FuzzyStudio

A Web Tool for Modeling and Simulation of Fuzzy Systems

Marcelo de Souza[1], Fernando dos Santos[2], Adilson Vahldick[1]
Universidade do Estado de Santa Catarina - UDESC
Ibirama, Brasil
[1]{bsi.marcelo, adilsonv77}@gmail.com
[2]fernando.santos@udesc.br

Adolfo Rodríguez de Soto
Escuela de Ingenierías Industrial e Informática
Universidad de León
León, España
adolfo.rdesoto@unileon.es

***Abstract*—The study of Fuzzy Logic in higher education is hampered by the limited time devoted to it, since it is frequently embedded into a discipline along with other topics. In spite of their potential for application in several areas of knowledge, such as education, administration and engineering, teaching Fuzzy Logic is also restricted by the absence of an environment to easily develop fuzzy systems. This work deals with the development of a tool for modeling and simulating fuzzy systems. The tool aims to facilitate the fuzzy systems building process. As an important feature, an execution environment to the model has been developed. This feature improves graphical execution, FCL standard code generation and Java code generation. Furthermore, the tool enables collaborative work, allowing that more than one user simultaneously work on the same project. By now, tool supports Mamdani-type fuzzy systems and its implementation is exploited with the jFuzzyLogic library. The tool was used in a class of Artificial Intelligence in the Santa Catarina State University. In the opportunity, the students performed exercises on the platform and also delivered a work proposed by the professor. After that it was conducted a survey with those students in order to evaluate the facility of using the software. The system showed to be efficient and simple for the construction of fuzzy systems as well as for the simulation and generation of fuzzy artifacts. So, it can be applied in an academic environment to practice fuzzy theories, as well as in the commercial context, in the construction of intelligent solutions that use Fuzzy Logic.**

Keywords—Fuzzy Logic; Fuzzy Systems; modeling; Fuzzy Control Language; jFuzzyLogic

I. Introduction

Fuzzy Logic was developed in the 60s by Lotfi Asker Zadeh [1], aiming to represent classes or sets with imprecise boundaries. Fuzzy Set Theory is able to model the uncertain and imprecise knowledge, used by humans everyday [2]. Consequently, Fuzzy Logic has rapidly grown in recent decades and expanded their use in many areas as artificial intelligence systems, industrial applications, control systems, and many others. Nowadays, it is common to see applications that make use of fuzzy concepts. Facing this context, Fuzzy Logic is an important field of study in higher education, particularly those involved with computing. This is not the view commonly found. Much of higher courses present the field into a discipline, like Artificial Intelligence. This impairs learning, given the limited time dedicated to it. In other words, the reserved time for teaching Fuzzy Logic is just enough to cover the general theoretical concepts. The students do not have the possibility to see a Fuzzy Logic based system in practical environments.

A solution to this problem is the use of tools that help the student to work with fuzzy systems and understand their structure and operation. Within this approach, another difficult emerges. The tools currently available have many impediments for a successful use in an academic environment. Some solutions are obtained upon payment and many tools are very difficult to use without advanced knowledge in Fuzzy Logic. Furthermore, the majority of solutions have lack of functionality.

The present paper presents the development of a tool to modeling and executing fuzzy system that aims to be used by students and researchers to learn Fuzzy Logic. To achieve this goal, the tool provides source code and graphs generation and collaborative work. A table with the features and the comparison with other solutions is presented as results. Finally, the paper seeks to verify the effectiveness of the solution in academic environment by the application with a group of students.

This paper is organized as follows. In Sect. 2 we present the basic theory of Fuzzy Logic and introduce some concepts about fuzzy systems. In Sect. 3 we present the jFuzzyLogic [3] library and the language provided by it. Related work is described in Sect. 4. The developed tool is described in Sect. 5. We discuss the experiments and analyzes the results in Sect. 6. Final remarks and future directions are presented in Sect. 7.

II. Fuzzy Logic and Systems

In this section we introduce some concepts about Fuzzy Logic and Fuzzy Systems. This paper does not address fuzzy theory in depth. The complete formalism and further theory can be found in [4-7]. Researchers conceptualize logic as the science to study the laws of reasoning [8]. Fuzzy Logic is concerned to study the principles of approximate reasoning. This field came in 1965, via Lotfi Asker Zadeh [1] by proposing a new perspective on the representation of a given state, seeking to represent imprecise and vague situations. In short, Fuzzy Logic introduces significant new capabilities in computer systems, since it allows the representation of imprecision and vague concepts, both characteristics of real world.

The traditional logic uses two truth values for a given proposition. One proposition is either true or false, i.e., it has a truth degree of one or zero. Mathematical structures as Boolean Algebra are built around this principle, representing the world laws of truth in computational environment [9]. Job [10] discusses the fact that often certain situation does not clearly be true or false. This imprecision is treated under different techniques, one of them is Fuzzy Logic. For the treatment of imprecision or vagueness of data, Fuzzy Logic proposes the concept of polyvalence. In this sense, a given proposition can have truth values in the range [0, 1], [1].

Through this approach, it is assumed that a particular proposition has a degree of relevance to a given set. This grade is expressed as a numerical value in the range [0, 1] and enables treating imprecision

978-1-4799-5618-0/14 $31.00 © 2014 IEEE
DOI 10.1109/BRACIS.2014.62

with greater accuracy through the use of set theory in mathematics. This potential can be explained by the fact that, for example, the person's height may belong with a certain degree of relevance to the set of "short" people. This does not exclude the possibility of belonging to the group of "medium" and "tall" people, with their respective degrees. So, the vagueness and imprecision characteristics of different areas of knowledge can be treated with greater fidelity in computer systems [10].

Based on this reasoning, the systems that use this kind of data processing are known as fuzzy systems. We can classify the fuzzy systems into two main models: Mamdani-type and Sugeno-type systems. The main difference between the two approaches is in the deffuzification process [11]. Mamdani-type fuzzy systems use fuzzy sets on the consequent part of rules, while Sugeno-type fuzzy systems use numbers or linear functions. Owe to that, defuzzification process is quite different. Mamdani-type systems need a defuzzification method and Sugeno-type systems just use weighted average to compute crisp outputs. In short, Mamdani-type is more human-like and more intuitive, Sugeno-type is computationally efficient and flexible. A more detailed comparison is presented in [11]. This paper only considers rule-based fuzzy system of Mamdani-type. Like some of intelligent systems, a rule-based fuzzy system aims to modeling actions according to the expert knowledge [12]. To achieve this goal, the fuzzy systems operate in three distinct moments. In the first, the input crisp values are transformed to fuzzy values. With these fuzzy magnitudes, the system performs some inferences, based on the modeled rules. Finally, the outcome values are translated to crisp numbers. This process is expressed by Fig. 1.

As we can see in Fig. 1, we can divide the fuzzy system into four basic components, called functional blocks. The first block is the fuzzification interface, that translate the crisp numbers to fuzzy values. These translated values are used by the next functional block, called inference machine. The inference machine collects the fuzzy values and, by the rule base (third block), calculates the output fuzzy values. Last functional block is called defuzzification interface and their task is the inverse of fuzzification process. In other words, the defuzzification block translates the fuzzy outcomes into crisp numbers [9].

A. Fuzzification and Membership Functions

To translate the crisp inputs to fuzzy values, the fuzzy systems use membership functions. According to [9], a membership function is a function, graphical or tabulated, that assigns a fuzzy membership value to a discrete value of one variable. In other words, a discrete value of a variable is transformed to a membership value, related to the modeled fuzzy set [13]. The membership function performs a mapping from the given domain to the interval [0, 1]. This process is called fuzzification. Possibility theory, provided from the concept of fuzzy sets and membership functions, cannot be interpreted as probability theory. The probability is additive, while the possibility is maxitive.

The membership functions are determined based on the experience in the problem domain. The greater the number of sets becomes better representation. However higher computational power is demanded. Generally, common membership functions are: triangular, trapezoidal and Gaussian function. A smoother representation can be obtained using more complex functions, which increases the computational requirements. It is possible to embody in fuzzy systems the precision of classical logic by the use of Singleton functions or pulse functions [14]. This approach is called discrete functions and assigns the value one to the membership degree, when satisfied.

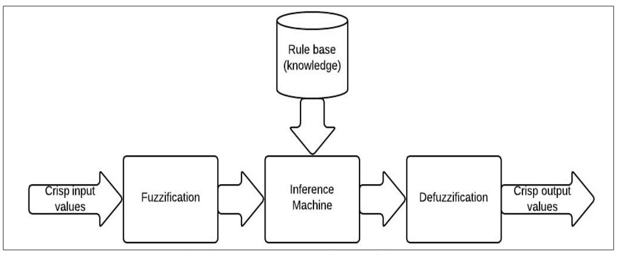

Fig. 1. Architecture of a fuzzy system (Adapted from [9])

B. Inference Machine and Rule Base

The inference machine processes fuzzy values. It uses the rule base to calculate the outcomes, as shown in Fig. 1. In short, the inference machine reflects the logic of the system, while the rule base can be seen as the system knowledge. Generally, the rule base is developed by a domain expert, which uses measurements and tests to model and to determine the fuzzy rules. According to [15], the success of the fuzzy system is determined by the quality of its rules.

A rule is expressed by a conditional structured of antecedent and consequent. Antecedents are conditions of the real world that can activate the rule. When activated, the consequent is performed as an action. In short, a rule can be expressed as IF *antecedent THEN consequent* [16]. Some characteristics must be satisfied in the rules modeling. The rule base must be consistent, without contradictions. The rule base must be complete, for each relation between input variable and linguistic term exist a rule covering this condition.

With the control actions modeled by the rules, the inference machine can calculate the outcomes. The processing of the rules follows the *modus ponens* model. A real value to each premise is computed and applied to the consequent, resulting in a fuzzy set to each output variable of each rule. In the fuzzy sets theory, the intersection task is performed by a set of operators called *t-norms*. To the union task, the set of operators is called *t-conorms* [17]. In Mamdani-type fuzzy systems, the MIN *t-norm* operator is used to make the inference [1], although there are studies that suggest the use of other operators with the same effect. In Mamdani-type fuzzy systems, the *t-conorm* is also used to aggregate the partial results prouced by rules [17]. Usually, to make the rule aggregation the operators MAX and SUM are used [18].

C. Defuzzification

The defuzzification stage is the process of transformation of the output fuzzy values in crisp numbers. In other words, the result of the fuzzy processing is expressed in magnitudes of the real world [8]. There are several methods of defuzzification in fuzzy systems. Some are based on the centroid of the outcomes and others in the maximum values resulted of the membership functions [9]. The most common methods are: Center of Gravity, Mean of Maximum and Center of Maximum.

The Center of Gravity is the most used function to defuzzification process in fuzzy solutions. It calculates the centroid of the area composed by the fuzzy outputs. This area is determined by the union of the output of all rules [16]. In short, the centroid value is the measurement of the center of gravity of the area under the curve. Fig. 2 expresses the behavior of the Center of Gravity defuzzification method.

III. JFUZZYLOGIC LIBRARY AND FUZZY CONTROL LANGUAGE

JFuzzyLogic, according to [3], is an open source library written in Java that allows the design of controllers based on Fuzzy Logic. The library follows the specification proposed by IEC in the 1131-7 norm [19], that defines the standard notation of fuzzy controllers. The main goal of jFuzzyLogic is to bring the benefits of an open source platform and provide a standardization of fuzzy systems to the developers community. Furthermore, the library supports extensions to the API, allowing the easy development of new features or other applications

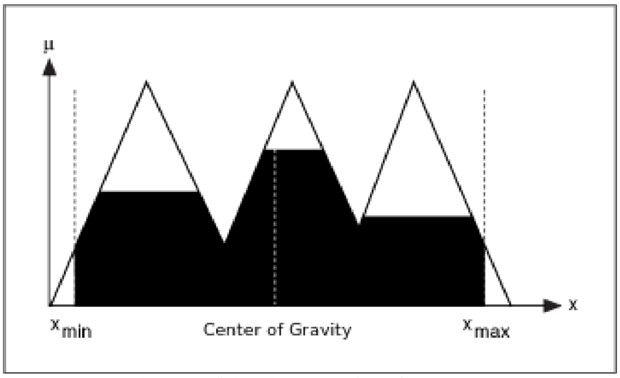

Fig. 2. Center of Gravity defuzzification (Adapted from [16])

based on their structure. Another issue related to the library choice is the independence of platform. The library was developed in Java, allowing its use in applications that runs in any hardware and software platform that supports Java.

JFuzzyLogic implements the fuzzy inference machines according to the Mamdani model. It delivers a varied collection of membership functions, many rule aggregation and activation techniques, and defuzzification methods. The importance of the library in this paper consists in the fact that it follows the IEC 1131-7 [19] and implements the Fuzzy Control Language (FCL). This language is described in [19] and consists in a standard created to define the concepts and implementation of fuzzy controllers. The main goal of the FCL is to allow the portability of these controllers to different platforms. Furthermore, the standard ensures the scalability of the application, i.e., a fuzzy model, developed following these specifications, can be used by software and hardware of distinct platforms and sizes.

IV. RELATED WORK

We found many software solutions to modeling and executing fuzzy systems. Many of them aim to be an alternative to learn about the field. A popular solution is Matlab [20]. Matlab is a flexible tool to mathematical calculations, modeling, simulations, graphs and numerical analysis. According to [21], Matlab is largely used in universities, especially in the area of exact sciences and engineering. It haves a collection of toolboxes to complement the available features, one of them is the Fuzzy Toolbox, to construct fuzzy models and systems.

The Matlab Fuzzy Toolbox offers a graphical interface to the creation of fuzzy systems. The tool is divided into five groups of features. The first one is dedicated to the edition of the inference system, where the user can build the input and output variables. The second group defines the membership functions of the variables. The third presents the rule base editor. The fourth is composed by the visualization of the rules, allowing the verification of all components and their influence in the results. Finally, the fifth group presents the surface visualizer, showing the surface graphs generated by the execution of the model, if the number of dimensions allows it [20].

In [22] the InFuzzy tool is presented, showing their features for fuzzy modeling, creation and execution. This solution presents a design interface, through which is possible insert, update and delete the variables and rules of the inference machine. In the variable editor, the fuzzy sets and membership functions are defined. InFuzzy allows the creation of many rule bases in the same model, facilitating the task of testing. Furthermore, the system allows the configuration of the implication and aggregation methods, defuzzification method and operators AND and OR. Other interesting feature of this solution is the simulation of multiple input values, i.e., it is possible to perform more than one simulation at the same time and compare its results. InFuzzy presents to the user a textual description about the model and the components used to do the inference. In this case, the user can see the parts of the model that influences the outcome and adjusts them. Finally, a last feature is the possibility of integration and communication with an external controller or application by network, using UDP protocol.

The last related work is the FuzzyGen software, developed by Embrapa [23] to embody the expert knowledge in its system. FuzzyGen is developed in Java and, despite the portability provided by the language, it presented problems running in different operating systems. To overcome this limitation, the organization developed a second software called WebFuzzy, providing a Web environment to execute the built fuzzy controllers [24]. In the construction of the fuzzy system, FuzzyGen allows the definition of aggregation and implication methods linked to the output variables. The rule base is unique to entire system as well as its connectors. The main benefits are the ease of use and the FCL generation. The tool is resumed in three tabs: modeling, inference and graphs. In the modeling tab the variables, functions and rules are edited. In inference tab the user can specify the input values and calculate the results. In the third tab are shown the graphs of the model [24].

In short, we can list the deficiencies that preclude the use in academic environments: difficult of access, platform dependency, difficult in the use, impossibility of code generation, lack of standardization and impossibility of collaborative work. These are the main features that encouraged the development of the FuzzyStudio tool. A more detailed comparison between the related works and the FuzzyStudio software is presented in the next section.

V. FUZZYSTUDIO TOOL

The idea of the solution presented here was designed based on the scenario faced by students of computing. The studied tools have some shortcomings, which discourage the development of systems based on Fuzzy Logic. These problems result in the impossibility of use of this software in academic environment, i.e., we cannot solve the difficult of learning fuzzy systems, occasioned by the short time devoted to the subject. In short, the FuzzyStudio was planned to serve as a tool for practical application of Fuzzy Logic and Systems.

Based on this context, to achieve the goal of presenting a tool to be used in academic environment, the solution here developed must be easy to use, providing agility in the modeling of fuzzy systems and the possibility of validation of the model through its execution. The features of the FuzzyStudio software were delimited based on the analysis of the related work. The tool can be considered a shell, i.e., the user is concerned with the definition of the expert knowledge. All the inference, calculation and interpretation tasks are responsibility of the system [25] [26]. In resume, the shell allows edition of the basic components of the fuzzy model, i.e., the input and output variables, linguistic terms, rules and inference machine. Furthermore, the solution presents an execution module, showing the outcome values based on the input values, reported by the user. Finally, some differentials were approached, like the appropriateness of IEC 1131-7 standard and FCL, generation of graphs and code in programming language and collaborative work.

A. Construction of the Model

The shell is based on the Mamdani fuzzy model, which implies that the system have a collection of output variables, modeled as a fuzzy set. To build a Mamdani model, the modeling stage is composed by the edition of the input and output variables, the edition of the rule base, the inference machine and general configurations. To each variable, the user can define the associated linguistic terms and membership functions. The interface presents the graph for each variable and the user can adjust the values of the membership function parameters to achieve its goals. Fig. 3 shows the variable edition interface.

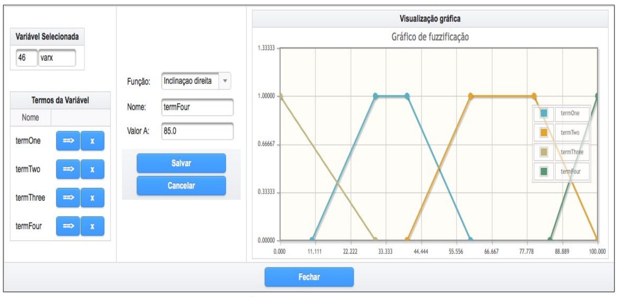

Fig. 3. Interface of variable edition

Also in the modeling step, the user defines the system rules. In this interface each variable is presented, the user must only choose the fuzzy sets associated to each variable and the rule connector. In the same interface the user can verify all the created rules. Fig. 4 shows the respective interface.

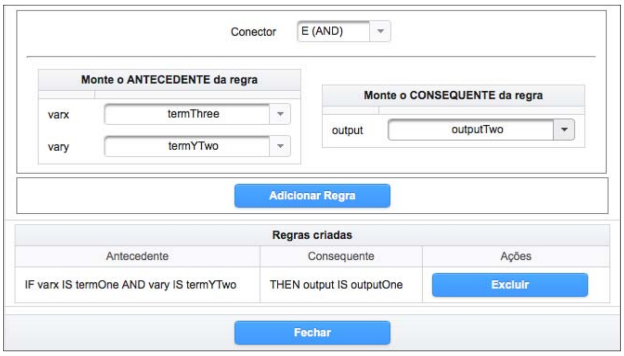

Fig. 4. Interface of rule base edition

The configuration of the fuzzy model is performed in the inference machine editor, as shown in Fig. 5.

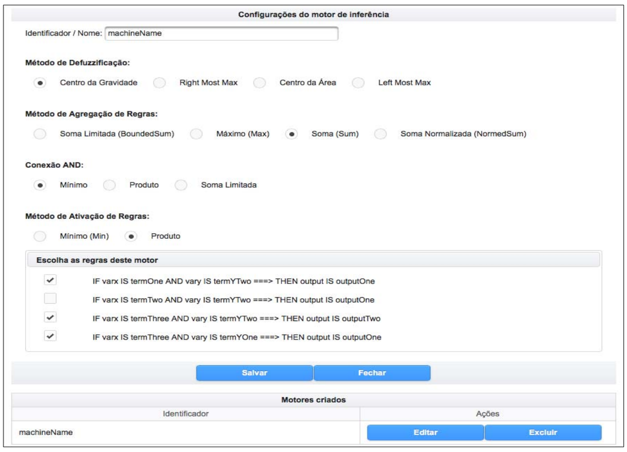

Fig. 5. Interface of inference machine edition

The user can create many inference machines for the system, associating the rules to this set of parameters. The benefits of this approach are presented in the next section. For each inference machine, the user can choose between four defuzzification methods: Center of Gravity; Center of Area; Right Most Max and Left Most Max. The FuzzyStudio implements four rule aggregation methods: Bounded Sum; Maximum; Sum and Normed Sum. With respect to the *t-norm* operators, the tool implements the Minimum, Product and Lukasiewicz t-norm; and with respect to *t-conorm* operators, the tool implements the Maximum and Probabilistic sum functions.

Additionally, while the components are created, the tool presents a design interface with graphical representations of the components. This facilitates macro view of the built model structure. The design component is shown in Fig. 6. Through it the user can also edit the shown components. In this moment, the user has created the system variables and its linguistic terms, the rule base and the inference machine, with the required configuration to execute the model.

B. *Execution of the Model*

The execution of the inference machine is the kernel of the shell. As discussed above, the tool allows the rule definition and creation of many inference machines to the same fuzzy model. In this case, the user can link each rule to each model, according to his desires (Fig. 5). In other words, one inference machine does not use all rules, but those chosen by the user. The configuration of the fuzzy system (defuzzification, aggregation and accumulation methods and connectors) is linked to the inference machine, i.e., each inference machine has only a set of all the rules and a specific configuration. Fig. 7 shows this independence in the execution of each machine.

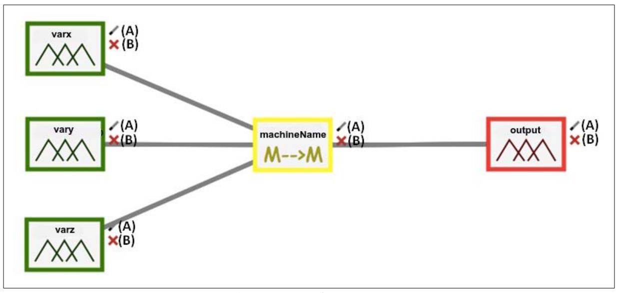

Fig. 6. Design Component of FuzzyStudio

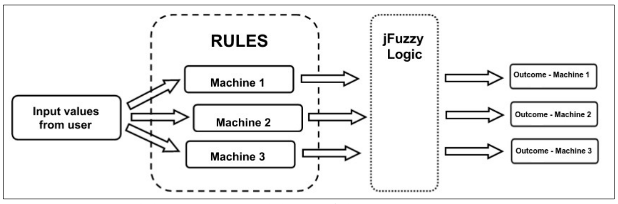

Fig. 7. Stages of the execution of the model

Decomposing the execution process in more detail, in the first step the inference machine captures its configuration parameters and the input values defined by the user. The linked rules are retrieved and, through them, the variables that comprise the system. With this data collection, the FCL notation is built by the shell. This code is sent to the jFuzzyLogic library, which is responsible for processing the data, calculating the outcomes and showing them. These results are presented to the user. With this implemented scenario, the conduction of tests and measurements is facilitated and it is possible to compare many fuzzy models into the same project. In other words, this feature enables the possibility to create many fuzzy systems into the same model. In the execution, the user supplies the input values and each inference machine shows its results.

C. *Artifacts Generation and Collaborative Work*

The artifacts generation configures a differential of the system and an important feature to achieve their goals. The tool is based on three types of results: representative graphs of the system model, notation in FCL standard and programming code in Java language. For the graphs generation, the resources provided by jFuzzyLogic library are used. For each variable of the model, a graph that shows the linguistic terms and the value informed by user or calculated by the shell is plotted. Specifically to the output variables, the system shows the defuzzification graphs. Through them, the user is able to analyze the behavior of the modeled system and perform adjustments.

The FCL standard generation is performed in the execution stage. The FuzzyStudio retrieves from the database each created component, constructing the model that represents the retrieved structure. In the standard, the fuzzy model is represented by three data groups. The first one represents the names and types of the input and output variables. The second group configures the fuzzification and defuzzification values, membership functions, and linguistic terms. The third group is composed by the rule block, which groups all rules and defines the methods for the fuzzy machine engine. Fig. 8 presents the output values generated to each inference machine of the model and the FCL code.

As we can see in Fig. 8, the user can introduce the input values (a) to each variable. The tool executes the fuzzy model independently for each inference machine, showing their correspondent results (a). For each inference machine the tool generates the FCL (b) and Java codes and the output graphs. Fig. 9 shows an example of generated graphs.

Finally, the tool generates Java code, devoted to users wishing to develop a system or a controller that uses the constructed model. In short, the system generates a Java class, that uses the jFuzzyLogic library and the FCL notation. Fig. 10 presents an excerpt of the generated code, illustrating this feature. In the execution of this class, the input values are requested to the user. The model, represented by FCL language, and the input values are sent to the library, which returns the inferred output values. The outcomes are presented by means of graphs.

FuzzyStudio software was designed to run on the Web, facilitating access to the tool. Therefore, it has control of users and requires authentication to access. This feature allows the user to keep your projects on the server and access them when needed. The tool has free access on the Web. In order to increase the use in academic environments, the tool allows to share a project with other users. Many students can work in the same project simultaneously. This feature allows group works, or monitoring of a work by a professor.

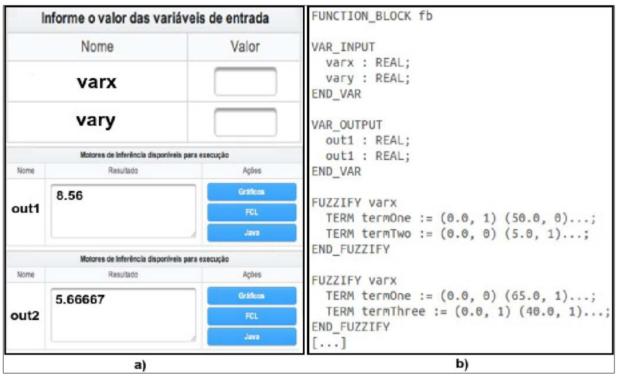

Fig. 8. *Examples of the outcome artifacts*

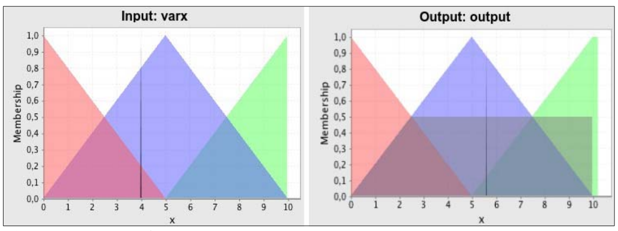

Fig. 9. *Generated Graphs*

```java
public class FuzzyStudioFCL {
    public static void main(String[] args) throws RecognitionException {
        Srting fcl = "FUNCTION_BLOCK mamdani \n"+ […] + "END_FUNCTION_BLOCK\n";

        FIS fis = FIS.createFromString(fcl, true);
        FunctionBlock functionBlock = fis.getFunctionBlock("mamdani");
        functionBlock.setVariable("xvar", Double.parseDouble(
                javax.swing.JOptionPane.showInputDialog(
                        "Set a value to variable x")));

        functionBlock.evaluate();
        functionBlock.getVariable("xvar").chart(true);
        functionBlock.getVariable("outputvar").chart(true);
        functionBlock.getVariable("outputvar").chartDefuzzifier(true);
        [...]
}}
```

Fig. 10. *Example of Java generated code*

VI. Experiments and Results

After the development of the FuzzyStudio tool, some experiments were conducted. The first was composed by the simultaneous access of two users. Many were created projects and shared between the users. In this opportunity, aspects of correction and improvement of the system were raised. The second experiment dealt with the application of the tool in a class of Artificial Intelligence, at the course of Information Systems of the State University of Santa Catarina. Firstly all students complete their registration on the environment. Then they modeled a fuzzy system provided by the professor. No instructions for use of the tool were provided, seeking to observe the ease in learning to use the software without needing help. All students were able to complete the modeling exercise and showed no difficulties with the software. FuzzyStudio tool presented good response time and no errors in the execution.

After this stage of the experiment, a complete homework was given to the students, consisting in a modeling, execution and generation of artifacts of a complete fuzzy model presented in [27]. Again, the solution presented good performance and no errors. At the end of this step, a survey was applied to the students, aiming to identify the facility of use related to the environment resources. In this respect, the survey was divided into fifteen activities (variable edition, rule creation, model execution, etc.) of the system. For each activity, the user answers with their satisfaction related with its facility in the use, ranging from completely dissatisfied to completely satisfied.

According to the results of the questionnaire, the users showed no major difficulties while using the software. The authentication and project creation were appointed as easier activities, while the rule creation were appointed as the activity with the less satisfaction degree. Generally, the questionnaire results and the mediated use in the class identify the FuzzyStudio as an easy to use tool. This is important, given the academic nature of the project.

As results of work, the developed tool presents some differentials from other solutions, which allow the use by inexperienced users and students. As main differentials we can mention: the Web platform of execution, ensuring the portability of the system; the possibility of creation of many inference machines and their independent and parallel execution; the artifacts generation and the collaborative work. These features make the solution closest to the needs of academic or beginner group in the area. Regarding the related work, a comparative table has been produced, showing some features of the studied solutions and the new features implemented by FuzzyStudio. This comparison can be observed in Table 1.

As we can see, some features presented only by FuzzyStudio make this solution a good candidate to application with students, aiming to learn about Fuzzy Logic. FuzzyStudio is the only solution that is executed in Web platform. This feature allows the complete portability into any operating system. The possibility of creation of multiple inference machines allows the definition of some fuzzy models into the same project and facilitates the test stage. The generation of Java code facilitates the development of a software using the created model. Finally, the collaborative work makes easy the

TABLE I. COMPARATIVE OF THE FEATURES IN RELATED WORK

Feature	*Matlab Fuzzy Toolbox*	*In Fuzzy*	*Fuzzy Gen*	*Fuzzy Studio*
Multiple inference machines				x
Programming code generation				x
Collaborative Work				x
Generation of FCL notation			x	x
Design interface of the fuzzy model	x	x		x
Export the model design		x		x
Execution of the system	x	x	x	x
Generation of graphs	x	x	x	x
Integration with other applications	x	x		
Multi-value simulation		x	x	
Possibility of set the *t-norm* and *t-conorm* operators	x	x		x
Visualization of the activated rules in the simulation	x	x		
Step debugging	x			
Surface graph generation	x			
Assigning weight to the rules	x		x	
Free distribution		x	x	x

group task and cooperative work in the same project, ideal for work and study teams.

VII. CONCLUSIONS

The first goal of this work was achieved: the development of a tool that met the deficiencies identified in the studied solutions. The developed features allow the use by inexperienced people, such as students. The tool permits the construction of a fuzzy model, execution of the system, setting of parameters and values and generation of model artifacts. The differentials previously detailed make the FuzzyStudio a better applicable tool in academic environments. Among the most important features to fill the gaps currently faced we mention: ease of use; FCL standard implementation and generation; Java code generation and collaborative work.

By the application of the tool in a student group, we identified the effectiveness of the software in this environment. The use of FuzzyStudio in learning allows the application of the fuzzy theoretical concepts in a practical activity and possibilities the visualization of the results derived from the use of a solution based on Fuzzy Logic. In other words, the students can to apply the concepts in a practical work and better understand the operation of a fuzzy controller and their components. In short, the FuzzyStudio tool facilitates the teaching of fuzzy systems in higher education and it was effective in the tasks for which it was intended. As future work, it is intended to extend the tool, providing such features as: new membership functions, allow the construction of models based on Sugeno-type fuzzy systems, assigning weights to the rules and debugging routines in the model execution. Furthermore, a more detailed comparison will be performed between the FuzzyStudio tool and the FuzzyGen software, since the two solutions present similar features.

REFERENCES

[1] L. A. Zadeh, “Fuzzy Sets”, Information and Control, vol. 8, pp. 338-353, 1965.

[2] T. J. Ross, Fuzzy Logic With Engineering Applications, 2nd ed. Chichester: John Wiley & Sons, 2004.

[3] P. Cingolani, and J. Alcalá-Fdez, “jFuzzyLogic: A Robust and Flexible Fuzzy-Logic Inference System Language Implementation”, WCCI 2012 IEEE World Congress on Computational Intelligence. Brisbane, pp. 1090-1097, 2012.

[4] D. Dubois, and H. Prade, Fuzzy Sets and Systems: Theory and Applications. New York: Academic Press, 1980.

[5] F. M. McNeill, and E. Thro, Fuzzy Logic: A Practical Approach. New York: AP Professional, 1994.

[6] H. T. Nguyen, and E. A. Walker, A First Course in Fuzzy Logic, 3rd ed. Boca Raton: Taylor & Francis Group, 2006.

[7] G. Bojadziev, and M. Bojadziev, Fuzzy Seets, Fuzzy Logic, Applications. Singapore: World Scientific Publishing, 1995.

[8] H. A. Oliveira Jr. et al., Inteligência Computacional Aplicada à Administração, Economia e Engenharia em Matlab. São Paulo: Thomson Learning, 2007.

[9] M. G. Simões, and I. S. Shaw, Controle e Modelagem Fuzzy, 2nd ed. São Paulo: Blucher, 2007.

[10] B. Coppin, Inteligência Artificial. Rio de Janeiro: LTC, 2010.

[11] A. Kaur, and A. Kaur, “Comparison of Mamdani-Type and Sugeno-Type Fuzzy Inference Systems for Air Conditioning System”, International Journal of Soft Computing and Engineering (IJSCE), vol. 2 (2), pp. 323 – 325, 2012.

[12] S. J. Russel, and P. Norvig, Artificial Intelligence: A Modern Approach. New Jersey: Prentice Hall, 1995.

[13] J. Jantzen, Foundations of Fuzzy Control. Chichester: John Wiley & Sons, 2007.

[14] J. Espinosa, and J. Wandewalle, and V. Wertz, Fuzzy Logic, Identification and Predictive Control. London: Springer-Verlag, 2005.

[15] Z. Kovacic, and S. Bogdan, Fuzzy Controller Design: Theory and Applications. Boca Raton: Taylor & Francis Group, 2006.

[16] T. J. Ross, Fuzzy Logic With Engineering Applications, 2nd ed. Chichester: John Wiley & Sons, 2004.

[17] S. Sandri, and C. Correa, “Lógica Nebulosa”, V Escola de Redes Neurais, Conselho Nacional de Redes Neurais, pp. c073 – c090, 1999.

[18] L. Weber, and P. A. R. Klein, Aplicação da Lógica Fuzzy em Software e Hardware. Canoas: Ulbra, 2003.

[19] IEC, 1131 – Programmable Controllers: Part 7 – Fuzzy Control Programming, Committee Draft CD 1.0, 1997.

[20] Mathworks, “Fuzzy Logic Toolbox: User's Guide”, Natick: The MathWorks, 2012.

[21] A. Gilat, “Matlab With Engineering Applications”. Porto Alegre: Artmed, 2005.

[22] E. L. Posselt, R. Frozza, and R. F. Molz, “Software InFuzzy 2011”. Programa de Mestrado em Sistemas e Processos Industriais, UNISC, 2011.

[23] Embrapa, “FuzzyGen”, Campinas: Embrapa Informática Agropecuária, 2009.

[24] H. P. Lima, and S. M. F. S. Masshurá, “Sistema FuzzyGen: Manual do Usuário”. Campinas: Embrapa Informática Agropecuária, 2009.

[25] R. Heinzle, “Protótipo de uma Ferramenta para Criação de Sistemas Especialistas Baseados em Regras de Produção”, Florianópolis: UFSC, 1995.

[26] A. B. Alexandre, “Protótipo de um Sistema Especialista Utilizando a Ferramenta Expert Sinta para Auxílio no Setor de Suporte de uma Software House”, Blumenau: FURB, 2000.

[27] J. N. Paula e Sousa, “Aplicação de Lógica Fuzzy em Sistemas de Controle de Tráfego Metropolitano em Rodovias Dotadas de Faixas Exclusivas para Ônibus”, Rio de Janeiro: UFRJ, 2005.