Int. J. Appl. Math. Comput. Sci., 2019, Vol. 29, No. 3, 489–501
DOI: 10.2478/amcs-2019-0036

A THREE–LEVEL AGGREGATION MODEL FOR EVALUATING SOFTWARE USABILITY BY FUZZY LOGIC

EVA RAKOVSKÁ [a,*], MIROSLAV HUDEC [a]

[a]Faculty of Economic Informatics
University of Economics in Bratislava, Dolnozemská cesta 1, 852 35 Bratislava, Slovakia
e-mail:{eva.rakovska,miroslav.hudec}@euba.sk

Rapid deployment of IT brings about new issues with software usability measurement. Usability is based on users' experience and is strongly subjective, having a qualitative character. The users' comfort is usually collected by surveys in their daily work. The present article stems from an experimental study related to the evaluation of the usability of tools by a rule-based system. The work suggests a robust computational model that will be able to avoid the main problems arising from the experimental study (a large and less-legible rule base) and to deal with the vagueness of IT user experience, different levels of skills and various numbers of filled questionnaires in different departments. The computational model is based on three hierarchical levels of aggregation supported by fuzzy logic. Choices for the most suitable aggregation functions in each level are advocated and illustrated with examples. The number of questions and granularity of answers in this approach can be adjusted to each user group, which could reduce the response burden and errors. Finally, the paper briefly describes further possibilities of the suggested approach.

Keywords: measuring software usability, rule-based system, fuzzy quantifiers, aggregation functions, questionnaire.

1. Introduction

In today's competitive world, information technologies (ITs) have a significant impact on the management and efficiency of enterprises as well as governmental and public agencies. These institutions make an effort to optimize their processes and often put a lot of investments into IT. The traditional perception of IT management (hardware support, network services, installation services, etc.) is rapidly changing. Information technologies are offered as services that are integrated into the companies to support the achievement of business goals. IT services are directly involved in the company as part of the business process.

These institutions frequently try to introduce the newest IT, although it is not always the right way to satisfy employees and to achieve the desired process efficiency. Therefore, the implementation of the newest IT may not bring the expected results in the business processes' effectivity. Modern IT management is governed by frameworks and standards for implementation and management, like ISO standards (ISO, 2018; 2011), the Information Technology Infrastructure Library (ITIL) (Greiner and White, 2019) as well as Control Objectives for Information and Related Technologies (COBIT) (ISACA, 2018) at different levels of management. Frameworks derived from software engineering are usually used for software development and its further management (maintenance, revision, re-engineering, etc.). Many software engineering tools assess the quality and reliability during the design and development phase. There are plenty of tools for software quality measurement; however, it is not easy to control and measure the actual performance of IT services and software in business processes in practice. Software testing is a phase of software engineering methodologies, but it is done by testing specialists, not regular daily users. It is not easy to use all the frameworks and standards; therefore some enterprises and public institutions prefer using the balance scorecard methodology for monitoring business performances, and using traditional non-financial and financial metrics for monitoring IT performances (Pavlík, 2018).

Although all methodologies specify various key indicators of software performance for achieving the

*Corresponding author

business goal and using metrics for software validation, they arise from the needs of software engineers and do not take into account the user's perspective. To validate software quality from the users' point of view means to assess the software as a product. Software users have no idea about the set of metrics used by software development. Their opinion includes such items as users' satisfaction and experience, or customer sentiment, which are not measurable quantitatively. Hence, the task is to design soft metrics and create an appropriate model for assessing software quality as a product.

As mentioned before, the soft metrics characterize items such as IT user satisfaction or IT user experience within different user groups. Satisfaction and experience come from subjective opinions of users (Albert and Tullis, 2013), such as whether the software application response time is adequate for them, whether the software availability is appropriate, whether the software has intuitive interface, whether the software saves the user's time, whether the user is able to use all functionalities intuitively, etc. User experience is usually monitored by surveys. We suppose that observability and measurability are attributes of user experience (Albert and Tullis, 2013).

The aim of the present paper arises from the experimental idea to design a rule-based expert system for monitoring the software usability in daily use. Therefore, this means monitoring which software or software functionality (or part thereof) is the most valuable in the company and in which department. Many companies use a mix of software and applications (especially small and medium-sized companies); those are not compatible and sometimes are useless. We examined the research and experimental study by Králiková (2017) as well as Rakovská and Hudec (2020), where a survey was given to users for evaluating software usability and gaining adequate knowledge for the preparation of fuzzy rules. The research used the survey data to design the rules in the rule-based expert system based on the Mamdani inference produced by the MATLAB Fuzzy Inference tool. The experimental study detected problems with the management of a high number of rules, by preparing appropriate understandable questions, which can be expressed by linguistic values, with an adequate flexible number of rules and computation by the Mamdani inference. The survey was realized by questionnaires and fuzzy-rule preparation, which revealed several problems. These observations led to the suggestion of a new solution. Thus, the research in the experimental study was finally focused on analyzing the benefits and drawbacks of expressing software usability by the questionnaires and shown that the rule base grows significantly, even when we reduce the number of rules with the disjunctive normal form (Zimmermann, 2001).

The next research started with the analysis of possibilities of handling uncertain and incomplete data, and aggregating them in an appropriate way. In order to mitigate incomplete data, questionnaires should be adjusted to skills of users in diverse departments. As a consequence, we expect a different numbers of questions and various granularity of possible answers. The research question in this environment is how to efficiently measure the users' satisfaction with employed software tools not only in departments, but also among departments inside an institution, and therefore rank these tools accordingly. In this direction, we tried to solve the problem of aggregating answers from various imbalanced groups (departments) together and rank the evaluated software accordingly. The preliminary results of this research are presented by Rakovská and Hudec (2020).

The paper is organized as follows. Section 2 provides motivations for this work based on the experiential study with a fuzzy rule-based system. Section 3 is dedicated to three-level aggregation with the support of fuzzy logic and an illustrative example, whereas Section 4 discusses the achieved results. Finally, Section 5 concludes the paper and sets out future research opportunities and applicability.

2. Motivation and background

As mentioned in Section 1, the research arises from practice when users are not satisfied with IT services and software in their companies. This situation is more frequent in the public sector, at schools and universities or in the health area, but also in small and medium-sized enterprises. The management often does not focus on the opinions of employees as to which software is useful for their work or whether the software is comfortable for them. The user perspective of software quality reveals the connection between the user and the product and is observable and measurable. Seffah *et al*. (2001) state: "A good quality in use model should define all the characteristics that are required for a product to meet predefined usability goals in a specified context of use." They mentioned, for example, characteristics such as efficiency, learnability, human satisfaction and safety, which are observable, but are not well quantified. The user perspective of software usability is closely linked to software efficiency and effectivity, which are involved in the user perspective.

As Albert and Tullis (2013) hold, efficiency is "the amount of effort required to complete the task" and effectiveness means "being able to complete the task". Software efficiency and effectiveness are usually connected with software development and software engineering, with an impact on the human–computer interaction area. ISO standards include the definition of usability from the human–computer interaction (see ISO 9241-11:2018 (ISO, 2018)): "Usability relates to the outcome of interacting with a system, product or service.

Usability, as defined in this document, is not an attribute of a product, although appropriate product attributes can contribute to the product being usable in a particular context of use." Other characterizations of software quality and usability are from the software engineering point of view (see ISO/IEC 25010:2011 (ISO, 2011)) named "Systems and software engineering—Systems and software Quality Requirements and Evaluation (SQuaRE)—System and software quality models."

Software engineering methodologies offer metrics for better software performance. A basic metric classification comes from hard and soft metrics used to measure the efficiency of performance in the company and coming from internal attributes of the software. The hard metrics can be easily expressed by financial indicators and do not need additional costs. Despite hard metrics, in our work we consider only soft metrics, which often express the degree of internal goal achievement or the degree of process improvement through software use. Soft metrics represent such concepts as effectiveness, efficiency, availability, safety, user comfort (understandable user interface) and accessibility on internationality as a set of factors or a set of criteria (completeness, minimum actions to achieve the goal of the task, minimum load memory, etc.) mentioned by Seffah *et al*. (2001). We also identified other concepts of software usability measurement, such as the clarity of concepts and logical sequence of operations, transparency of design, quick availability of certain items, expression of frustration of the users, misinterpretation of some parts and information in the software, etc. All these factors and criteria are thought of as purely qualitative and observable, so it is easy to collect data concerning these issues by using questionnaires. Surveys conducted by questionnaires, although effective, usually cope with the issues of the item and unit non-response and measurement error, which are far from negligible (Bavdaž *et al*., 2011; Bavdaž, 2010).

We observed the same behaviour (to a lower extent) in the case of a survey distributed among users in the company. In this case, it causes an unbalanced number of filled questionnaires (e.g., 20 answered in one department and 12 in another) and non-response items (users have not responded to all questions). Further, respondents were not always careful in filling out the questionnaires, i.e., they filled in neighbouring values in categorical answers. The next section is dedicated to the evaluation of data collected by questionnaires, concerning user satisfaction in a company running its business in the IT sector. This task was solved by the fuzzy rule-based system (Králiková, 2017; Rakovská and Hudec, 2020), because IF-THEN rules related to degrees of software usability can be easily created and linguistically explained. The next subsection explains the experiments carried out and the obtained results.

2.1. Experimental study as a background. The experimental study was undertaken in a mid-sized IT company. It was based on a survey which maps users' perspectives and experience with information systems and software within the company. As mentioned before, we applied a fuzzy rule-based approach for evaluating the survey. We expected that the fuzzy rule-based system might be a suitable method for evaluating the survey with the use of the Likert scale for answers (Likert, 1932). There was also a possibility of managing and controlling the implication by IF-THEN rules. Nevertheless, the fuzzy rule approach has the following main problems:

- how to create questionnaires on a common platform for each piece of software in each department in the enterprise;
- how to take into account different levels of software users;
- how to aggregate the answers within the departments in an appropriate way to get a useful result;
- how to evaluate the high number of conditions in the rules so that the result reliability is not lost;
- how to deal with large numbers of rules.

The evaluation of software usability was based on user satisfaction surveys using the approach suggested by Allen and Seaman (2007). The questionnaire was inspired by certain methods and their combinations (System Usability Scale, Software Usability Measurement Inventory, etc.)

First, we produced three different types of questionnaires for three groups (management, development, group of other users), each questionnaire containing 27 questions. We used the Likert scale (Likert, 1932) to acquire answers; the scale was mostly from 1 to 5 and from 1 to 10 in a few questions when we needed more precise response granularity. Each questionnaire started with the same question: "Type the software you most frequently work with. This piece of software will be further the subject of the questionnaire, and all questions will be asked only about it" (Králiková, 2017). The other 26 questions mapped user satisfaction with this software from various points of view. We asked for total software satisfaction; whether the software product is up to date; how users work with the software product; whether they use service and maintenance services; whether the software is intuitive and consistent; or whether they will prolong the license. Some of the other questions about the interaction between the user and the software; whether the software sometimes shut down unexpectedly; whether it is a satisfactory software language; whether it has all the necessary features; whether the user feels frustrated when

working with the software or if the software environment is easy to use; how often he or she works using the software; or whether the software is slower after hours of work, etc.

Table 1 shows some results from questionnaires at the economic department of the company, where the users preferred and frequently used software *S* and some parts of it (e.g., *S1* and *S2*). Then the users answered the questions concerning the chosen software (Q1–Q26). For the majority of questions the answers were on the scale from 1 to 5 (1 = disagree, 2 = rather disagree, 3 = I cannot judge, 4 = rather agree and 5 = agree), whereas some questions using a more detailed scale (1 to 10) for evaluating the attributes such as interaction with the software. Every row in Table 1 represents 26 collected answers from one user (Q0 is choosing the software). Several rows contain also the missing values (e.g., the first and the last), where the respondents did not answer (for instance, the first question in the first row).

We considered 26 inputs for the premise of each rule (for instance, from *A11* to *A126* for the first rule). The supposed rule-system within one department was suggested as follows:

- IF $A11$ AND $A12$ AND $A13 \ldots$ AND $A1j$ THEN $B1$,
- IF $A21$ AND $A22$ AND $A23 \ldots$ AND $A2j$ THEN $B2$,
- $\ldots$,
- IF $Ai1$ AND $Ai2$ AND $Ai3 \ldots$ AND Aij THEN Bi,

where j is the number of answers (in our case, 26) in the questionnaire and i represents the number of all combinations that are given from all possible answer values (based on the scale from 1 to 5). Then we set the linguistic values (see above) into rules. Secondly, we processed the rules with the Mamdani fuzzy inference, where AND connective = min, OR connective = max, Implication = min (Mamdani technical implication (Gupta and Qi, 1991)), Aggregation = max, Defuzzification = centroid using the MATLAB tools. In our case, considering one group, one questionnaire with 27 questions, we computed the number of possible answer combinations. The result was to have more than 11 billion rules in the knowledge base (Rakovská and Hudec, 2020). In order to cover all the options, we would have to create as many rules as possible. After multiplying three types of questionnaires, the number would increase even more.

Therefore, keeping the balance between precise capturing of all the details in the knowledge base model and acceptable computational complexity was a big challenge for further research activities. It seemed that a satisfactory solution was to aggregate the rules in an appropriate way. Further, we divided the constructed fuzzy rules into 5 groups in each questionnaire and then gradually aggregated them using again the Mamdani fuzzy inference. Although the number of rules decreased, it was still unsatisfactory and the response time of the expert system would not be relevant. In that case, we processed only the answers from the scale from 1 to 5. If we took into account a more precise granularity of the answer, the number of rules would be even higher.

The other option is full fuzzification of input and output variables. The first step in constructing such a fuzzy-rule based system is fuzzification of input variables. The fuzzification of the output variable is required for the Mamdani inference system (Zimmermann, 2001). In this way, we avoid using singletons in the above created rules. For the [1, 10] scale of the respondents' answers) (Table 1), inputs are fuzzified into three fuzzy sets: *low*, *medium* and *high* as

$$\begin{aligned} \text{Low} &= \{(1,1),(2,),(3,0.75).(4,0.25)\},\\ \text{Medium} &= \{(3,0.25),(4,0.75),(5,1),(6,1),\\ &\quad (7,0.75),(8,0.25)\},\\ \text{High} &= \{(7,0.25),(8,0.75),(9,1),(10,1)\}. \end{aligned}$$

The output attribute *usability* is fuzzified into five elements: *very low*, *low*, *medium*, *high* and *very high*. Hence, the number of rules is $26^3 = 17576$. Several of them are as follows:

- IF $A1$ is LOW AND $A2$ is LOW AND $\ldots$ AND $A26$ is LOW THEN B is VERY LOW,
- IF $A1$ is LOW AND $A2$ is LOW AND $\ldots$ AND $A26$ is MEDIUM THEN B is VERY LOW,
- $\ldots$,
- IF $A1$ is HIGH AND $A2$ is HIGH AND $\ldots$ AND $A26$ is HIGH THEN B is VERY HIGH,

Although the number of rules is significantly reduced, it is still too high for business or domain expert users. Moreover, fuzzification into three fuzzy sets is not beneficial for a scale consisting of 5 or fewer elements. However, in this way it does not cope effectively with the different features of departments (the number of workers and their respective skills).

Even the disjunctive normal form is not a solution because there are few rules with common input parts. Finally, we decided for restriction of the number of rules by using qualitative heuristics. We reduced the number of rules to 1200 using the FIS (fuzzy inference system) matrix in MATLAB, but the number was still significant. We did not use weighted rules, because we supposed all the questions were of the same significance.

Table 1. Illustrative sample of answers from users' questionnaires in the experimental study.

soft./question	Q1 Q14	Q2 Q15	Q3 Q16	Q4 Q17	Q5 Q18	Q6 Q19	Q7 Q20	Q8 Q21	Q9 Q22	Q10 Q23	Q11 Q24	Q12 Q25	Q13 Q26
S	[illegible] 2	4 4	2 2	3 4	4 5	4 4	4 2	5 8	4 4	4 4	4 4	8 1	2 8
S2	4 2	4 4	4 4	4 4	4 5	4 4	4 1	5 7	4 4	4 4	4 4	7 1	4 8
S2	2 2	4 4	4 2	5 4	4 4	2 4	3 2	5 8	2 4	4 4	2 5	7 1	2 8
S2	4 2	4 2	2 2	5 3	4 4	2 3	2 2	4 6	2 4	3 4	3 2	5 2	2 6
S	2 1	4 2	4 2	5 4	5 5	4 4	2 1	4 8	2 3	4 2	5 2	7 1	2 8
S	4 4	4 3	3 4	4 4	4 4	4 4	[illegible] 2	5 8	4 4	4 4	4 [illegible]	8 4	4 7
S1	4 4	4 4	3 3	4 4	3 4	4 4	4 4	4 7	4 4	[illegible] 4	4 4	7 4	4 7
S1	4 4	4 5	4 4	5 4	4 4	3 4	4 [illegible]	5 9	[illegible] 4	5 4	4 4	8 4	[illegible] 8
S	4 4	4 4	3 4	4 5	4 4	4 5	3 4	4 8	4 4	4 4	4 4	7 4	4 7
S	4 4	5 4	4 4	4 4	3 4	4 4	4 5	4 8	4 4	4 4	4 4	8 4	[illegible] 8
S	4 4	3 4	4 4	4 4	2 4	3 4	4 4	3 8	4 4	4 4	3 4	7 4	4 8
S	4 4	4 4	3 5	4 4	[illegible] [illegible]	4 3	4 4	4 8	4 5	4 4	4 4	7 4	4 8
S	4 4	4 4	5 4	4 4	4 4	4 4	4 5	4 [illegible]	5 4	4 4	4 4	8 5	5 8

2.2. Experiments conclusion and discussion. After the experiment, we can summarize its pros and cons. To sum up, the input was the following:

- only three groups of users;
- the same number of questions in each questionnaire (although some questions were different depending on the user group);
- only two types of questions: the first type has the answers from the [1–5] scale, whereas the second type from the [1–10] scale, so the granularity of answers was different;
- number of respondents was lower than fifty.

The experiment shows us that, although we expected a good result using the well-known fuzzy rule-based method and MATLAB software for evaluating questionnaires, some problems as well as high computing complexity were recognized. The main drawback was that computing complexity was very high and therefore we had a difficulty with the size of the model and result reliability. That was the reason for starting considering other possibilities of evaluating the usability by applying different aggregation strategies.

These experiments showed benefits and weak points of fuzzy rule-based evaluation. It is not an easy task to create a less-complex fuzzy rule base, and therefore avoid computational intensive activities (Rakovská and Hudec, 2020). Another problem is efficiently spreading questionnaires for the same tools among departments, e.g., tools used in several departments like the text processor, spreadsheet calculations, managerial tools, etc. In this case, we cope with the problems of different levels of skills and unequal numbers of filled questionnaires (different numbers of workers in departments). There were missing values recorded in this survey, although it was carried out within a company. In the experiment, responses containing missing values were not processed further. Usual statistical ways for estimating missing values are not applicable due to a low number of respondents per department.

The most suitable solution is a motivation to cooperate in the survey, e.g., by anonymized questionnaires. In order to keep the response burden as low as possible, we should adjust the

design to different respondent groups (Calinescu and Schouten, 2012; Snijkers *et al*., 2013). This means that we should offer a set of three possible categorical answers: negative, neutral, positive for the least skilled workers in terms of IT and general literacy (less demanding work positions), whereas for the expert users (high level of IT skills or deep domain knowledge) we may offer finer granularity like very negative, negative, more neutral than negative, neutral, more neutral than positive, positive, very positive. The next section explains the developed robust approach to cope with the recognized problems. In addition, the categorical answers might be expressed by linguistic terms instead of numbers, as indicated in Table 6.

That was the reason for finding other possibilities of aggregating the data collected from questionnaires in order to keep the computational complexity of a reasonable size and minimize effect of the measurement error (e.g., respondents marked the neighbouring answer). A promising direction is quantified aggregation (Yager, 1982) adjusted to questionnaires and various aggregation functions (more about these functions is given, e.g., by Beliakov *et al*. (2007) and Grabisch *et al*. (2009)).

3. Hierarchical aggregation

According to the observations in Section 2, the main features of the problem considered are as follows:

- evaluating usability of the common software among diverse departments;
- diverse groups of respondents with different levels of expertise and experience;
- different number of filled questionnaires among groups (i.e., unequal size of groups or not all respondents cooperative in the survey);
- some respondents may not fill in the questionnaire carefully, i.e., they can fill in the neighbouring value from the set of categorical answers.

Therefore, the aim is to develop a flexible survey system, which can efficiently solve these problems.

3.1. Preliminaries of fuzzy sets. Flexible evaluation relies on the theories of fuzzy sets and fuzzy logic, where belonging to a set is a matter of degree. A fuzzy set F over the universe of discourse X is defined by the membership function μ_F that matches each element of X with its degree of membership to the set F (Zadeh, 1965),

$$\mu_F(x) : X \to [0, 1], \tag{1}$$

where $\mu_F(x) = 0$ means that an element x does not belong to F, while $\mu_F(x) = 1$ means that x is a full member of F. A value $\mu_F(x) \in]0, 1[$ indicates the intensity with which the element x belongs to F.

An example is the set *high opinion*, where the maximal rating score means clearly belonging to this set, whereas significant rating means belonging with a slightly lover degree. This set is plotted in Fig. 2. When the universal set X contains few elements (e.g., scale of possible answers), we directly assign the matching degree to each element, i.e., $FC = \{(\text{very low}, 1), (\text{low}, 0.75), (\text{medium}, 0.25)\}$.

People are familiar with linguistic aggregation by elastic quantifiers, e.g., *most of* and *about half*. These so-called fuzzy relative quantifiers are formalized by fuzzy sets. The proportion of records in a set X that fully and partially belongs to the fuzzy set F is defined as

$$y = \frac{1}{n} \sum_{i=1}^{n} \mu_F(x_i), \tag{2}$$

where x_i is the i-th record in a data set, or in our case the answer of the i-th question in a questionnaire.

Thus, the validity (truth value) of aggregation with the quantifier *most of* is

$$\mu_Q(y) = \begin{cases} 1 & \text{for} \quad y \geq 0.85, \\ \dfrac{y - 0.5}{0.35} & \text{for} \quad 0.5 < y < 0.85, \\ 0 & \text{for} \quad y \leq 0.5, \end{cases} \tag{3}$$

where y is the proportion of records which belong to the fuzzy set (1). The validity assumes values from the unit interval, i.e., when all collected opinions are clearly positive, the validity is 1. If the proportion of positive opinions is decreasing, it causes the decrease of the truth value for the quantifier *most of*. When the proportion is low, the truth value is 0.

Main benefits of such evaluation are as follows: (i) similar observations are similarly treated and (ii) understandable interpretation for domain experts, without a considerable level of mathematical literacy. Fuzzy sets are proposed for aggregation on the second and third levels. The other relevant concepts and functions are explained in the successive subsections.

3.2. Questionnaire organization. Generally, we consider S_k, $k = 1, \ldots, K$, software tools for calculating their usability; G_l, $l = 1, \ldots, L$ user groups (or respondent groups to be in the line with the mainstream literature terminology in surveys), e.g., G_1 is a group of managers, G_2 is a group of IT developers, G_3 is a group from the accounting department, etc.). The number of groups depends on the type of organization (enterprise). Further, we have respondents R_{ilk}, $i = 1, \ldots, m_{lk}$, where m_{lk} is the number of respondents in group l that evaluate software k and x_{ijlk}, $j = 1, \ldots, n_{lk}$, is an

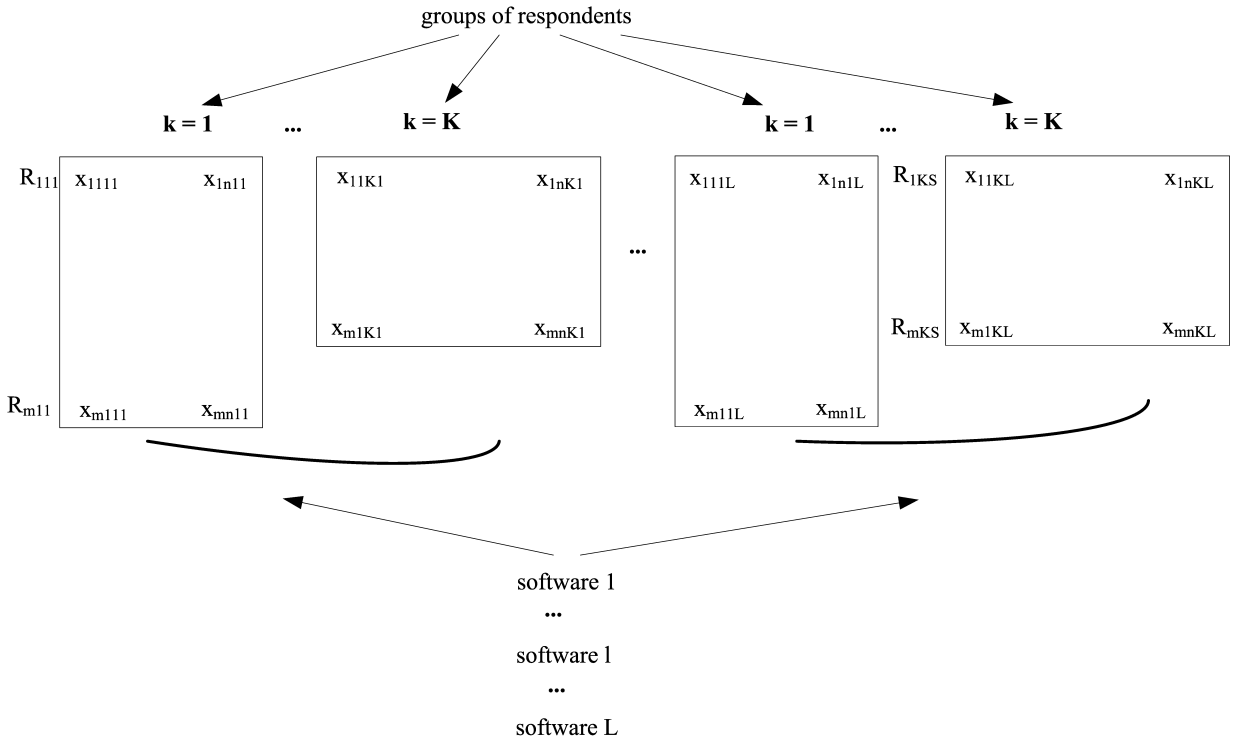

Fig. 1. Hierarchical structure of questionnaires.

answer of the i-th respondent to the j-th question while n_{lk} is the number of questions for respondents in group l for software k. Figure 1 illustrates this structure for software S_1. For instance x_{m111} stands for the answer of respondent m to the first question in the first user group regarding the first evaluated software and x_{mnKL} is the n-th answer recorded for the m-th respondent in group L for software tool K.

Each user group may have a different number of categorical questions and a number of possible answers. In this way, we adjust questionnaires to particularities of user groups. For instance, for group G_1, the number of possible answers is from a scale consisting of five elements, whereas for G_2 the number of possible answers is 10, i.e., the possible answers are numbers from the scale [1, 10] of integers, where 1 is the worst and 5 or 10 the best opinion, respectively. Answers might be also expressed with linguistic terms, where one number corresponds to one linguist term. In this way, questionnaire design is adjusted to the expected respondents skills among departments which may influence the reduction in non-responses and errors (Calinescu and Schouten, 2012; Snijkers *et al.*, 2013).

Comparing answers to the same question providing a different number of categorical answers is not a problem, because the transformation among categorical sets suggested by Herrera and Martínez (2001) carries out the necessary conversions by linguistic pairs. With this transformation, we are able to transform all answers from term sets of various granulations into the basic linguistic term set for the required analyses.

The problem of a different number of answers per respondent, i.e., they may answer only several questions (i.e., those for which they can easily provide an opinion) is compensated by a large number of respondents (Morente-Molinera *et al.*, 2018). However, the nature of our problem is not the same. It is true that we have a different number of questions per group and the granularity of possible answers differs, but our task has relatively small and compact groups, and all of them should provide an answer to each question.

3.3. Aggregation. Aggregation operators reduce a set of values into a unique representation or a meaningful number (Beliakov *et al.*, 2007). In this direction, we searched for suitable aggregation for the problem plotted in Fig. 1, where we see different sizes of questionnaires for diverse respondent groups. The standard classification of aggregation functions divides them into conjunctive, averaging, disjunctive and hybrid (Calvo *et al.*, 2002; Dubois and Prade, 2004). In order to cover the requirements and create a robust and flexible system for evaluating usability, we organized aggregation into the following three levels: the respondent level, the department level (groups of respondents) and the software level.

3.4. Aggregation at the respondent level. At the lowest level, there are questionnaires for particular groups of respondents, where x_{ijlk}, $j = 1, \ldots, n_{lk}$, is the answer of the i-th respondent to the j-th question and n_{lk} is the number of questions for respondents in group l for software k. In our study, aggregation at the first level should consider all answer for each respondent. Hence, conjunctive, disjunctive and hybrid aggregation functions are not suitable. The domain for these functions is the unit interval. Although the answers do not belong to the unit interval, they might be straightforwardly converted to this interval. This is the reason why we considered all classes of aggregation functions. A suitable one is the class of averaging functions. In this case, we express this aggregation as

$$A^1_{ilk}(\mathbf{x}) = f_{av}(\mathbf{x}), \tag{4}$$

where index ilk is the i-th respondent in group l for software k and f_{av} is an averaging aggregation function. Generally, any averaging function might be used. However, we applied the arithmetic mean due to full neutrality (i.e., low values are fully compensated by high ones). It does not hold for the other means (for instance, geometric or quadratic means), which are closer to the minimal or maximal observations (Dujmović, 2007).

The score for each questionnaire can be also calculated by the sum of respondents' answers,

$$A^1_{ilk}(\mathbf{x}) = \sum_{j=1}^{N_{ilk}} x_{ijlk}, \tag{5}$$

where j_{ilk} is the j-th answer for the same group and software, and x is the numerical value of the answer.

The solution for (5) assigns a value from the set of natural numbers, whereas the solution for (4) assigns a real number from the interval between the smallest possible and the highest possible answer. These functions were chosen due to their computational efficiency, although the solution is not in the unit interval.

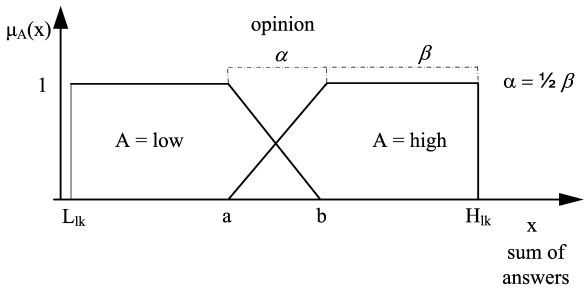

Fig. 2. Linguistic variable *opinion* and its two labels expressed by fuzzy sets, where $a = 13.95$ and $b = 17.05$ for the sum of answers, and $a = 2.79$ and $b = 3.41$ for the average of answers.

Table 2. Illustrative group G_1, its respondents, their respective answers, the sum and arithmetic mean of answers.

users in G_1	answers assume values [1–5]					sum of answers	arithmetic mean
R1	1	2	2	3	2	10	2
R2	5	2	4	5	4	20	4
R3	2	1	4	3	4	14	2.8
R4	3	3	3	3	3	15	3
R5	2	5	4	1	5	17	3.4
R6	4	4	4	5	4	21	4.2
R7	3	3	3	3	3	15	3
R8	1	1	2	3	1	8	1.6
R9	4	4	4	4	4	20	4
R10	4	5	4	5	4	22	4.4
R11	4	5	3	3	5	20	4
R12	5	4	3	2	5	19	3.8
R13	4	4	4	5	4	21	4.2
R14	4	5	5	4	5	23	4.6
R15	4	5	4	4	3	20	4
R16	3	2	5	3	4	17	3.4
R17	1	2	3	1	1	8	1.6

This level is considered to be the preparation step for the next levels. An example of answers for respondents belonging to *group 1* (an economic department with the scale of answers [1, 5]), the sum of answers and arithmetic mean of answers are shown in Table 2.

3.5. Aggregation at the department level. This level of aggregation should calculate the utility of given software within a particular group of respondents and should not be dependent on the aggregation used in Table 2. In order to envelop the aforementioned features of groups, we suggested aggregation by the relative quantifier *most of* of the structure *most of the respondents in a department have highly rated software k*.

In order to solve this task, the term *highly rated software* and the quantifier *most of* should be formalized. Regarding the latter, it is a usual relative fuzzy quantifier (Kacprzyk and Yager, 2001; Kacprzyk *et al*., 2000), in our case expressed by the increasing function (3).

The predicate *high opinion* is dependent on the properties of each respondent group, or on the structure of the questionnaire. From the minimal and maximal score among questionnaires (Table 2), the fuzzy granules *high* and *low* rates were created in the sense of Ruspini (1969) by uniformly covering the domain of scores (Tudorie, 2008). In this example, the linguistic variable *opinion* consists of two labels: *low* and *high*, as illustrated in Fig. 2.

The formalization is derived from the basic structure of linguistic summaries created by Yager (1982) in the

Table 3. Matching degrees of all respondents to the concept *positive opinion* calculated from the data in Table 2 by uniformly covering the domain of aggregated values; see Fig. 2.

users in G_1	matching degrees to *high opinion* for sum of answers	average of answers
R1	0	0
R2	1	1
R3	0.0161	0.0161
R4	0.339	0.389
R5	0.984	0.984
R6	1	1
R7	0.339	0.338
R8	0	0
R9	1	1
R10	1	1
R11	1	1
R12	1	1
R13	1	1
R14	1	1
R15	1	1
R16	0.984	0.983
R17	0	0

following way:

$$A^2_{lk}(A^1_{ilk}) = \mu_Q(y), \tag{6}$$

where y is the proportion of respondents (2) which provided a high opinion, in our case

$$y = \frac{1}{m_{lk}} \sum_{i=1}^{m_{lk}} \mu_{po}(A^1_{ilk}), \tag{7}$$

with m_{lk} being the number of respondents in group l evaluating software k and μ_{po} a membership function formalizing the predicate *positive opinion*.

Regarding the group of respondents G_1 from Tables 2 and 3, the validity of quantified aggregation (6) for the sums of answers is 0.532, whereas that for the arithmetic means of answers is 0.531. We observe that the validity is almost the same. Hence, there is no significant sensitivity in applying the sum or arithmetic mean at the first level. In the case of the geometric mean, the validity is 0.454. The difference is around 10%, but for the high number of categorical questions it might be higher. Thus, we conclude that the validity of the sentence *most of respondents in department G_1 have high opinion* is 0.531.

The result of this aggregation is in the unit interval. Thus, we can at the next level examine aggregation functions and apply the most suitable ones.

3.6. Aggregation at the software level. The result of aggregation in Section 3.5 assumes values from the [0, 1] interval, which allows applying a variety of aggregation functions. The aforementioned four classes of these functions are (Dubois and Prade, 2004) conjunctive, averaging, disjunctive and hybrid. These classes are able to cover a large variety of requirements, but the selection of a suitable one should be in agreement with the semantic meaning or expectations.

When each piece of software should be at least partially recognized in all departments, disjunctive functions are not a solution because of the absorbing element 1; i.e., it suffices that software is ideally evaluated in one department regardless of extreme poor evaluation in remaining ones. A similar observation holds for averaging functions having no absorbing element like the arithmetic mean (two extreme poor evaluations are compensated by two extreme positive evaluations). Observe that the most suitable function at the first level is the arithmetic mean. On the other hand, the geometric mean is a suitable option here due to the existence of the absorbing element 0. The solution might be a conjunctive function. But, values higher than minimal are either ignored (minimum t-norm) or the solution is lower than the lowest grade by departments (downward reinforcement). More details about these functions can be found in, e.g., the works of Beliakov *et al*. (2007) and Grabisch *et al*. (2009).

When the valuable software tool should be emphasized (upward reinforced by disjunctive behaviour) and at the same time poorly evaluated tools attenuated (downward reinforced by conjunctive behaviour), the solution is a hybrid function belonging to the uni-norm category (Calvo *et al*., 2002). A binary uni-norm function is depicted in Fig. 3, where e is the neutral element, i.e., $u(x, e) = u(e, x) = x$. Obviously, for $e = 1$ we get conjunctive functions, whereas for $e = 0$ we get disjunctive ones. By shifting e in the interval (0, 1) the sizes of four areas change.

A possible choice is a $3 - \Pi$ function suggested by Yager and Rybalov (1996),

$$v(S_k) = \frac{\prod_{l=1}^{L} x_l}{\prod_{l=1}^{L} x_l + \prod_{l=1}^{L}(1 - x_l)}, \tag{8}$$

where index k stands for the k-th software evaluated by

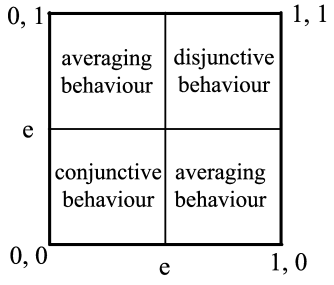

Fig. 3. Graphical interpretation of the uni-norm function.

all groups of users, x_l is the aggregated value from the previous step. In order to avoid indefinite results, we adopt $0/0 = 0$. The product in the numerator ensures assigning zero to any piece of software which does not pass the evaluation in at least one group of respondents (department).

An illustrative example is shown in Table 4, where values of G_1 for S_1 are taken from the previous subsections (Tables 2 and 3). The other values are provided directly without showing the whole procedure to illustrate the proposed method. Upward reinforcement holds for S_2, whereas downward reinforcement holds for S_5. For software S_1 and S_4, aggregation is behaving as an averaging function. An interesting case is S_6. Due to existence of the absorbing element 1 (for the disjunctive part of the domain), the solution is 1 regardless of whether the software is weakly evaluated in some departments, but software has not failed in any department.

Aggregation by the geometric mean is realized as

$$v(S_k) = \sqrt[L]{\prod_{l=1}^{L} x_l}, \tag{9}$$

where the variables have the same meaning as in (8). The solution is in Table 5.

Thus, reasonable choices are uni-norms and averaging functions having the absorbing element 0 like geometric mean. The ranks by (8) and (9) differ.

Table 4. Aggregation of software usability within all departments by the uni-norm function.

group / software	G_1	G_2	G_3	G_4	uni-norm (8)
S_1	0.532	0.151	0.850	0.350	0.3815
S_2	0.835	0.725	0.778	0.931	0.9985
S_3	0.250	0.320	0.410	0.220	0.0298
S_4	0.630	0.826	0.253	0.366	0.6125
S_5	0.110	0.220	0.320	0.180	0.0035
S_6	0.56	1.000	0.220	0.580	1.000
S_7	0.886	0.900	0.135	0.000	0.000

Table 5. Aggregation of software usability within all departments by the geometric mean.

group / software	G_1	G_2	G_3	G_4	geom. mean (9)
S_1	0.532	0.151	0.850	0.350	0.393
S_2	0.835	0.725	0.778	0.931	0.816
S_3	0.250	0.320	0.410	0.220	0.291
S_4	0.630	0.826	0.253	0.366	0.468
S_5	0.110	0.220	0.320	0.180	0.193
S_6	0.560	1.000	0.220	0.580	0.517
S_7	0.886	0.900	0.135	0.000	0.000

The main requirement that software should pass in all departments is met. The geometric mean is a simple multiplicative scoring (all inputs are mandatory regardless of their importance), whereas the arithmetic mean is a simple additive scoring where all inputs are optional (Dujmović, 2018). These models are useful as parts in a complex evaluation model (in our case, implemented into the first and third level). The geometric mean provides the result between the best and the worst evaluation for all cases, whereas uni-norms provide such a solution only for tools which are badly evaluated in several departments and positively in others.

4. Discussion

The theory of usability has been developed to explain human behaviour in decision making (Olson, 1996). Therefore, this approach should not rely on preferential independence and expectation that the usability of the whole can be calculated as the weighted sum or weighted average of utilities of evaluated parts (Dujmović, 2018; Hensher *et al.*, 2015). Thus, we should include logic properties of human evaluation, which is not a linear one. Further, using both numerical and linguistic data and information enables more effective decision-making (Piegat and Pluciński, 2015).

This paper, however, focuses on a different utility task, but the same principle should be met. During a considerable time a company might have various tools within departments: MS Office, Open Office, Libre Office; various web browsers, and so forth. This might cause a mess in the inventory and maintenance of updates. In this work, the goal is to find tools which are positively evaluated among all departments by the majority of workers.

The suggested solution is flexible and robust because it is able to cover the evaluation of the usability of software used across the entire institution regardless of different features within departments. The experiments were carried out within a company running its business in the IT sector. Generally, this approach can be used for a large variety of tasks focusing on the evaluation of the usability of software tools considered, or for the evaluation of a variety of topics where respondents are divided into several clearly distinctive groups. For instance, this approach may be applicable in surveys related to perspectives of various big-data sources for departments of a company, or within different research groups. The same holds for evaluating suitability of methodologies considered or collecting opinions of the several suggested acts to cope with the recognized problem in an institution or broaden and support informed decisions. In such tasks, evaluators should focus their work only on the given task to create the relevant number of questions and possible answers and adjust them to

particularities of each group or department.

Expressing answers by numerical values is recognized as a problem. Users may not be careful in filling questionnaires, i.e., they can fill in the neighbouring value in categorical answers and may not associate properly the intensity of feelings to numbers like they can without difficulties associate feelings to linguistic terms. This problem was recognized by random checking of answers by re-asking the respondents. A higher number of such cases may significantly bias the aggregation, even when uncertainties are managed by fuzzy logic. The solution expresses answers linguistically.

In surveys within different groups of respondents, the granularity of terms can be adjusted to particularities of each user group, similarly as for numerical categories. We also considered the fact that user groups often express their observations by linguistic terms, rather than numbers and different users groups prefer various granularities of answers (Morente-Molinera *et al.*, 2018). For instance, for the less skilled and educated users, we can offer a set of three terms, whereas for experts we may offer a larger set, e.g., a set of seven terms. Generally, the number of terms should be within the range of 3 to 9, where 9 is the upper bound for cognitive processing of information (Miller, 1956). A possible mapping from linguistic interpretation to numbers for the afore–explained aggregations is shown in Table 6. These scales can be straightforwardly transformed into the unit interval, if required. Generally, for the first level of the suggested approach this transformation does not influence the solution.

5. Conclusion

Evaluation of software usability in a company is not an easy task. The quality of a software product usually is involved in the process of software development. Evolution of the testing software quality has a long history, and software engineering is changed according to the new object-oriented programming paradigm. The main role in testing the quality is not played by a real user, but by a tester in an IT company who is not familiar with the domain area of the user. Future directions for software quality testing are more oriented to user satisfaction, and the creation of a computational model for a knowledge-based system should bring new possibilities in software quality testing.

Usually, rule-based systems are used to express knowledge regarding software usability. However, these systems, as shown in the experiments, might be very large, especially when the institution has a larger number of software tools, complex questionnaires and higher numbers of departments. This work demonstrated the benefits and drawbacks of fuzzy rule-based systems in measuring software utilities. There are recognized benefits of evaluating the software tools independently, i.e., each department evaluates specific tools required for their respective daily work.

This work recognized the demand for evaluating and ranking software tools used in a company. In order to solve this problem, we suggested three levels of aggregation for the schemes plotted in Fig. 1 by fuzzy logic. The aggregation at the questionnaire level is realized by the sum or arithmetic mean because other mean functions are less suitable. Thus, the quantified aggregation in the second step is not sensitive to the aggregated option at the first level and, moreover, it is irrelevant—the transformation from the linguistic answers to the numerical intervals. The result at the second level is in the unit interval, which opens the space for a large variety of aggregation functions at the third aggregation level. At the third level, suitable functions are uni-norms due to their full reinforcement property or the geometric mean (the opposite observation than for the first level due to averaging behaviour and having zero as an absorbing element).

For future research, we would like to examine other aggregation functions and carry out further experiments regarding computational effort. The suggested aggregation model has broader applicability such as customer satisfaction, marketing evaluation, etc. Furthermore, this approach may be used for evaluating opinions regarding the benefits and drawbacks of various big-data sources within departments in a company, or collecting opinions of the several suggested internal regulations (directives) to cope with the recognized problem. Based on the suggested aggregation model, it is possible to develop a valuable system for supporting informed decisions. The applicability in the mentioned and possibly others fields might be a topic for future research.

Table 6. Categorical answers expressed by linguistic interpretation and a possible mapping into the interval [0, 10] for expressing the answers in Table 1.

three terms		five terms		seven terms	
term	value	term	value	term	value
negative	1	very negative	1	very negative	0.5
neutral	5	negative	3	negative	2
positive	9	neutral	5	more neut. than neg.	3.5
		positive	7	neutral	5
		very positive	9	more neut. than pos.	6.5
				positive	8
				very positive	9.5

Acknowledgment

This work is partially supported by a project VEGA no. 1/0373/18 entitled *Big Data Analytics as a Tool for Increasing the Competitiveness of Enterprises and Supporting Informed Decisions* of the Ministry of Education, Science, Research and Sport of the Slovak Republic.

References

Albert, W. and Tullis, T. (2013). *Measuring the User Experience, Collecting, Analyzing, and Presenting Usability Metrics (Interactive Technologies), 2nd Edition*, Elsevier, Amsterdam.

Allen, I.E. and Seaman, C. (2007). Likert scales and data analyses, *Technical report*, QP-Quality Progress, http://asq.org/quality-progress/2007/07/statistics/likert-scales-and-data-analyses.html.

Bavdaž, M. (2010). Sources of measurement errors in business surveys, *Journal of Official Statistics* **26**(1): 25–42.

Bavdaž, M., Biffignandi, S., Bolko, I., Giesen, D., Gravem, D. and Haraldsen, G. (2011). Final report integrating findings on business perspectives related to NSIS' statistics, *Technical report*, Deliverable 3.2., FP7 Blue-Ets Project, European Commission, Brussels, https://cordis.europa.eu/project/rcn/94081/results/en?rcn=143042.

Beliakov, G., Pradera, A. and Calvo, T. (2007). *Aggregation Functions: A Guide for Practitioners*, Springer-Verlag, Berlin/Heidelberg.

Calinescu, M. and Schouten, B. (2012). Adaptive survey designs that minimize nonresponse and measurement risk, *Technical report*, Statistics Netherlands, The Hague/Heerlen.

Calvo, T., Kolesárová, A., Komorníková, M. and Mesiar, R. (2002). Aggregation operators: Properties, classes and construction methods, *in* T. Calvo *et al.* (Eds), *Aggregation Operators: New Trends and Applications*, Physica, Heidelberg, pp. 3–104.

Dubois, D. and Prade, H. (2004). On the use of aggregation operations in information fusion processes, *Fuzzy Sets and Systems* **142**(1): 143–161.

Dujmović, J. (2007). Continuous preference logic for system evaluation, *IEEE Transactions on Fuzzy Systems* **15**(6): 1082–1099.

Dujmović, J. (2018). *Soft Computing Evaluation Logic: The LSP Decision Method and Its Applications*, Wiley/IEEE Computer Society, Hoboken, NJ.

Grabisch, M., Marichal, J.-L., Mesiar, R. and Pap, E. (2009). *Aggregation Functions*, Encyclopedia of Mathematics and its Applications, Cambridge University Press, Cambridge.

Greiner, L. and White, S. (2019). What is ITIL? Your guide to the it infrastructure library, in digital magazine CIO from IDG, https://www.cio.com/article/2439501/infrastructure-it-infrastructure-library-itil-definition-and-solutions.html.

Gupta, M. and Qi, J. (1991). Theory of t-norms and fuzzy inference methods, *Fuzzy Sets and Systems* **40**(3): 431–450.

Hensher, A., Rose, J. and Greene, W. (2015). *Applied Choice Analysis*, Cambridge University Press, Cambridge.

Herrera, F. and Martínez, L. (2001). A model based on linguistic 2-tuples for dealing with multigranular hierarchical linguistic contexts in multiexpert decision-making, *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics* **31**(2): 227–234.

ISACA (2018). Service it governance professionals, COBIT5, an ISACA framework, http://www.isaca.org/cobit/pages/default.aspx.

ISO (2011). Systems and software engineering, ISO/IEC 25010:2011: Systems and software quality requirements and evaluation (square). System and software quality models, https://www.iso.org/standard/35733.html.

ISO (2018). ISO, online browsing platform, ISO 9241-11:2018: Ergonomics of human–system interaction. Part 11: Usability: Definitions and concepts, https://www.iso.org/standard/63500.html.

Kacprzyk, J. and Yager, R. (2001). Linguistic summaries of data using fuzzy logic, *International Journal of General Systems* **30**(2): 133–154.

Kacprzyk, J., Yager, R.R. and Zadrożny S. (2000). A fuzzy logic based approach to linguistic summaries of databases, *International Journal of Applied Mathematics and Computer Science* **10**(4): 813–834.

Králiková, L. (2017). *Testovanie efektívnosti softvéru v podnikovej praxi z hľadiska užívateľov (Software Effectiveness Testing in Business Practice from a User Perspective)*, Master thesis, University of Economics in Bratislava, Bratislava.

Likert, R. (1932). A technique for the measurement of attitudes, *Archives of Psychology* **22**(140): 1–55.

Miller, G. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information, *Psychological Review* **63**(2): 81–97.

Morente-Molinera, J., Kou, G., Pérez, I., Samuylov, K., Selamat, A. and Herrera-Viedma, E. (2018). A group decision making support system for the web: How to work in environments with a high number of participants and alternatives, *Applied Soft Computing* **68**: 191–201.

Olson, D. (1996). *Decision Aids for Selection Problems*, Springer-Verlag, London.

Pavlík, L. (2018). Metrics for evaluating information systems, Posterus, Portl pre odborn publikovanie, http://www.posterus.sk/?p=18957.

Piegat, A. and Pluciński, M. (2015). Computing with words with the use of inverse RDM models of membership functions, *International Journal of Applied Mathematics and Computer Science* **25**(3): 675–688, DOI:10.1515/amcs-2015-0049.

Rakovská, E. and Hudec, M. (2020). Two approaches for the computational model for soft-ware usability in practice, *in* J. Kacprzyk *et al.* (Eds), *Information Technology, System Research and Computational Physics, ITSRCP 2018,* Advances in Intelligent Systems and Computing, Vol. 945, Springer, Cham, pp. 191–202.

Ruspini, E. (1969). A new approach to clustering, *Information and Control* **15**(1): 22–32.

Seffah, A., Kececi, N. and Donyaee, M. (2001). QUIM: A framework for quantifying usability metrics in software quality models, *2nd Asia-Pacific Conference on Quality Software, Hong Kong*, pp. 311–318.

Snijkers, G., Haraldsen, G., Jones, J. and Willimack, D. (2013). *Designing and Conducting Business Surveys*, Wiley, Hoboken, NJ.

Tudorie, C. (2008). Qualifying objects in classical relational database querying, *in* J. Galindo (Ed.), *Handbook of Research on Fuzzy Information Processing in Databases*, Information Science Reference, Hershey, pp. 218–245.

Yager, R. (1982). A new approach to the summarization of data, *Information Sciences* **28**(1): 69–86.

Yager, R. and Rybalov, A. (1996). Uninorm aggregation operators, *Fuzzy Sets and Systems* **80**(1): 111–120.

Zadeh, L. (1965). Fuzzy sets, *Information and Control* **8**(3): 338–353.

Zimmermann, H. (2001). *Fuzzy Set Theory and Its Applications*, Kluwer Academic Publishers, Dordrecht.

Eva Rakovská received her PhD degree in applied informatics at the University of Economics in Bratislava, Slovakia, in 2010. She obtained her MS degree in mathematics at Comenius University, and worked as a programmer and an IT developer in various companies till 2001. Since then, she has been an assistant professor at the University of Economics in Bratislava, Faculty of Economic Informatics. Her research interests are in applied informatics and artificial intelligence in education and knowledge management. The particular emphasis is on applications of soft computing and expert systems.

Miroslav Hudec is an associate professor at the University of Economics in Bratislava, Faculty of Economic Informatics. He received his MS and PhD degrees from the University of Belgrade, where he has recently took the position of a visiting professor. His work is mainly focused on fuzzy logic, knowledge discovery, and information systems. He has published more than 50 articles in these fields. He has been a member of program committees of several international conferences and serves as an editorial board member for several journals.

Received: 20 October 2018
Revised: 24 April 2019
Re-revised: 27 June 2019
Accepted: 3 July 2019