Formalization of software requirements for information systems using fuzzy logic

Y S Yegorov¹, V R Milov¹, A S Kvasov¹, S N Sorokoumova², O V Suvorova³

¹ Nizhny Novgorod State Technical University n.a. R.E. Alekseev, 24, Minina St., Nizhny Novgorod, 603950, Russia
² Nizhny Novgorod State University of Architecture and Civil Engineering, 65, Ilyinskaya St., Nizhny Novgorod, 603950, Russia
³ Minin Nizhny Novgorod State Pedagogical University, 1, Ulyanova St., Nizhny Novgorod, 604950, Russia

E-mail: ckar@list.ru

Abstract. The paper considers an approach to the design of information systems based on flexible software development methodologies. The possibility of improving the management of the life cycle of information systems by assessing the functional relationship between requirements and business objectives is described. An approach is proposed to establish the relationship between the degree of achievement of business objectives and the fulfillment of requirements for the projected information system. It describes solutions that allow one to formalize the process of formation of functional and non-functional requirements with the help of fuzzy logic apparatus. The form of the objective function is formed on the basis of expert knowledge and is specified via learning from very small data set.

1. Introduction

At present, in Russia, software development (software) is governed mainly by GOSTs of the 19th and, in the case of the creation of automated systems, the 34th series. These standards define the list of stages, the sequence and order of their implementation, the forms of reporting documents, etc. The specified GOSTs in terms of software development are outdated and do not meet modern trends in the IT industry.

In practice, the classical cascade model is recognized as inefficient due to the lack of flexibility, which determines the low ability to respond to changes in requirements for the information system (IS), which, as a rule, leads to breakdown of the project deadlines [1]. In this regard, there is a contradiction between the current technological level of development and acceptance of IS and obsolete requirements of Russian regulatory documents and government standards.

Integration of modern flexible methods of designing IS at the stages of the cascade model of the life cycle of software development, as well as intellectual support of decision-making processes, will allow improving both methodological approaches and the development process, which will ensure the availability of high-quality software that meets all specified requirements (verification), as well as the expectations of stakeholders (validation). If verification is provided through testing, then validation is performed when the IS is launched and during its operation by monitoring, the criterion is the degree of satisfaction of all stakeholders (stakeholders) with the operation of a particular IP.
2. Flexible methods of software development

Flexible methods of development make it possible to eliminate a number of drawbacks of the classical method of waterfall, which are the lack of flexibility and formal management of the life cycle of the project to the detriment of time, cost and quality. The lack of flexibility in the development process leads to inability to respond to emerging changes in IS requirements, and can ultimately lead to budget overrun, the disruption of the project delivery deadline and the lack of demand for the software [2].

The reasons for such changes may be adjustment of software requirements, errors made in the formation of requirements, strict limitations imposed by requirements, changes in the subject area, etc. It is noted [3] that a change in the functional requirements for software in the general case is almost inevitable, is a necessary condition for validation, and is also accepted as one of the principles of flexible methodology in [4].

A popular flexible method is Scrum [5]. Its basic principles allow us to provide the Customer with workable software with new capabilities in fixed and fairly short iterations, called sprints. At the same time, the collection of requirements, design, development of software and its demonstration to the Customer can be performed in each such integration. However, according to the classical cascade method, in the general case, all steps are performed sequentially [2].

The introduction of the same flexible methods is connected with the change of the main technological and organizational processes of software development, ways of interaction with the Customer, planning, control of development, delivery and acceptance of the project, etc. In doing so, it provides a gradual iterative implementation of the project with constant feedback from the end user. The project develops in small iterations, within each of which in the general case a relatively small increment in the functional of the system is realized.

Formation and management of requirements are important components of the process of designing information systems. A flexible development methodology focuses on the dynamic formation of requirements, which enables interested parties to formulate new requirements based on the results of each iteration. However, the dynamic formation of requirements complicates the process of managing them, reduces the predictability of the development process and makes planning work difficult. In some cases, the new requirements contradict what has already been implemented, which breaks the architecture of the software developed earlier and can lead to a change in the scope of work, the timing of the project or its cost. The project, as a rule, is carried out according to the previously agreed and approved product requirements document (PRD) and the terms and cost of work are often rigidly defined by contractual obligations. Thus, there is a need to resolve the contradiction, between the need to dynamically change the requirements and perform a well-defined task list in the PRD [2], i.e. continuous accounting (reconciliation) of all requests for change can lead to a significant increase in the timing and cost of the project, but it may not improve the degree of satisfaction of stakeholders.

3. Determination of requirements in accordance with the objectives of the development of information systems

Despite the emergence of a number of new approaches to the management of IT-projects, many software products and projects continue to slowly degrade without doing any good. As a result, a huge amount of time and money is wasted due to incorrect initial assumptions, unfocused, poor communication, misunderstandings and discrepancies with the global goals of organizations [6].

Optimal quality of products and services can be achieved by being compliant with requirements. So requirements have to be or are defined, exchanged and agreed and the “solutions” (design, products, and services) have to be compliant with them.

In the design of information systems in the field of problems, the application of demand management methods that are goal-oriented is promising.

A goal model is an element of requirements engineering that may also be used more widely in business analysis. Goal modeling is particularly useful in the early stages of a project, since it allows one to consider how the target system meets business goals. Requirements can be considered complete if they fulfill all the goals in the target model.
To support decision making in the design of information systems, it is necessary to determine to what extent compliance with requirements contributes to the achievement of business objectives. Such analysis with iterative development allows you to focus on the implementation of functionality that meets the requirements that have the most significant impact on achieving business goals. Impact mapping [6] is an approach to establishing an influential development in the information system to achieve business goals. An impact map is a visualisation of scope and underlying assumptions, created collaboratively by senior technical and business people.

Improving the effectiveness of life-cycle management of information systems can be achieved by assessing the functional relationship between requirements and business objectives. To this end, it is proposed to evaluate the degree of achievement of business objectives at each iteration of the development and measure how much the requirements are met. As a result, a dataset is formed, which allows the use of machine learning and a neural network model of relation between goals and requirements. For the training of neural networks, the developed procedures can be used [7]. However, this problem should be solved in conditions of small [8] and even very small samples, which leads to the need to use specific approaches for machine learning [9]. An increase in the adequacy of the models being formed is possible when using the knowledge of experts formalized in particular by fuzzy logic rules [10].

Let us consider in more detail the approach to measuring the degree of fulfillment of requirements.

4. An approach to the formalization of requirements using a fuzzy logic device

The requirements are verified during the acceptance tests of the IS. The purpose of verification is to establish the quality of the software product by checking it for compliance with user-defined, functional and non-functional requirements. In this case, the constraints imposed by the requirements act as a criterion for evaluating the quality of software. Often, such check is performed formally, and non-compliance with non-critical restrictions that do not reduce the value of the developed software can lead to the fact that IP will not be put into trial operation.

In ITIL, with respect to information services, a value-based model is proposed, based on a combination (“on AND”) of utility and warranty that can be considered in terms of fulfilling functional and non-functional requirements, respectively. In particular, the warranty is defined as combining such characteristics as availability, capacity, continuity, security.

Let us denote a set of requirements for the characteristics of the nth component of the IS as \( R_{nl} \), \( l = 1, L_n \). Usually in practice, when choosing system-technical solutions, the requirements are formalized by setting the areas for corresponding characteristics \( y_{nl} \) in the form of one-sided or two-sided inequalities, for example, \( y_{nlL} \leq y_{nl} \leq y_{nlH} \). The allowed values of the discrete characteristics can be specified by enumeration.

To take into account the requirements, let us introduce indicator variable \( r \) such that \( r = 1 \) if requirement \( R \) and the corresponding inequality are satisfied, and \( r = 0 \) otherwise. As a result, a number of feasible solutions include only those alternatives for which all requirements for all IS components are met. Let us note that with such rigid method of specifying requirements, a set of feasible solutions can contain a relatively small number of alternatives and even be empty in some cases. In addition, it is very difficult to take into account the characteristics when ranking alternative configurations, which is in demand in systems for supporting the adoption of design solutions. Finally, flexibility in setting requirements is greatly reduced.

In this connection, an approach is proposed for the formalization of requirements using a fuzzy logic apparatus. To formalize the requirements, instead of specifying the threshold values of the characteristics with the definition of two grades of fulfilling the requirements (it is not and is not being fulfilled), it is suggested to specify the requirements using the membership functions (MF) for fuzzy
set \( r = \mu_R(y) \). In this case, the degree of fulfillment of requirement \( R \) is defined as the value of corresponding membership function \( r \in [0; 1] \).

As the main options for formalizing the requirements related to the quantitative characteristics of the IS components, it is proposed to use the MF [5]: a linear s-shaped membership function (\( \gamma \) class MF); a linear z-shaped membership function (\( L \) class MF) and a trapezoid membership function.

Thus, the application of OP for the formalization of requirements can be considered as a transition from rigid constraints with one threshold to two-threshold soft constraints. This form of presentation allows decision-makers to understand the degree of freedom that exists in choosing the solutions that provide the highest overall score, by seeking a compromise between characteristics that meet different requirements [11].

Given there is the need to meet all the requirements and the inability to compensate for the non-fulfillment of certain requirements by the performance of others, let us determine the compliance coefficients with the use of the t-norm of the “minimum” type. Let all \( L \) requirements are ordered and the requirements from the first to \( L_F \) are functional, and the subsequent requirements from \( L_F +1 \) to \( L \) are nonfunctional. Then the expressions for the coefficients of correspondence to functional and nonfunctional requirements will be:

\[
K_F = \min_{l=1,L_F} \eta_l, \quad K_{NF} = \min_{l=L_F+1,L} \eta_l.
\]

Here \( \eta_l = \mu_{R_l}(y_l) \) is the degree of fulfillment of the lth requirement. If it is necessary to take into account the significance of individual requirements, a weighted t-norm can be used:

\[
K_F = \min_{l=1,L_F} \{1 - w_l(1 - \eta_l)\}, \quad K_{NF} = \min_{l=L_F+1,L} \{1 - w_l(1 - \eta_l)\},
\]

where \( w_l \) is the weight that determines the significance of the lth requirement.

Based on coefficients \( K_F \) and \( K_{NF} \), using the parametric families (classes) of t-norms [12], objective function \( V = T_C(K_F, K_{NF}; p) \) is defined that allows one to characterize the degree of achievement of business objectives. The following families of Dombi, Sugeno-Weber, Yager, Hamacher, Frank t-norms are considered. The parameter \( \hat{p} \) value and t-norm class \( \hat{C} \) are determined via learning from a very small data set.

5. Conclusion

Thus, a method of managing the amount of work on the formation and implementation of software requirements under conditions of flexible development can become a decision support system that allows formalization of requirements using a fuzzy logic apparatus. Clarification and specification of functional requirements based on the proposed approach will allow one to vary the complexity of the implementation of PRD constraints, carry out a preliminary evaluation of the developer's proposals for comparing the value of alternative options and decision making, and also provide the opportunity for more flexible verification of requirements. The resulting evaluation of the functional relationship between the degree of fulfillment of requirements and the achievement of a business goal allows us to make informed decisions in the design and development of information systems.

References

[1] Futrell R T, Shafer D F, Shafer L I 2002 Quality Software Project Management (Prentice Hall)
[2] Karpov A V, Karpov V V 2016 Programnye producty i sistemy 1(113) 5–12
[3] Wiegers K, Beatty J 2013 Software Requirements (Microsoft Press)
[4] Baranov V G, Sedakov D V, Milov V R 2017 Informacionno-izmeritelnye i upravliaushie sistemy 8 26–33
[5] Sutherland J 2014 The Art of Doing Twice the Work in Half the Time (Currency)
[6] Gojko A 2012 Impact Mapping: Making a big impact with software products and projects (Provoking Thoughts)
[7] Kondrat'ev V V, Milov V R 2002 Doklady Mathematics 66(2) 301–303
[8] Milov V R, Savchenko A V 2017. 2017. Communications in Computer and Information Science, 661 338–345
[9] Andonie R 2010 Int. J. of Computers, Communications & Control, 3 280–291
[10] Avdeenko T V, Makarova E S 2017 J. Phys.: Conf. Ser. 803 012008
[11] Hull E, Jackson K, Dick J 2011 Requirements Engineering (Publisher: Springer-Verlag London)
[12] Klement E P, Mesiar R, Pap E Triangular norms (Springer Netherlands)