An RVML extension for modeling fuzzy rule bases

N O Dorodnykh, A Y Yurin

Matrosov Institute for System Dynamics and Control Theory, Siberian Branch of Russian Academy of Sciences (ISDCT SB RAS), 134, Lermontov str., Irkutsk, 664033, Russia

tualatin32@mail.ru, iskander@icc.ru

Abstract. Rules are still the most widespread way to represent expert knowledge despite the popularity of semantic technologies. The effective use of rules in decision-making in the case of inaccurate or uncertain information requires the development of specialized means and software for visual and generative programming. This paper considers an extension of the Rule Visual Modeling Language called FuzzyRVML designed for modeling fuzzy rule bases. FuzzyRVML supports a fuzzy datatype, concepts of a linguistic variable, terms, and certainty factors. The descriptions of FuzzyRVML basic elements, main constructions, and an illustrative example containing FuzzyCLIPS source code generation are presented. The evaluation and implementation of this notation are made based on the Personal Knowledge Base Designer software.

1. Introduction

Extensive experience and a wide range of different methods and tools for representing and processing knowledge have been accumulated in the field of artificial intelligence. Despite the popularity of semantic technologies and, in particular, ontologies for knowledge representation, the logical and associative rules stay the most widespread and popular way for description and decision making by domain experts [1]. The attractiveness of this knowledge representation model is due to its simplicity and clarity for experts, high modularity, ease of making changes, and transparency of the inference.

Many programming languages and standards implement this formalism, for example, C Language Integrated Production System (CLIPS) [2], Java Expert System Shell (JESS) [3], Semantic Web Rule Language (SWRL) [4], Drools [5], Rule Interchange Format (RIF) [6], etc. The use of these languages together with their graphical supporting tools could significantly increase the effectiveness of their application.

Graphical or visual programming approaches provide the creation of visual abstractions corresponding to elements of rules, with their subsequent translation into source codes for some knowledge representation language. Currently, many specialized graphical notations provide modeling of logical and cause-effect relationships. However, some of them, for example, VIsual Imperative Programming (VIPR) [7], contain non-standard constructions (artifacts), which is not always intuitive for the developers. In this regard, the extensions or profiles of well-known languages, for example, Unified Modeling Language (UML) [8], are more applicable. UML-Based Rule Modeling Language (URML) [9] and Rule Visual Modeling Language (RVML) [10] are examples of such extensions.

Most of the considered visual languages and their extensions are designed for the representation of explicit knowledge without incompleteness and inaccuracies. However, knowledge used in the

decision-making of real-world practical tasks is often inaccurate or uncertain, for example, in the case of diagnostics of unique technical systems [11].

In this paper, we propose an extension of one of the visual programming languages for modeling logical rules, in particular, RVML [10]. The proposed extension called FuzzyRVML and supports the main elements of the theory of fuzzy logic and sets [12] and can be applied for modeling fuzzy variables and rules, and generating source codes for FuzzyCLIPS [13].

A feature of the extension proposed is the use of separate elements for displaying linguistic (fuzzy) variables and their terms, as well as the integration of new elements with the elements of the basic version of RVML. The advantage of FuzzyRVML is the ability to correctly transform the FuzzyRVML elements to the FuzzyCLIPS language constructs, as well as the clarity of the visual representation of elements that are based on the main UML elements, such as "class" and "relationship".

The paper is organized as follows. Section 2 presents an analytical overview of related works and background. Section 3 describes the extension proposed including its basic elements, supporting software, and an illustrative example, Section 4 contains discussion, while Section 5 presents some concluding remarks.

2. Background and related work

2.1. Modeling of rule bases

Logical and associative rules remain the main technique for formalizing and codifying business logic and knowledge. Based on the classification [14] and its subsequent modification, the following main groups of approaches can be defined (they were implemented in software that supporting the creation of rules):

- Textual approach providing direct manipulation of language constructs. The approach is aimed at programmers, and it implemented in the form of specialized editors.
- Tabular approach is based on the creation of decision tables and their translation into source codes. Both the standard decision table formalism and its specializations, such as eXtended Tabular Trees (XTT2), are used.
- Graphical approach providing the creation of visual elements corresponding to the components of logical rules, with their subsequent translation into source codes. This approach is the most promising because it minimizes manual (hand) coding errors, as well as attracts non-programming users who have visual modeling skills to the development process.

In turn, the graphical approach can be divided into the following ones:

- Using domain-specific notations designed for description of certain domain or task, e.g. event or failure trees that are used in failure and risk analysis. The special software to transform these models is used, in particular [15].
- Using universal semantic graph structures such as concept maps, ontologies, "entity-relationship" diagrams, etc. However, the lack of a generally accepted interpretation of the relationships between concepts when translating such models into logical rules makes it difficult to widely use this approach when creating knowledge bases and expert systems.
- Using extensions or specializations of popular notations that can provide modeling of logical and causal relationships. In this connection, notations that are extensions or profiles of well-known languages, such as Unified Modeling Language (UML), are promising. One such extension that is implemented in tools and has application is the Rule Visual Modeling Language (RVML).

2.2. Visual modeling of fuzzy rule bases

It should be noted that the visual modeling of fuzziness is rather weak represented by the specific tools and notations. In most cases, the researchers use graphics of mathematical functions to represent

linguistic variables and their terms rather than the special or general-purpose notations. Nevertheless, elements for displaying fuzziness were introduced in some extensions of well-known notations. As a result, the fuzzy cognitive maps [16], fuzzy entity-relationship models (ER models) [17], fuzzy UML models [18], etc. were designed. However, these notations are not used for modeling logical or associative rules, and therefore it is proposed to expand RVML in terms of support for linguistic (fuzzy) variables and certainty factors.

2.2. Rule Visual Modeling Language

RVML [10] is a visual language designed for modeling knowledge bases containing logical rules and generating source codes at programming languages implementing this formalism. RVML is based on UML and can be considered as its extension profile using the class diagram terminology, so "class" and "association" concepts are used as basic elements. This language abstracts from the features of specific programming languages and represents logical rules in a generalized form. At the same time, it contains some built-in means for specifying rule priorities and "default" values of slots.

Some RVML features are the following:

- Separate graphics elements for all components of rules without any stereotypes or typed classes as in UML (Figure 1).
- Clear definition of rule actions (add, delete, modify, stop).
- can be considered as an UML extension profile that uses the terminology of class diagrams: the concepts "class" and “association" are the basic elements.
- Abstraction from various knowledge programming languages: logical rules are represented in the generalized form.
- Specific elements that take into account the features of knowledge programming languages: priority (importance) of the rule, "default" values for slots, etc.
- It can be used for synthesizing source codes in CLIPS, DROOLS, etc.

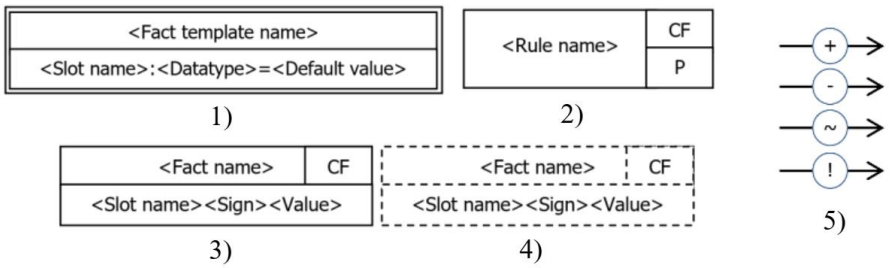


Figure 1. The main RVML elements: 1) a fact template; 2) a rule node; 3) a fact; 4) a condition; 5) connectors of elements with the indication of actions.

RVML is supported by the Knowledge Base Development System (KBDS) [19] and the Personal Knowledge Base Designer (PKBD) [20].

3. Proposed extension of RVML

We extend RVML to support fuzzy knowledge base modeling.

3.1. Basic elements

The main feature of a new extension, namely FuzzyRVML, is the use of linguistic (fuzzy) variables and certainty factors to take into account the fuzziness in reasoning and the uncertainty in reasoning. In this case, the value of a linguistic variable is determined through the so-called fuzzy sets [12]. A fuzzy set is defined through some basic scale (a set of basic values) and a membership function $\mu(x)$. A membership function is a curve that defines how each point in the domain (range) is mapped to a membership value (or degree of membership) between 0 and 1. The domain (range) is sometimes referred to as the universe of discourse. So, a membership function determines the subjective degree of

expert confidence that a particular value of a base scale corresponds to a defined fuzzy set. There are two ways to specify a membership function: *tabular* and *analytical*. The following types of description of a membership function for the analytical method are defined: *triangular*; *trapezoidal*; *S-shaped function*; *Z-shaped function; U-shaped function,* etc.

Visually, this extension is implemented by adding a new data type: *Fuzzy*, and separate graphical elements for membership functions (Figure 2) and terms. The fuzzy elements are displayed with dotted lines, as semi-defined elements.

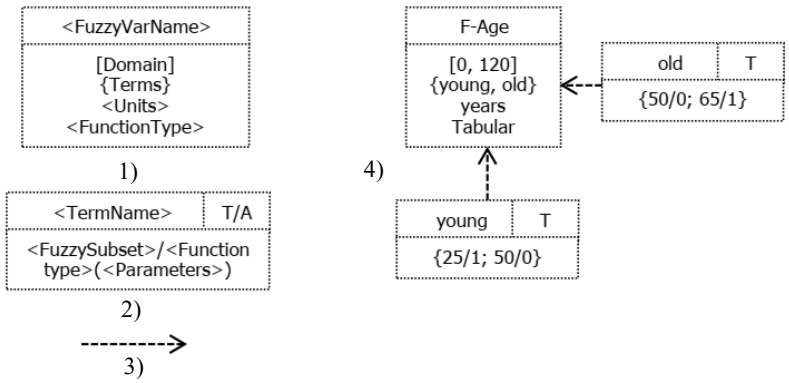


Figure 2. The main FuzzyRVML elements: 1) a linguistic (fuzzy) variable; 2) a term; 3) a connector of a "dependence" type; 4) the representation of the relationship between a fuzzy variable and its terms.

Some FuzzyRVML features are the following:

- It is based on RVML.
- It contains new elements:
  - New data type (*Fuzzy*);
  - Linguistic (fuzzy) variable (*FuzzyVar*) and a set of fuzzy terms (*Terms*) as possible values of a linguistic variable.
  - Certainty factor (*Certainty Factor*).
- It can be used for FuzzyOWL and FuzzyCLIPS source code generation.

FuzzyRVML is integrating with RVML, and their elements can be used together, in particular, figure 3 shows examples of the description of a fact template with a linguistic (fuzzy) variable, as well as the description of a fact with a fuzzy term.

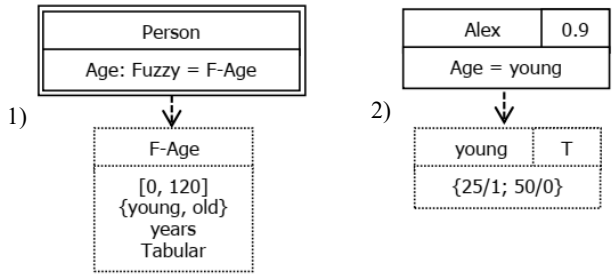


Figure 3. Examples of the integration of RVML and FuzzyRVML elements: 1) a fact template with a linguistic (fuzzy) variable; 2) a fact with a term.

3.2. Software

The support of FuzzyRVML is implemented in the Personal Knowledge Base Designer (PKBD) [20]. It is a tool for prototyping rule-based expert systems and knowledge bases.

PKBD supports RVML, and has a modular architecture (Figure 4) that provides the ability to add modules (dynamic link libraries) that provide generation of source codes and integration with domain

model designers. Currently, CLIPS, Drools, PHP, IBM Rational Rose, StarUML, XMind, CMapTools, and Microsoft Excel support DLLs are included.

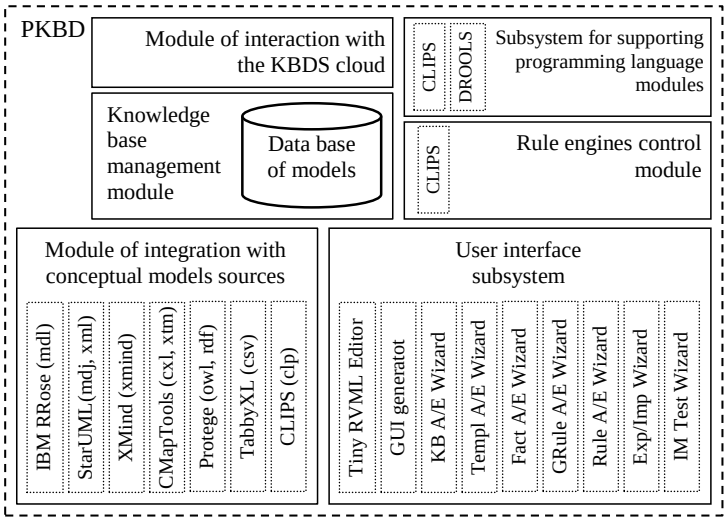


Figure 4. The PKBD architecture [19].

To support FuzzyRVML the following abilities to PKBD were added:

- Descriptions of linguistic (fuzzy) variables. To support this capability some of the PKBD dialogs (wizards) were upgraded, in particular: facts, facts templates, and rules adding/editing wizards (Figure 5). The capabilities to select the *Fuzzy* datatype and to describe a fuzzy variable were added to the facts templates adding/editing wizard. The capability to choose terms of a certain fuzzy variable when describing a slot with a Fuzzy datatype was added to the facts and rules adding/editing wizards. Moreover, a certainty factor can be defined when describing a specific rule or a fact.

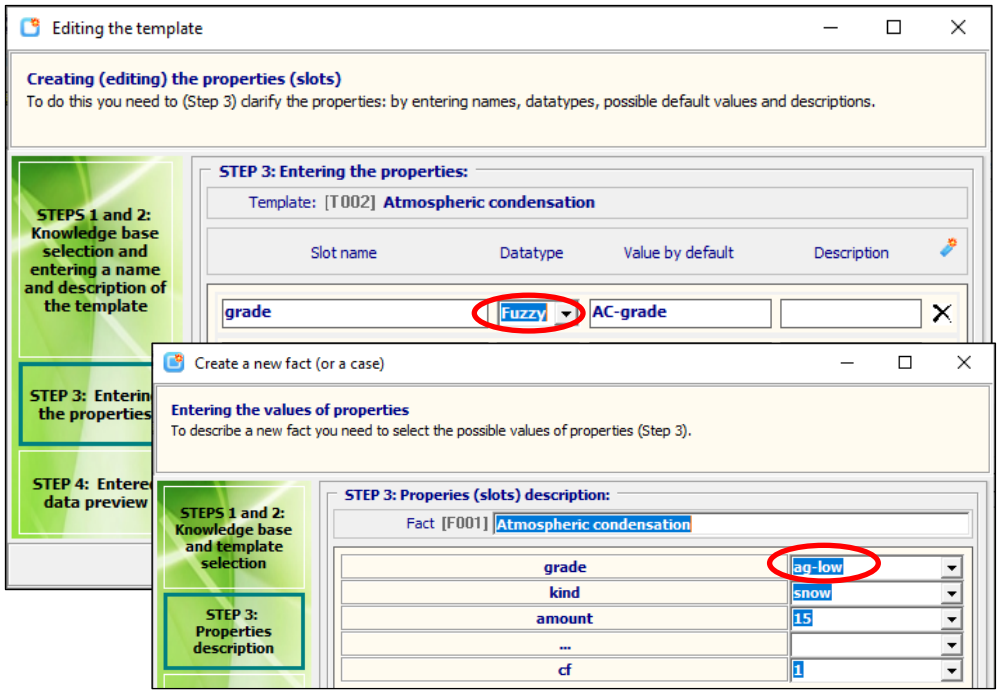
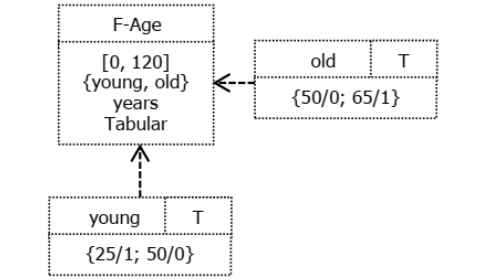


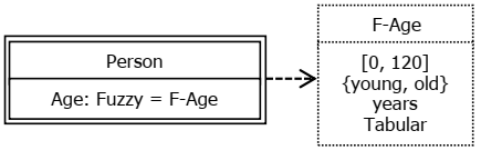
Figure 5. Fragments of the PKBD dialogs upgraded.

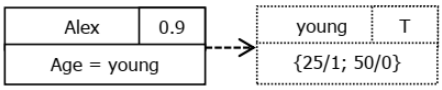
- Representation of fuzzy elements. To support this capability the built-in RVML visualization subsystem (Tiny RVML Editor) was upgraded.
- Generation of source codes on FuzzyCLIPS. To support this capability a new dynamic link library (fzcs.dll) was created with the aid of Object Pascal. The main purpose of the library is unambiguous mapping of FuzzyRVML constructs to source codes; examples of correspondences of elements are presented in table 1. The input of the library is an XML-like description of a knowledge base or its separate elements. This description is processed by the following functions: `GetKnowledgeBaseInfo`, `GetTemplateInfo`, `GetRuleInfo`, `GetFactInfo`, `GetScaleInfo`. The output (result) of the library is the string of FuzzyCLIPS source code corresponding an input data. Besides, for automated recognition and linking this library when starting PKBD, the following functions returning its brief description were added: `DllInfo` and `About`.

Table 1. Examples of correspondences for FuzzyRVML and FuzzyCLIPS elements

Examples of FuzzyRVML elements	Corresponding FuzzyCLIPS elements

	`(deftemplate F-AGE`
	` 0 120`
	` (`
	` (YOUNG (25 1) (50 0))`
	` (OLD (50 0) (65 1))`
	` )`
	`)`

	`(deftemplate Person`
	` (slot age (default "F-AGE"))`
	`)`

	`(Alex`
	` (age "YOUNG")`
	` ) CF 0.9`

Certainty Factor of rules	`(defrule <RuleName>`
	` (declare (CF <CertaintyFactorValue>))`
	…

3.3. Illustrative example

Let's consider an illustrative example. We used fuzzy elements in the development of a knowledge base for assessing the risk of flooding (this task was used in the educational process).

As a result of identification and conceptualization, the main concepts of the domain were defined: Atmospheric condensation, River, Risk, Flood hazard, and Conclusion. A fragment of the obtained domain model in the form of a UML class diagram is shown in figure 6. Next, the UML model was imported to PKBD with the transformation of the main concepts and relationships to fact and rule templates.

Then, we used the linguistic (fuzzy) variables to describe "Atmospheric condensation" and "River" concepts, in particular, "grade" and "water level" properties.

For the "grade" property of the "Atmospheric condensation" concept, we used data on the average amount of precipitation in Irkutsk during the year [21], in particular: the range of possible values [0, 120] mm; possible terms: ag-low, ag-average, ag-high. The values of the terms of the fuzzy variable were set in a tabular way:

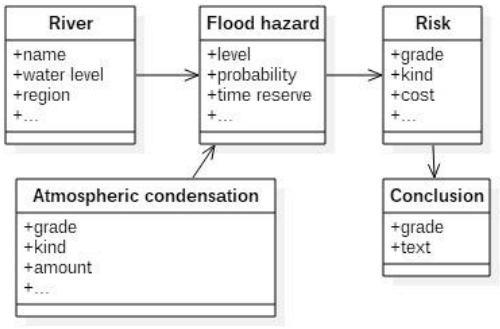


Figure 6. A fragment of the domain model for the flood hazard assessment task in the form of a UML class diagram.

```
ag-low: 0 mm / 1; 20 mm / 0.5; 40 mm and more / 0
ag-average: 0 mm / 0; 20 mm / 0.5; 40 mm / 1; 60 mm / 0.5; 80 mm and more / 0
ag-high: 20 mm / 0; 40 mm / 0.5; 60 mm and more / 1
```

For the "water level" property of the "River" concept data on the level of the Angara river in the area of the first weather station in Irkutsk were used [22]: the range of possible values [0, 204] cm; possible terms: rwl-low, rwl-average, rwl-high. The values of the terms of the fuzzy variable were set in a tabular way:

```
rwl-low: 0 cm / 1; 30 cm / 0.5; 50 cm and more / 0
rwl-average: 0 cm / 0; 30 cm / 0.5; 68 cm / 1; 90 cm and more / 0
rwl-high: 30 cm / 0; 68 cm / 0.3; 90 cm and more / 1
```

Examples of PKBD GUI forms with the fuzzy variable description for the "grade" property of the "Atmospheric condensation" concept and FuzzyRVML constructs and their integration with RVML are shown in figure 7.

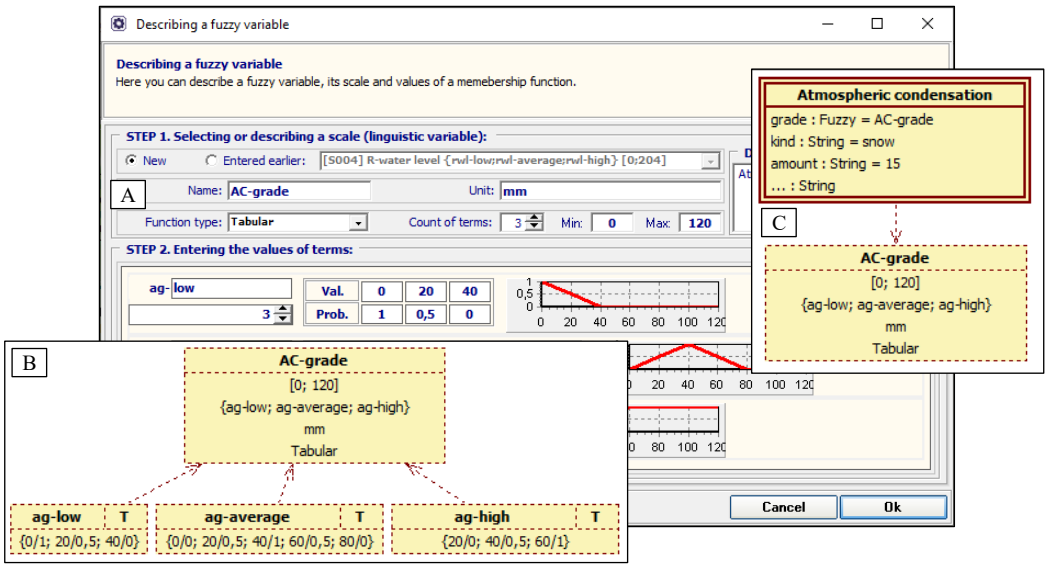


Figure 7. Examples of PKBD GUI forms: (A) a GUI form for describing a linguistic (fuzzy) variable, (B) a FuzzyRVML representation of a linguistic (fuzzy) variable with its terms, (C) an integrated representation of RVML and FuzzyRVML elements when describing a fact template with a linguistic (fuzzy) variable.

The logical rules were also described and the initial facts were set with the aid of PKBD. The following rule templates were obtained as results of the UML class diagram import:

```
IF Atmospheric condensation and River THEN Flood hazard
IF Flood hazard THEN Risk
IF Risk THEN Conclusion
```

An example of a rule template is shown in figure 8 (A). The rule templates reflect the explicit relationships between fact templates and can be used when creating specific rules. For this reason, *Certainty Factors* for rule templates were not defined. In turn, the specific rules define the relationships between facts with concrete slot values, so it's the *Certainty Factors* were defined and depended on the certain facts' values composition. Also, figure 8 shows examples of a fact with a slot value in the form of a fuzzy variable term; a specific rule containing facts with terms of fuzzy variables; and generated source code in FuzzyCLIPS.

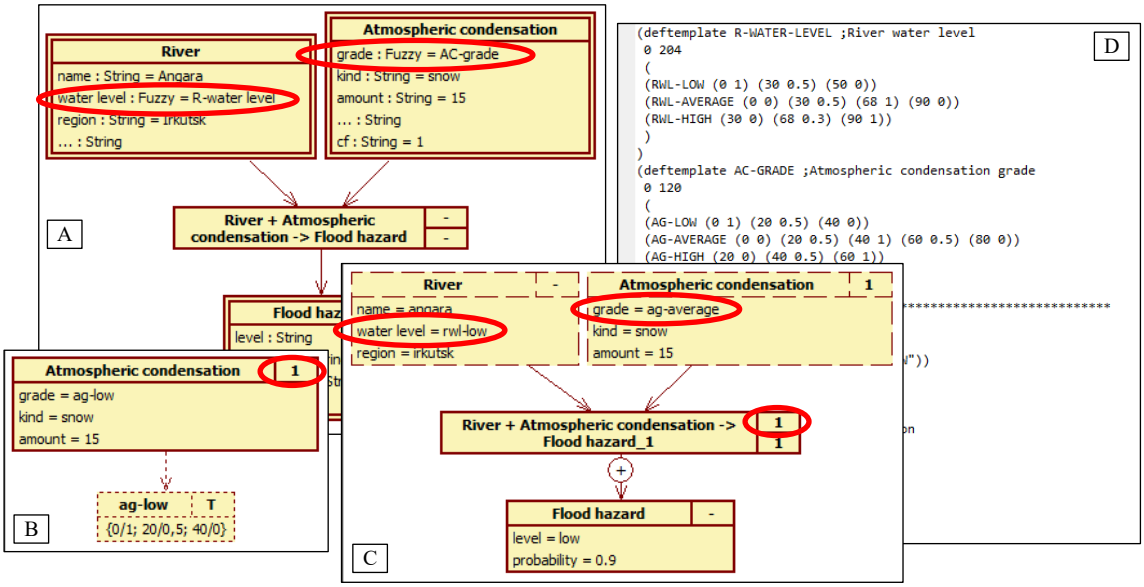


Figure 8. Examples of RVML-schemas with FuzzyRVML elements: (A) a rule template with linguistic (fuzzy) variables; (B) a fact with a slot value in the form of a fuzzy variable term and a certainty factor; (C) a specific rule containing facts with fuzzy variable terms and a certainty factor; (D) generated source code on FuzzyCLIPS.

FuzzyCLIPS code was generated for all FuzzyRVML schemas. Next, we present the source code example generated for specific rule in figure 8 (C):

```
(defrule Atmospheric-condensation+River->Flood-hazard-1 "Description …"
 (declare (CF 1))
 (Atmospheric-condensation ;Atmospheric condensation
 (grade "AG-LOW")
 (kind "SNOW")
 (amount "15"))
(River ;River
 (name "ANGARA")
 (water-level "RWL-LOW")
 (region "IRKUTSK"))
 =>
(assert
(Flood-hazard ;Flood hazard
 (level "LOW")
 (probability "0.9"))))
```

4. Educational empirical research

Evaluation of the effectiveness of the proposed extension is carried out when solving tasks from the educational process of the Institute of data analysis and information technologies of Irkutsk National

Research Technical University (IrNRTU). Students who participated in the empirical research studied «CASE-tools» and «Tools of information technologies» courses, therefore, they are familiar with the basics concepts of software design, UML, knowledge engineering, and expert systems.

4.1. Goals

The time criterion was used to evaluate the proposals, i.e. the main goal of the educational empirical research is to determine the time spent on the development of a fuzzy knowledge base using the proposed RVML extension and software with a comparison with other approaches.

4.2. Research samples

We used the previously developed dataset of educational tasks [23] as research samples. These tasks were revised in the context of using fuzziness (Table 2). The selected tasks are limited to a certain number of domain entities and relationships, which provided to repeat our research until the results were accurately captured.

Table 2. Descriptions of educational tasks used

Task ##	The domain	The number of domain entities	The number of connections	The number of cause-effect relationships	The number of instances of cause-effect relationships	The number of fuzzy variables / terms
1	Car diagnosing	6	5	3	5	1/3
2	Mushrooms identification	5	6	3	5	1/2
3	Computers diagnosing	8	5	3	5	1/2
4	Harvest prognosis	8	7	3	6	3/9
5	Electric kettle diagnosing	9	5	3	5	1/2
6	Public opinion prognosis	5	6	3	7	2/6
7	Iron diagnosing	6	5	3	7	1/2
8	Weather prognosis	7	5	3	6	3/9
9	River flood hazard prognosis	5	4	3	5	2/6
10	Forest fires prognosis	8	7	3	6	3/12

4.3. Research methods

So, in our educational empirical research, we developed knowledge bases with fuzzy rules. We used the following two methods:

- A1: a method based on FuzzyRVML and PKBD. This method implies the use of visual modeling tools or CASE-tools to build a domain model. The main stages of the method are the following: import of the previously developed domain model into PKBD with the automated creation of the knowledge base structures in the form of templates for facts and rules; revising the structure obtained and addition of fuzzy variables, terms, facts and rules with certainty factors; automated code generation for FuzzyCLIPS. It should be noted that this method does not require programming skills.
- A2: a method based on manual manipulation of FuzzyCLIPS constructs (hand-coding method). CLIPSwin is used as the main tool, and the method can also use visual modeling tools or CASE tools, but CLIPSwin is not integrated with them in terms of importing models, so even when using it, the transfer of information about concepts and relationships will be carried out manually. The main stages of the method are the following: transferring elements of the previously developed domain model to the knowledge base structures; adding fuzzy variables, terms, facts, and rules with certainty factors; debugging. This method requires programming skills.

It is necessary to record the time spent when applying these methods on certain tasks with a further comparison.

4.4. Research results

The results of methods for educational tasks are shown in tables 3 and 4.

Table 3. The time spent for A1 and A2 methods

Task ##	A1 time spent (min.)	A2		A2 vs. A1	
		Time spent (min.)	Coding errors (pcs.)	(A2 - A1) (min.)	(A2 - A1)/ A2
1	14.58	40.71	2	26.12	0.65
2	18.16	34.71	0	16.55	0.48
3	20.45	59.02	1	38.57	0.66
4	34.92	59.65	2	24.73	0.42
5	25.61	58.37	3	32.76	0.57
6	37.26	56.51	1	19.25	0.35
7	26.73	45.41	3	18.68	0.42
8	35.58	68.21	0	32.63	0.48
9	26.22	42.39	1	16.17	0.39
10	39.21	46.85	3	7,64	0.17
Avg.	**27.87**	**51.18**	**1.6**	**23.31**	**0.45**

Table 4. The average time of creation of knowledge base elements for A1 and A2 methods

Task ##	The average time of creation of knowledge base elements (min.)						
	A1			A2			
	rule	fuzzy variable	fact	fact template	rule	fuzzy variable	fact
1	1.96	3.54	0,62	2.1	4.11	3.95	1.8
2	2.69	2.93	0.89	2.78	3.05	2.24	1.66
3	2.71	4.98	0.64	2.82	5.68	2.69	1.79
4	2.59	5.23	1.23	1.16	5.72	4.03	1.32
5	3.87	5.04	0.61	2.13	6.05	3.21	2.87
6	3.18	5.7	1.2	1.2	4.73	5.1	2.4
7	2.83	4.5	1.21	1.87	4.2	3.14	1.12
8	2.28	6.17	1.13	2.27	5.18	4.03	3.05
9	3.08	3.79	1.08	2.84	3.64	4.93	1.71
10	2.14	8.44	0.35	1.12	4.28	3.11	0.96
Avg.	**2.73**	**5.03**	**0.89**	**2.02**	**5.12**	**3.24**	**1.86**

4.5. Discussion

The analysis of the results of the educational empirical research showed the superiority of A1 (our proposals) over A2 (manual coding). At the same time, the results obtained did not significantly differ from the previously obtained estimates [23], while still providing a higher performance of the automated developing fuzzy knowledge bases using PKBD and FuzzyRVML compared to completely manual (hand) coding on FuzzyCLIPS.

The following conclusions can be made:

- Detailed analysis (Table 4) showed higher efficiency of manual coding when describing fuzzy variables, which is due to more complex manipulations with the dialog control elements in the case of A1. However, the overall score was not affected due to the small number of fuzzy elements in the examples considered.

- The main contribution to the total time spent in the creation of specific rules, as the most time-consuming process that can be reduced in A1 through the use of substitutions and rule templates.
- The capability to import existed domain models in A1 significantly reduced the time to create the knowledge base because the fact and rule templates were created automatically; in this connection, it was only necessary to specify the data types of the slots and, if necessary, enter fuzzy variables. For this reason, the creation of fact templates was not taken into account when calculated the total time spent in A1.
- Additional time spent in A2 is associated with debugging and searching for coding errors while using more advanced programming tools (with copy, paste, replace, and substitution functions) could reduce this time.
- A1 does not require programming skills, so it is aimed at non-programmers.
- The A1 superiority is achieved by the use of wizards (PKBD GUI dialogs and scenarios), integration with visual designing tools, and automatic code generation.

5. Conclusion

Rules are still the most widespread way to represent expert knowledge despite the popularity of semantic technologies. The effective use of this formalism requires the development of specialized means and software for visualization and generative programming. It is especially true in the case of real-world practical tasks dealing with inaccurate or uncertain information, in particular in the field of reliability and safety of unique technical systems [11].

In this paper, we propose an extension of RVML called FuzzyRVML designed for modeling knowledge with fuzziness and uncertainty. FuzzyRVML supports a fuzzy datatype, concepts of a linguistic (fuzzy) variable, and a certainty factor, and implemented in PKBD software [20].

The evaluation of the proposed extension showed its suitability for describing fuzziness in knowledge bases. FuzzyRVML can be used for the generation of source codes in FuzzyCLIPS, providing rapid prototyping of knowledge bases and expert systems with fuzziness.

A direct comparison with other software systems for modeling fuzziness, in particular, Fuzzy Logic Designer (which is a part of MATLAB) is difficult: on the one hand, Fuzzy Logic Designer has more functionality than the PKBD with FuzzyRVML and supports many ways for describing linguistic variables and visualization of the results of inference, on the other – it is not possible to export the created knowledge base and generate source codes for integration into other applications, for example, FuzzyCLIPS, while FuzzyRVML and PKBD are designed specifically for this task.

Currently, only a tabular way for describing the values of terms of linguistic variables is implemented in PKBD, which is due to the focus on generating source codes for FuzzyCLIPS. In the future, it is planned to add the support for the extension proposed to the Knowledge Base Development System [19] and to describe the evaluation of FuzzyRVML in the case of diagnostics of unique technical systems.

6. Acknowledgments

The present study was supported by the Russian Foundation for Basic Research (Grant no. 19-07-00927). Some results were obtained within the framework of the State Assignment of the Ministry of Education and Science of the Russian Federation for the project "Methods and technologies of a cloud-based service-oriented platform for collecting, storing and processing large volumes of multi-format interdisciplinary data and knowledge based upon the use of artificial intelligence, model-guided approach and machine learning".

References

[1] Wagner W P 2017 Trends in expert system development: A longitudinal content analysis of over thirty years of expert system case studies *Expert Systems with Applications* **76** 85–96

[2] CLIPS: A Tool for Building Expert Systems, http://clipsrules.sourceforge.net/

[3] Friedman-Hill E 2003 Jess in Action: Rule-based Systems in Java, Manning
[4] SWRL: A Semantic Web Rule Language Combining OWL and RuleML, https://www.w3.org/Submission/SWRL/
[5] Drools, https://www.drools.org/
[6] RIF Overview (Second Edition), https://www.w3.org/TR/rif-overview/
[7] Koznov D V 2008 *Basics of visual modeling* (Moscow: BINOM) p 246
[8] Booch G, Maksimchuk R A, Engle M W, Young B J, Conallen J and Houston K A 2007 *Object-Oriented Analysis and Design with Applications* (New York: Addison-Wesley Professional) p 717
[9] Lukichev S, Giurca A, Wagner G, Gasevic D and Ribaric M *2007 Using UML-based rules for web services modeling* Proceedings IEEE 23rd International Conference on Data Engineering Workshop 290–297
[10] Yurin A Yu, Dorodnykh N O, Nikolaychuk O A and Grishenko M A 2018 Designing rule-based expert systems with the aid of the model-driven development approach *Expert Systems* **35(5)** 12291
[11] Berman A F, Nikolaychuk O A, Yurin A Yu and Pavlov A I 2014 A methodology for the investigation of the reliability and safety of unique technical systems *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* **228** 29-38
[12] Dubois D, Prade H and Yager R R 2014 *Readings in Fuzzy Sets for Intelligent Systems* (Amsterdam: Elsevier) p 928
[13] Orchard R A 1995 FuzzyCLIPS Version 6.04A – User's Guide. Institute for Information Technology, National Research Council Canada
[14] Gavrilova T A and Gulyakina N A 2011 Visual knowledge processing techniques: A brief review *Scientific and Technical Information Processing* **38(6)** 403-408
[15] Berman A F, Dorodnykh N O, Nikolaychuk O A and Yurin A Yu 2019 Event trees transformation for rule bases engineering *Proceedings of the 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)* 1138-1143
[16] Kosko B 1986 Fuzzy Cognitive Maps *International Journal of Man-Machine Studies* **24** 65–75
[17] Zhang F, Ma Z M and Yan L 2008 Representation and reasoning of fuzzy ER model with description *logic Proceedings of IEEE International Conference on Fuzzy Systems* 1358–1365
[18] Sicilia M A, Garcia E and Gutierrez J A 2002 Integrating fuzziness in object oriented modeling language: towards a fuzzy-UML *Proceedings of International Conference on Fuzzy Sets Theory and its Applications* 66–67
[19] Dorodnykh N O 2017 Web-based software for automating development of knowledge bases on the basis of transformation of conceptual models *Open Semantic Technologies for Intelligent Systems* **7** 145–150
[20] Yurin A Yu, Dorodnykh N O 2020 Personal knowledge base designer: Software for expert systems prototyping *SoftwareX* 11, 100411
[21] The climate of Irkutsk, https://ru.wikipedia.org/wiki/Климат_Иркутска
[22] Allrivers: water level on-line, https://allrivers.info/gauge/angara-irkutsk
[23] Yurin A Yu, Dorodnykh N O and Nikolaychuk O A 2021 The rapid development of knowledge bases using UML class diagrams *International Journal of Computer Aided Engineering and Technology* **14(1)** 39-61