Запропоновано підхід, що реалізує валідацію вимог до якості програмних сервісів (на прикладі вимог до продуктивності й надійності) шляхом їхнього порівняння з інтерактивними оцінками змодельованої якості сервісів зацікавленими особами

Ключові слова: якість обслуговування, продуктивність сервісів, надійність сервісів, валідація вимог, оцінювання якості

Предложен подход, реализующий валидацию требований к качеству программных сервисов (на примере требований производительности и надежности) путем их сравнения с интерактивными оценками смоделированного качества сервисов заинтересованными лицами

Ключевые слова: качество обслуживания, производительность сервисов, надежность сервисов, валидация требований, оценивание качества

We introduce an approach for validating quality of service (QoS) requirements (exemplified by performance and reliability requirements) by comparing them to interactive assessments of simulated service qualities by business stakeholders

Key words: quality of service, service performance, service reliability, requirements validation, quality assessment

УДК 681.518

TOWARDS VALIDATING QOS REQUIREMENTS USING STAKEHOLDER ASSESSMENTS OF SIMULATED SERVICE QUALITIES

V.A. Shekhovtsov
Кандидат технических наук, доцент, докторант
Кафедра автоматизированных систем управления
Национальный технический университет «Харьковский политехнический институт»
ул. Фрунзе, 21, г. Харьков, Украина, 61002
Контактный тел.: 050-965-34-27
E-mail: shekvl@yahoo.com

1. Introduction

The importance of a requirement validation as a process of checking if the defined set of software requirements meets the stakeholders' expectations is widely acknowledged [1-3]. It allows the business stakeholders and the development team (as a holder of the respective requirements) to agree in what they expect from the system. This is especially important for service-oriented systems as dealing with requirements for such systems require knowledge of the possible uses of the respective services which is difficult to obtain without the involvement of stakeholders.

The research we present in this paper is a part of addressing the problem of stakeholder involvement into the development of service-oriented systems. In [4, 5] we proposed the ISAREAD-S framework to address the problem aimed at investigating the ways to support such involvement in a form of assessing the perceived quality (exemplified by performance and reliability) of the service-oriented systems in their usage context. To implement such support we plan to elaborate a set of simulation-based methods aimed at making QoS (quality of service) assessment mechanisms accessible to the business stakeholders and using their assessments as a foundation for software development activities related to early stages of the software lifecycle.

This paper is devoted to establishing requirements validation policies to be integrated into this framework. Their purpose is to check if external requirements are valid from the point of view of stakeholders (expressed via quality assessments).

The paper has the following structure. Section 2 describes the current state of the art and formulates the problem statement, Section 3 shows the principles of the existing service-level and process-level procedures (mechanisms) for organizing the interaction with stakeholders; these mechanisms form the foundation for the validation solutions proposed in the paper, Section 4 outlines the proposed approach introducing a policy for validating external QoS requirements using stakeholder assessments obtained as a result of applying the mechanisms, Section 5 makes conclusions and describes the directions for future research.

2. State of the art and problem statement

Current techniques addressing requirements validation problem include generating narrative description of the requirements model for informal analysis [6], formal methods [2], using scenario-based and goal-oriented techniques [1]. The main disadvantage of applying such techniques to the problem of validating QoS requirements is that stakeholders cannot experience the system before validating process starts. As a result, the validation becomes biased towards the view of the IT people and the quality of the resulting requirements suffers. It is desired to make validation process closer to requirements verification process (when the stakeholders check if implemented system satisfies the requirements) but replace the implemented system with its simulation model.

As a result of reviewing the state of the art we can formulate research questions defining the problem statement.

We start from the general question introduced in [5]: *How to involve business stakeholders into the development process for service-oriented software systems as a means of control for the performance and reliability of the produced artifacts?* This question is addressed by proposing the ISAREAD-S framework [4, 5]; it provides low-level *mechanisms* for interactive assessment of simulated service performance and reliability; for the convenience of the readers we present an outline of these mechanisms in the next section.

Prior to introducing high-level policies based on the proposed mechanisms, we address two research problems related to allowing simulations depend on the artifacts of the development process. First problem is related to the idea of making simulations reflect the chosen software architecture; it leads to the research question: *How to make service quality simulations depend on the software architecture?* To answer, we need to investigate how the architecture affects simulation parameters by addressing the question: *What is the dependency between the software architecture and the factors influencing service qualities?* An example of such dependency could be the situation when the chosen architecture makes it possible to increase performance by reducing the network load. To solve this problem, we proposed to establish specific adapters (namely, architecture and resource adapters [7, 8]) to reflect dependencies between external information and input values for the assessment mechanisms.

The knowledge obtained so far allows us to follow the mechanism-policy separation principle by elaborating higher-level requirements engineering policies based on the proposed assessment mechanisms. As we define the problem of requirement validation as checking if some (e.g. externally defined) performance and reliability requirements are valid from the point of view of stakeholders, we can formulate the specific research question related to the topic of this paper: *How to validate performance and reliability requirements using the mechanism of interactive assessment of simulated service qualities?* To answer this question, it is necessary to establish the set of necessary procedures which define *requirements validation policy.* It should rely on both assessment mechanisms and the techniques allowing simulations depend on the artifacts of the development process.

3. ISAREAD-S foundations and assessment mechanisms

In [5] we described the proposed approach to establish service-level and process-level assessment mechanisms. In this section (as in [7, 8]), we briefly outline this approach to the degree necessary to understand the proposed validation solutions.

3.1. Modeling service quality attributes and requirements

We need to provide the definition of quality for ISAREAD-S and the means of conceptualizing the service quality (QoS) attributes and requirements. We plan to use Quality-Aware Predesign Model for Services QAPM-S [9] for this purpose (extended to satisfy the needs of ISAREAD-S project).

QAPM-S is based on Klagenfurt Conceptual Predesign Model KCPM [3] as it models service operations with the notion of operation-type (defining the operations, their actors and objects/parameters; these parameters are modeled with the notion of thing-type generalizing attributes, entities or values); it also uses tabular model representation (called glossary) which is well understood by stakeholders [10]. It models service quality as a hierarchy of quality characteristics, represents the facts that quality characteristics influence each other and that stakeholders perceive qualities differently, follows aspect-oriented paradigm [11] in representing service quality and functionality as separate concerns.

3.2. Service-level mechanisms

IAS mechanisms (short for Interactive Assessment of Services) aim at an assessment of simulated service qualities at the level of the particular service. According to the model-driven methodology [12, 13] it is necessary to have two mechanisms of this kind: IASC (for model composition) and IASE (for its execution). IASC inputs include the set of qualities of interest to be simulated and assessed and the set of factors influencing the simulation (simulation parameters). To get the integrated quality simulation model, we compose simulation modules corresponding to the qualities of interest and the necessary parameters together with the base simulation structure. Also, we integrate into this model the set of user interaction models for the qualities of interest. The resulting service-level simulation and assessment model becomes the IASC output. It is transferred to IASE for standalone execution.

IASE is responsible for execution of both simulation and assessment interaction submodels of IASM. The input for every IASE run is the set of parameter values corresponding to the parameters used to build IASM. As a result of the run, the set of simulated values for the qualities of interest is obtained and presented to the service user for assessment via interaction processes described by interaction models integrated into IASM. The IASE outputs are this set of simulated qualities and the set of assessment results.

3.3. Process-level mechanisms

IAP mechanisms (short for Interactive Assessment of Processes) aim at interactive assessment of simulated service qualities in context of usage processes at the level of the particular process, in particular: IAPC (for model composition) and IAPE (for model execution). They rely on service-level mechanisms dealing with individual services.

IAPC forms the simulation model of the usage process making it ready for interactive assessment of service qualities. It combines the control flow model (CFM) for the usage process (conforming to the network BPM notation such as BPMN) with the role model for the usage process. The role model includes the set of roles defined for process participants (clerk, manager etc), the sets of interaction activities

for different roles (they make participants affect the state of the process simulation), the sets of assessment activities for different roles (they correspond to the services of interest to be simulated and assessed by stakeholders) and the sets of qualities of interest and necessary parameters defined for every service of interest.

While composing the integrated model IAPM for the process, IASC creates the IASM model for every service of interest; this model later becomes integrated into IAPM. For every interaction activity, a mechanism for constructing the interaction model is invoked and the resulting interaction model is also integrated into IAPM. The resulting model will contain the simulation logic defined by CFM for the process and simulation submodels of different IASM models (for the services of interest); the assessment logic defined by interaction submodels of these IASM models; the interaction logic defined for all interaction activities.

The IAPM is executed by IAPE. Every run is presented to the stakeholder belonging to the particular role. During the run, the basic simulation flow is managed by the model derived from the CFM of the usage process; when the logic of the run requires invoking an activity representing the service of interest, the simulation of its qualities and the assessment interaction logic are handled by IASE invoked for its IASM. IASE inputs are parameter values for all the slots of this service; when this logic requires interacting with the simulation, the logic of this interaction is handled by the corresponding interaction mechanism. The outputs for IAPE run include the set of all simulated quality values for all the services of interest and the set of corresponding assessment results.

4. Outline of the proposed solution

Assessment mechanisms can be used as building blocks for high-level policies. Most of them are supposed to be used at early stages of the system development lifecycle. We propose to add to the set of such policies the *requirements validation policy* intended to solve the problem stated in this paper.

The proposed policy starts from the set of *requirements under validation* (RUV) represented as an instance of QAPM-S; they in most cases originate from external *requirement sources* such as natural language-processing techniques developed for the NIBA project [3, 14]). This instance defines the set of services and qualities of interest. To build it, we need to perform a conversion from external requirement formats to QAPM-S representation. Different conversions need to be performed for different formats, the specifics of the particular conversion is supposed to be handled by an instance of *Requirement Source Interface Model* (RSIM) defined for the particular source; we plan to establish a library of such models.

After the first step, we believe it is possible to use two alternate approaches to requirements validation: *assessment-centered* and *requirement-centered* validation policies. Below we will describe them in more detail.

4.1. Assessment-centered validation policy

The first approach is to perform validation by comparison between assessments obtained from stakeholders and those originated from RUV. We call such approach *assessment-centered validation policy* (its BPMN representation is shown on Fig. 1).

To implement this policy, we need to convert every RUV into a set of artificial *requirement-based assessments* (RBA set) with elements corresponding to simulated qualities. For example, for the simulated quality "*the latency for the service X is 0,6 sec*" the external latency requirement with a threshold of 0,2 *sec* can be converted into a requirement-based assessment "*the latency for the service X is three times worse than specified in an external requirement*".

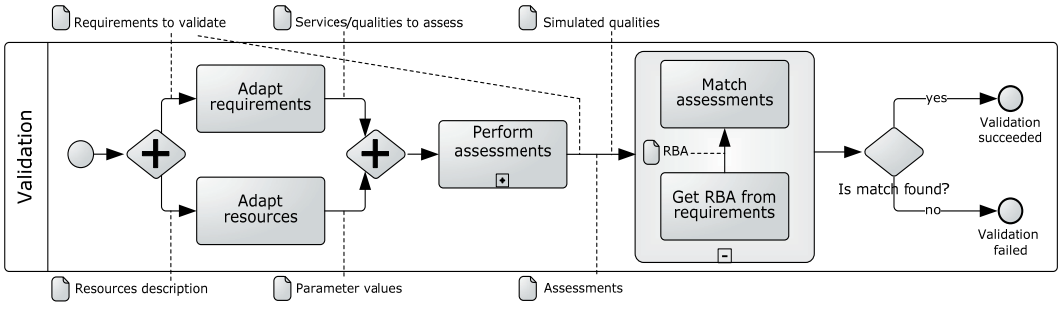

Fig. 1. Assessment-centered validation policy

Assessment-centered validation policy consists of the following activities:

1. Deriving QAPM-S representation of the set of services and qualities to assess from RUV as described above; it becomes an input for an assessment mechanism. In parallel, the set of parameter values (obtained e.g. using the resource adapter [7]) also becomes an input for that mechanism.

2. Running the assessment mechanism as defined in [5] and, as a result of this run, obtaining the set of stakeholder assessments together with the set of simulated qualities. These assessments can be precise or imprecise (fuzzy) depending on the allowed degree of uncertainty.

3. Converting RUV to RBA. This conversion is done for every instance of simulated quality obtained from the run of the assessment mechanism.

4. Comparing obtained RBA and the stakeholder assessments. In this case, an additional input for a policy is an instance of validation influence model VIM which reflects e.g. tolerance intervals for assessment consistence checking, stakeholder priorities etc.

5. Making the decision based on the results of comparison. If the assessments are consistent (taking into account the specified tolerance interval) the RUV is accepted as valid, otherwise the validation process ends in failure. This consistence checking-based validation can be done by business stakeholders.

4.2. Requirement-centered validation policy

Another possible approach could be to convert stakeholder assessments into requirements via elicitation policy [8] and compare these elicited requirements to RUV (Fig.2). If the elicited requirement is consistent with the RUV (with

some tolerance interval) the validation of the latter can be considered successful. We call such approach *requirement-centered validation policy.*

An example of the question asked by this policy looks as follows: "can we say that the requirement that *the quality X should not exceed a threshold Y1* is valid if the stakeholders state that it should not exceed a threshold Y2".

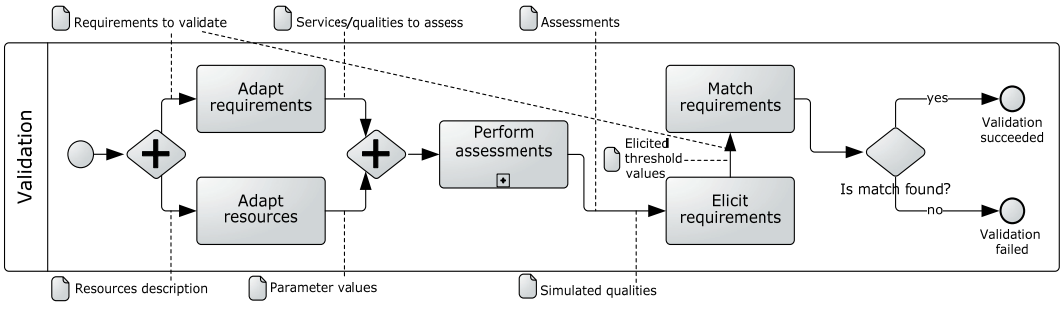

Fig. 2. Requirement-centered validation policy

The elicitation policy is defined as obtaining the required levels (threshold values) of service performance and reliability. An example of elicited requirement could be "the latency for the *service X* must not exceed the *threshold*"; the goal is to define the corresponding threshold values. We introduce the derivation mechanism which gets these values out of the captured output of the assessment mechanism (both simulated qualities and assessments). The additional input for this mechanism is an instance of a derivation influence model DIM containing such information as relative importance of stakeholders or contexts, degree of confidence for stakeholders etc., it should take into account possible imprecision of assessments.

A simplified example of using the elicitation policy is as follows. Suppose we defined the software architecture with the *CheckOrder* service and we are to get the threshold values for its performance and reliability from business stakeholders. To do this, we run IIA to produce stakeholder assessments of simulated qualities of this service in different usage contexts (e.g. for order confirmation and verification), different roles and stakeholder sessions such as *"the score for the latency of CheckOrder in order confirmation context produced by stakeholder X in a role of order clerk for the simulated value of 0,5 sec is "two times worse than necessary""*. The derivation mechanism collects these assessments and derives requirement thresholds out of them (if other assessments agreed with our case, this process could eventually lead to the latency requirement threshold of *0,25 sec*).

Requirement-centered validation policy consists of the following activities:

1. Deriving QAPM-S instance representing the set of services and qualities from RUV (as specified for the assessment-centered policy); it becomes an input for the elicitation policy together with the set of parameter values.

2. Running the elicitation policy (which, in turn, executes the assessment mechanism as defined in [5]) and deriving the set of requirements (*elicited threshold values*) from the stakeholder assessments. These values can be precise or imprecise depending on the allowed degree of uncertainty.

3. Comparing elicited threshold values and the ones specified as part of the RUV. An additional input is an instance of VIM which reflects e.g. tolerance intervals for requirements consistence checking, requirement priorities etc.

4. Making the decision based on the results of comparison. If the threshold values are consistent (taking into account the specified tolerance interval) the RUV is considered valid, otherwise the validation process ends in failure.

4.3. Comparing the proposed approaches

The proposed approaches differ in one important aspect: the subject performing a comparison between the artifacts originated from the results of assessment and the ones obtained from the RUV.

For assessment-centered policy, this comparison can be naturally performed directly by business stakeholders. The reason is that prior to this they work with quality assessments – not directly with requirements so they are more familiar with this kind of information. For requirement-centered policy, this comparison is more likely to be performed by requirement engineers. As a result, we can make a conclusion that in most cases assessment-centered policy will allow for more involvement of business stakeholders into software development process.

4.4. Combining requirements negotiation and requirements validation

Both above approaches are based on the assumption that the RUV cannot be adjusted to reach the compromise with stakeholder assessments. In this case, such requirements are simply rejected if they cannot be validated immediately.

To avoid this "yes-or-no" decision process, we need to extend the validation policy with negotiation mechanisms inspired to those we proposed in a special negotiation policy [7] aimed at reaching a compromise between the available resources and the stakeholder opinions represented by assessments. To reach this compromise, it applies systemwise optimization methodology [15]. Special multicriterial optimization problem has to be stated and solved to find the value of adjustment for either resource constraints (forming *feasible area* D_0 in a quality assessment space) or stakeholder expectations (forming *directive area* D^d in the same space) to make the desired point y^* (reflecting stakeholder assessments) feasible at the same time (initial feasible point is denoted as y_0).

In our case, the compromise between RUV, stakeholder assessments, and the available resources needs to be reached. To reflect this, we can split the directive area into two (possibly non-intersecting) subareas: one corresponding to RUV (*RUV area* D^{d1} with a *required point* y^{*1}) and another corresponding to the stakeholder assessments (assessment area D^{d2}, the desired point now is denoted as y^{*2}). Different statements of the systemwise optimization problem can be formulated to reflect different strategies for adjusting the respective subareas: it is possible to adjust RUV (altering D^{d1}), assessments (altering D^{d2}), or resource constraints (altering D_0). On Fig. 3, we show one possible example of stating this problem when it is possible to adjust the required area to move the required point closer to the desired area (the assessment area is omitted from the picture). For illustration, we follow [7] in using a trade-off diagram with a

two-dimensional slice of the quality assessment area (exemplifying software qualities by performance and reliability).

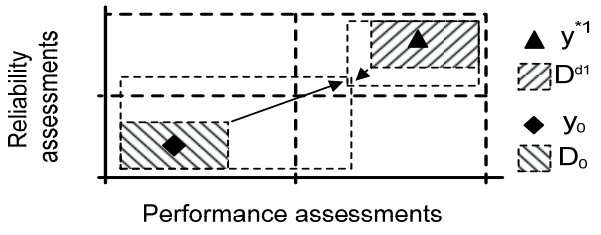

Fig. 3. Adjusting the RUV area

5. Conclusions and future work

In this paper, we defined the principles of new high-level procedures (policies) aimed at validating performance and reliability requirements by comparing them to the values obtained from the assessments of simulated software qualities. Their advantage as compared to known requirements validation methods is that stakeholders are able to experience the prospective system before expressing the opinions on its quality in a process of requirements validation. We also proposed an approach to adjusting the requirements under validation to reach a compromise with either stakeholder assessments or the available resources based on a methodology of systemwise optimization by V.M.Glushkov.

In future, we plan to elaborate models underlying the validation policy and procedures for its implementation, investigate the applicability of different methods for solving the multicriterial problem of finding the optimal adjustments for RUV or necessary resources, and establish the evaluation studies for the proposed solutions.

References

1. Uchitel S. Fluent-Based Animation: Exploiting the Relation between Goals and Scenarios for Requirements Validation / S. Uchitel, R. Chatley, J. Kramer, J. Magee // Proc. RE'04. – IEEE. – 2004. – P. 208-217.
2. Sukumaran S. A Rigorous Approach to Requirements Validation / S. Sukumaran, A. Sreenivas, R. Venkatesh // Proc. SEFM'06. – IEEE. – 2006. – P. 236-245.
3. Fliedl G. From textual scenarios to a conceptual schema / G. Fliedl, C. Kop, H.C. Mayr // Data Know. Eng. – 2005. – Vol. 55. – N 1. – P. 20-37.
4. Kaschek R. Towards Simulation-Based Quality Requirements Elicitation: A Position Paper / R. Kaschek, C. Kop, V.A. Shekhovtsov, H.C. Mayr // REFSQ 2008. – LNCS, Vol. 5025. – Springer. – 2008. – P. 135-140.
5. Shekhovtsov V.A. Interactive assessment of simulated service qualities by business stakeholders: principles and research issues / V.A. Shekhovtsov // Проблеми програмування. – 2010, in press.
6. Kroha P. Text Generation for Requirements Validation / P. Kroha, M. Rink // Proc. ICEIS 2009. – LNBIP, Vol. 24. – Springer. – 2009. – P. 467-478.
7. Shekhovtsov V.A. Towards negotiating QoS requirements originated from stakeholder assessments of simulated service qualities / V.A. Shekhovtsov // Радіоелектронні і комп'ютерні системи. – 2010. – N 1.
8. Shekhovtsov V.A. Towards eliciting QoS requirements from stakeholder assessments of simulated service qualities / V.A. Shekhovtsov // Proc. SAIT'2010. – Kyiv. – 2010, in press.
9. Shekhovtsov V.A. Relational service quality modeling / V.A. Shekhovtsov, R. Kaschek, C. Kop, H.C. Mayr // J.Suzuki (ed.) Developing Effective Service Oriented Architectures: Concepts and Applications in Service Level Agreements, Quality of Service and Reliability. – IGI Global. – 2010, in press.
10. Galle D. A Uniform Web Service Description Representation for Different Readers / D. Galle, C. Kop, H.C. Mayr // Proc. ICDS'08. – IEEE CS Press. – 2008. – P. 123-128.
11. Moreira A. Crosscutting quality attributes for requirements engineering / A. Moreira, J. Ara jo, I. Brito // SEKE'02. – 2002. – P. 167-174.
12. Mellor S.J. MDA Distilled: Principles of Model-Driven Architecture / S.J. Mellor, K. Scott, A. Uhl, D. Weise. – Addison-Wesley. – 2004. – 176 p.
13. Pastor O. Model-Driven Architecture in Practice / O. Pastor, J.C. Molina. – Springer. – 2007. – 302 p.
14. Fliedl G. Linguistically based requirements engineering - The NIBA-project / G. Fliedl, C. Kop, H.C. Mayr, et al. // Data Knowl. Eng. – 2000. – Vol. 35. – N 2. – P. 111-120.
15. Glushkov V. Systemwise optimization / V. Glushkov // Cybernetics and System Analysis. – 1980. – Vol. 16. – N 5. – P. 731-733.