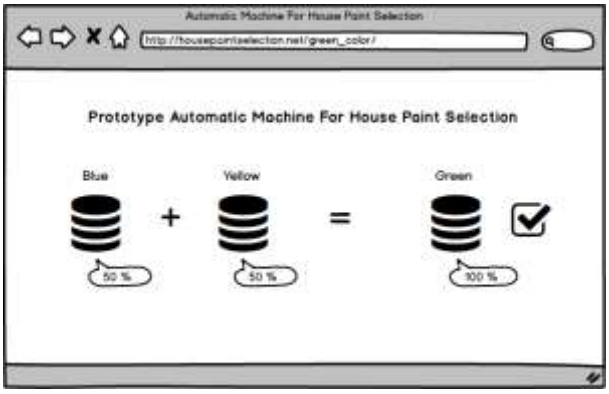


Figure 8. Prototype of Automated Paint Mixing Final Result: Green

Figure 8 is a display end result of the process of stirring or mixing colors to produce a green color. In this application also displays the composition of each color to help users determine the proportion of each paint.

4.7 Prototype Testing

As an important part of assessing the success of a design prototype, so in this study is also testing a system to produce valid and reliable. There are several methods in testing of the prototype as black box testing, white box testing, or usability testing. But in this study using black box testing, the testing model by observing the process inputs (input) and output (output) in the system software to ignore what is going on in the system.

Tabel 4. Blackbox Prototype Testing

No	Indicators	Scenario	Expected Output	Obtained Output
1	Form Design	User run applications from the beginning to the end (to get the paint color desirably)	• Colors selection form • Paint mixing process form • Result paint selection form	• Acceptable • Acceptable • Acceptable
2	Navigation	User run applications from the beginning to the end (to get the paint color desirably)	*Button* process abd navigation system for each forms	Acceptable
3	Respon	User run applications from the beginning to the end (to get the paint color desirably)	Duration of response of applications ranging from input to final results	Acceptable
4	Transition Diagram	User run applications from the beginning to the end (to get the paint color desirably)	Concurrence between output with grammar and FSA Transition Diagram - NDFA	Acceptable

5. CONCLUSION

Based on the problems and objectives of this study, it can be delivered several conclusions as follows:

1. This research resulted in Diagram Finite State Automata - Non Deterministic Finite State Automata, Production Rules, Transition Equivalence, and Diagrams Finite State Automata - Deterministic Finite State Automata, also designed a prototype of an automatic machine simulation application selection of paint house as one of the models implementation.
2. Based on the results of the testing of a prototype application, it can be seen that the concept of Finite State Automata (FSA) used in designing an automatic machine simulation application selection with house paint to generate a grammar (grammar) to produce nine color options can house paint proved successful according to the research objectives.

REFERENCES

[1] Melly, R. I. (2012). Penerapan Konsep Finate State Automata (FSA) Pada Mesin Pembuat Kopi Otomatis. *Jurnal komputasi , 1*, 95-102.
[2] Linz, P. (2001). *An Introduction to Formal Language and Automata* . John and Bartlett Publisher.
[3] Utdirartatmo, F. (2005). *Teori Bahasa dan Otomata.* Yogyakarta: Graha Ilmu.
[4] Kelley, D. (1999). *Otomata dan Bahasa-bahasa Formal.* Prenhanindo.
[5] Juarna, A. (2008). *Teori Bahasa dan Automata.*
[6] Ullman, J. E. (1979). *Introduction to Automata Theory, Languages & Computation.* Addison Wessley Inc.
[7] Ulman, A. A. (1972). *The Theory of Parsing, Translation and Compilling* (Vol. I). Prentice Hall.
[8] Melly, R. I. (2012). Penerapan Konsep Finate State Automata (FSA) Pada Mesin Pembuat Kopi Otomatis. *Jurnal komputasi , 1*, 95-102.
[9] Suprihatin. (2013). *Finite State Automata Untuk Parsing Pemenggalan Suku Kata Dalam Bahasa Indonesia.* Yogyakarta: Universitas Negeri Yogyakarta.
[10] Dihya, A. (2010). *Penerapan Graf Transisi dalam Mendefinisikan Bahasa Formal.* Bandung.
[11] McLeod, R. &. (2004). *Sistem Informasi Manajemen Edisi Kedelapan* . Jakarta: PT Indeks.
[12] Raharjo, B. (1998). *Penggunaan Formal Methods dalam Desain Perangkat Keras* . Pusat Penelitian Antar Universitas bidang Mikroelektronika (PPAU ME) Institut Teknologi Bandung.