УДК 681.03

INTERACTIVE ASSESSMENT OF SIMULATED SERVICE QUALITIES BY BUSINESS STAKEHOLDERS: PRINCIPLES AND RESEARCH ISSUES

Vladimir A. Shekhovtsov

National Technical University «Kharkiv Polytechnical Institute»
21 Frunze Str., Kharkiv, Ukraine 61002
Phone: +38057 7076474. Fax: +38057-7076520. Email: shekvl@yahoo.com

We present the principles of an approach supporting the stakeholder involvement in a software process for service-oriented systems in a form of assessing the perceived quality of the software under development in its usage context. This method relies on interactive simulation of service performance and reliability; simulation models are parameterized by the factors influencing service execution; business stakeholders experience simulated service qualities in simulated usage contexts and assess this experience; the obtained assessments can be later used throughout the system lifecycle as a means of control for the quality of the software under development.

Наведено принципи підходу, що підтримує участь зацікавлених осіб у процесі розробки сервіс-орієнтованих програмних систем у вигляді оцінювання сприйманої якості розроблюваної системи в контексті її використання. Цей підхід спирається на інтерактивне імітаційне моделювання продуктивності та надійності сервісів; параметрами імітаційних моделей є фактори, що впливають на виконання сервісів; зацікавлені особи висловлюють своє відношення до значень продуктивності та надійності, отриманих при взаємодії з імітаційними моделями якості сервісів у рамках виконання імітаційних моделей їх контекстів використання, надані оцінки можуть бути використані на різних етапах життєвого циклу програмного забезпечення як засоби контролю його якості.

1. Introduction

Collecting the opinions of business stakeholders is recognized in the current software engineering research and practice as an important part of the software process. For example, software engineering lifecycle standards such as ISO/IEC 12207 [35] emphasize the need of collecting such opinions in a form of *stakeholder requirements* by establishing special *Stakeholder Requirements Definition Process* to deal with them. This is particularly true for service-oriented software systems as developing software services require knowledge of their possible uses which is difficult to obtain without the involvement of their prospective users [25].

Currently in most cases business stakeholders are involved in the development process to express their opinions of the prospective system's functionality [53]. It is not the only possible form of involvement, however: another kind is the involvement through *assessments of the quality of the prospective system*.

A motivating example is as follows. Suppose the software under development (SUD) is a service-oriented system intended for business stakeholders. If they will not have a chance to present their opinions regarding the desired quality of the prospective system early on its development lifecycle these opinions could be easily overlooked and lost. As a result, the understanding of the desired quality of the system becomes biased towards the view of the IT people: "the inmates are running the asylum" [10]. This could lead to the stakeholder dissatisfaction with SUD quality late in the development lifecycle, the loss of confidence in the developers, and the failure of the project [61].

The current problem is that such assessment and its use in the software process in many domains still remains a poorly investigated, difficult and error-prone task. Several arguments are in favor of this claim:

1. It is not realistic to expect that the business stakeholders can be fully involved in a software development process if they cannot experience the SUD. Without such experience, they are forced to be speculative in their opinions e.g. by formulating narrative statements such as "the system should be reliable" and "the system should have good performance under any load". Such speculations are not well suited to be a means of control for the quality of the SUD.
2. The process of assessing the SUD quality depends on the anticipated or implemented interaction of the SUD with its environment. This assessment obviously may be a complex task especially for service-oriented systems. It is difficult for stakeholders to invent the interaction scenarios "on-the-fly" without appropriate support.

We propose a method motivated by the above problem. It is based on the investigation of the ways to support the stakeholder involvement in a form of assessing the perceived performance and reliability of the service-oriented SUD in its usage context. To implement this support, we make service quality assessment mechanisms accessible to the stakeholders without IT background. On top of these mechanisms, we establish higher-level policies solving particular requirements engineering and architectural design problems (requirements elicitation, verification, architecture assessment etc.). In this paper, we describe the concepts of the method and the proposed assessment mechanisms.

2. State of the art

The importance of the problem of involving business stakeholders in the development process as a means of control for the quality of the produced artifacts was supported by its extensive discussion in the scientific literature. In this section, we review several categories of methods addressing this problem mostly following the classification of the methods to represent the quality of the prospective system proposed by Bosch [7], which includes scenario-based techniques, prototyping, and simulation.

Human interaction with stakeholders. Significant number of methods addresses the problem directly by performing human interaction with stakeholders and collecting their opinions.

Request-centered techniques. These techniques emphasize the way of questioning the stakeholders and processing their opinions. Social research interaction techniques [15] such as surveys, interviews, brainstorming, questionnaires or checklists are used to solve this problem [45, 53] together with software engineering-specific techniques such as CRC cards [4] or user-centered tabular glossaries [22]. The problem with these approaches is that many of the stakeholders neither are used to nor trained in reasoning about the system in general and its qualities in particular without having working experience regarding the targeted SUD.

Scenario-centered techniques. Such techniques [9] organize scenarios of stakeholder interaction with a prospective system; they often rely on request-centered techniques to process the opinions of stakeholders. In the manual scenario-centered approach, the stakeholders are requested to go through the scenarios (without actual interacting with the SUD or its executable model) and express their opinions. The techniques of this kind are extensively used to evaluate software architectures (examples are such methods as ATAM [41] and PASE [86]) or elicit or analyze quality requirements (examples are Quality Attribute Workshops [2] and SRA tool [27]). Manual scenarios are best suited for arranging the assessment of qualities which are revealed through complicated interactions (e.g. security [71]) or development-related quality attributes such as modifiability [44].

There are shortcomings of these techniques as a means of addressing our problem: (1) they do not allow stakeholders to experience quality with their senses in natural way (scenarios are usually narrative and do not offer reality-like experience); (2) manual scenarios do not allow studying the dependency between performance-influencing factors and the observed performance levels [7]; (3) for reliability, the scenarios in most cases cannot replace interaction with the real system or its executable model as they are able to express it only by example [1].

Traditional prototyping. These techniques use system prototypes as an aid for stakeholders. The notion of traditional prototype refers to a scaled-down version of the final system running in its intended context [19].

Horizontal prototypes. Early attempts to involve stakeholders in a development process using exemplification under the title of *rapid prototyping* were made in the 80ies [51]. In particular, horizontal prototypes (lacking implemented functionality) were introduced to simulate user interfaces in order to allow stakeholders to experience their future environment. Such prototypes are still widely used, their usage is often embedded in the system usage scenarios [72, 75]. The shortcomings of horizontal prototyping as an approach to address our problem are as follows: (1) it cannot make stakeholders assess measurable software qualities as there is nothing to perceive as measurable in the prototyping interactions; obtained information is limited to narrative notes; (2) such prototypes are mostly limited to making stakeholders look at the system via its prospective user interface; this is of less value for service-oriented systems as the client user interface can be not readily known at the time of development; (3) they are not well suited to taking into account factors influencing system execution and experimenting with different sets of values for such factors.

Vertical prototypes. Such prototypes implement some part of the system functionality in deeper (up to complete) detail. They can be used to make stakeholders experience operational software qualities (such as performance) and assess such qualities. They are also not very suitable for our problem due to the following limitations (discussed in e.g. [7]): (1) creating and maintaining such prototypes requires programming skills; (2) running prototype realistically requires access to the target system including its hardware and human users, otherwise the stakeholder experience will differ from the reality; (3) it is difficult to experiment with the vertical prototype trying out the different sets of factors influencing software qualities (doing “what-if” analysis) as such experiments are usually too costly and cannot be performed by non-programmers; (4) the solutions elaborated for such prototypes cannot be easily reused; (5) it is not possible to “compress” or “expand” the time during the execution of prototype; it complicates realistic exposing of the system reliability as it usually requires running in compressed time to reveal the relevant quality values.

«Live» prototyping. Several research-based [29, 39] and industrial (iRise studio, Axure RP etc.; the survey is in [54]) tools extend the rapid prototyping technique by allowing non-programmers to build and execute interactive software imitations (called “simulations” in their documentation) to get stakeholder’s feedback.

By simulation, their authors mean an imitation of system behaviour visible through its user interface. To implement such imitation, they provide interactive user interface animations (similar to those used by rapid prototyping) driven by the proposed scenarios of SUD behaviour. The only difference from the traditional rapid prototyping is that these animations are “live”: the imitated program displays an anticipated user interface allowing stakeholders to play with its visible elements making it show other elements according to the predefined navigational scenario, formulate requirements while looking at this interface, and attach them to its visible elements. As with rapid prototyping, the elicited information is limited to narrative notes. Underlying the animation is an interpreter of the language for specifying scenarios (either proprietary [54] or based on standard notation such as statecharts [29, 39]).

These tools are not directly suitable for our research problem because they implement different type of stakeholder involvement. Their purpose is to make stakeholders think about the system functionality while looking at some version of the behaviour of its user interface. To solve our research problem, however, it is necessary to make stakeholders think about the system quality while perceiving the *numerical values* of its attributes. This reflects the fundamental difference between functional and quality requirements: whereas the functional requirements are specifications of the anticipated systems behaviour (mostly in the narrative form as e.g. in the use case specifications), the quality requirements are qualities that the product must have accompanied with quantified criteria for measuring those qualities: “If you cannot quantify and measure a [non-functional] requirement, it is not really a requirement” [61].

Another problem with these tools is that they are rather inflexible. As their output is limited to textual (free-form) requirements their applicability is limited to the problem of requirements elicitation. The problem of stakeholder involvement in a software process is wider than that: it is desirable to be able to involve business stakeholders into more extended set of activities, such as requirements verification, requirements negotiation or software architecture evaluation. These tools also do not target service-oriented systems.

Simulation-based techniques. These techniques use simulation in a sense of "the process of designing and creating a computerized model of a real or proposed system for the purpose of conducting numerical experiments" [42] to model service qualities in a way that can be used to support stakeholder involvement.

Service-level simulations. Some solutions simulate qualities of the particular software services or their underlying components. Performance prototyping technique [31] allows performance simulations for the components to be integrated into real-life environment as alternative for prototypes, other service performance solutions include MOSES approach [12] based on UML 2 service representation, [16] implementing context-dependent discrete-event service performance simulation, [3] using two-level representation of service performance. Service reliability simulations are proposed in [24, 28], service error distribution is also simulated in [48]. The results of such simulations are used to guide the evaluation of the software architecture [7, 26], requirements validation etc. The specific problem with such simulations is that they are supposed to be used in standalone mode without explicit integration into the usage processes.

Process-level simulations. Such solutions support quality simulations in context of usage scenarios (in a sense of scenario-centered techniques) or complete business processes. Simulated scenarios [18, 30] are elaborated by analysts to reveal particular quality characteristics. Business process simulation solutions are numerous (e.g. [36, 64]) but not much of them produce numerical quality values. Among these solutions, UML-Ψ tool [50] simulates performance of the processes described by annotated UML diagrams, [20, 21] augments BPM process definitions (expressed in e.g. BPMN [8]) with information necessary to simulate performance, [73] enhances BPEL [84] programs with Java language blocks implementing performance and reliability simulation. The specific problem with such simulations is that they are aimed at process-aware applications [17] executed by BPM engines (where the process is a part of the SUD), so they treat services as "black boxes" lacking detailed control on their models. As a result, they are less suited to our problem where processes provide the usage context for the SUD treated as a set of services. Good idea would be to combine service-level and process-level simulation models into a coherent model being able at the same time to address them separately.

There are also common problems with the current simulation solutions: (1) they, as a rule, are designed for use by the persons with the IT background – not by business stakeholders; (2) many of them are not interactive [81]: the results are available after the run is finished; there are, however, approaches to *running* business processes interactively mostly based on task modeling [76]; (3) they mostly address particular quality attributes without paying attention to the inter-attribute tradeoffs.

3. Problem statement

Investigating the state of the art in the area of stakeholder participation in software process activities reveals the research problems to be addressed by the method. We describe these problems in logical sequence by formulating the corresponding research questions.

The overall research question that motivates the development of the method is: *How to involve business stakeholders into the software development process as a means of control for the quality of the produced artifacts?* As the concept of software quality is broad, we plan to focus on two specific quality characteristics: *performance* (treated as service latency) and *reliability*. These operational quantitative characteristics are well suited for representation by simulation and the number of contexts for their assessment is more manageable in comparison with e.g. security (which requires extensive set of complicated scenarios to be assessed in full). We also focus on *service-oriented systems* as the specific category of software under development. As a result, the main research question this method trying to answer is ***(RQ)*** *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?*

As an idea of addressing this research question, we propose to use *interactive simulation of service performance and reliability* possessing the following features:

1. Service performance and reliability simulations are parameterized by the factors influencing execution of service-oriented systems and executed interactively in the context of system usage;
2. Business stakeholders experience simulated service performance and reliability in simulated usage contexts corresponding to their roles and assess this experience using specific scale;
3. The obtained assessments are used as a basis for software lifecycle activities such as requirements elicitation, verification, and negotiation and as a means of control for software lifecycle decisions.

The reasons of selecting the simulation as an approach to the stated problem are as follows. It is free from the problems of horizontal or "live" prototyping as it can really offer the possibility to model the measurable (quantitative) quality attributes of the system under development in its environment. It is also free from the shortcomings of vertical prototyping as [55]: (1) simulations can be made configurable by non-programmers; (2) no access to the target system is required; (3) experimenting with simulation (trying different values for the factors influencing qualities of the target system) is easier and less expensive (in some situations separate prototype can be necessary for every major configuration); (4) simulation solutions can be reused as the same model can be instantiated to simulate different usage contexts; (5) software reliability can be simulated; (6) simulation offers the control over the model time: it is possible to compress or expand it. On the other hand, known problems with simulation solutions (outlined above) need to be addressed.

After establishing the basic idea of a solution, we formulate more specific research questions. We start from the basic notion of service quality as we need to establish its conceptual representation (covering both performance and reliability) for the purpose of the project. The corresponding question is: *(**RQ-1**) How to conceptualize service performance and reliability in a way most suitable for their assessment by business stakeholders?*

Prior to addressing the simulation, we plan to investigate the ways of assessment of quantified service qualities by business stakeholders. The corresponding question is: *(**RQ-2**) How to establish the interaction procedures allowing business stakeholder to experience and assess quantified service performance and reliability?*

To address this question, we need to obtain the knowledge of the current practice in this field so we formulate more specific research question: *(**RQ-2.1**) What is the current practice for business stakeholders to perceive and assess service performance and reliability?* After gathering this knowledge, we need to formalize it in a way suitable for reuse by addressing another more specific research question: *(**RQ-2.2**) How to formalize and reuse common practices of such assessment?*

The reason of conducting this research prior to investigating simulations is that stakeholder perception of qualities does not depend on their origin so we can use "mock-up" quality values before simulations are available. On the other hand, while working on simulations, it is desirable to be able to test them via assessment interactions.

Main research question related to the simulation of service performance and reliability is *(**RQ-3**) How to simulate service performance and reliability in a way compatible with common practices of their assessment by business stakeholders?* To address this question, we need to understand the factors that influence real-world service performance and reliability and should be taken into account in simulations (such as hardware capabilities, network bandwidth, peak user load etc); this leads to the specific research questions *(**RQ-3.1**) Which factors influence service performance and reliability?* After that, we need to make use of the gained knowledge in simulation-related context by addressing the question: *(**RQ-3.2**) How to make simulations of service qualities depend on these factors?* After we understand the influencing factors the next step is to develop the formal simulation models for performance and reliability in a way suitable for reuse; this leads to the specific research question *(**RQ-3.3**) How to formalize and reuse simulation solutions for service performance and reliability?*

After establishing approaches for simulating and assessing separate service qualities we plan to establish integrated service-level quality assessment mechanisms by addressing the research question *(**RQ-4**) How to combine service performance and reliability simulation solutions with the corresponding assessment procedures to establish service-level mechanisms for interactive assessment of simulated service qualities?*

After establishing service-level mechanisms, we plan to investigate the idea of providing them to business stakeholders in simulated usage contexts; this leads to the research question *(**RQ-5**) How to integrate simulation and assessment of service qualities into service usage contexts?* To make stakeholders experience usage contexts we plan to simulate business processes providing these contexts; this leads to the research question *(**RQ-5.1**) How to simulate service usage processes with integrated service-level simulation and assessment mechanisms?*

Addressing the process-level research questions leads to elaborating mechanisms defined at the level of the particular simulation session where a particular stakeholder in a particular role participates in a particular simulated usage process. To reveal all the stakeholder opinions, the final assessment mechanism needs to gather assessment data out of all the necessary usage processes, roles and stakeholder sessions; we plan to investigate the idea of such mechanism by addressing the research question *(**RQ-6**) How to make mechanism of process-driven assessment of simulated service qualities gather all necessary assessments?*

4. Description of the proposed approach

In this section, we describe in detail the proposed approach for answering the stated research questions. The preliminary research leading to this approach was presented in [40, 69].

General methodology issues. The research is based on model-driven [33, 58] and ontology-based [32] software engineering methodologies. Ontologies are used to describe the knowledge resulting from the investigation of the problem space. Mechanisms addressing research questions in a solution space have underlying prescriptive models; every model is defined by a metamodel. To elaborate this set of basic notions, conceptual modeling and system analysis methodologies need to be applied; the treatment of system analysis follows the approach of M.Z.Zgurovsky and N.D.Pankratova [88]. For dissemination, metamodels and ontologies need to be expressed using both practice-oriented (e.g. UML, OWL) and formal (e.g. set theory, description logic) notations. As a result, a complete formal description of the approach has to be obtained.

Modeling service quality attributes. To address the research question *RQ-1* we provide the definition of quality for proposed method and the means of conceptualizing the service quality attributes. We use ***Q***uality-***A***ware ***P***redesign ***M***odel for ***S***ervices *QAPM-S* [70] for this purpose (extending it to satisfy the needs of the method).

QAPM-S 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) [52]; it also uses tabular model representation (called glossary) which is well understood by stakeholders [22]. It models service quality as a hierarchy of quality characteristics [38], represents the facts that quality characteristics influence each other and that stakeholders perceive qualities differently, follows aspect-oriented paradigm [56] in representing service quality and functionality as separate concerns.

Interactive stakeholder assessment of quantified qualities. This stage of research addresses the question *RQ-2*. The relevant research has to be conducted in two steps corresponding to *RQ-2.1* and *RQ-2.2*: gathering knowledge related to the stakeholder perception of qualities and developing formal interaction models of quality assessments.

Stakeholder perception of quantified qualities. To address the question *RQ-2.1*, it is necessary to investigate the principles of stakeholder perception of performance and reliability and the relevant representation/assessment scenarios. For this purpose, it is necessary to establish field studies involving, in particular, interviews and surveys, working with paper and user interface prototypes. The methodologies used in these studies are information perception and visualization [14, 83], usability engineering and human-computer interaction applied to software engineering [68, 78], social scaling [13, 15] etc. The results should describe the relevant knowledge by means of the *ontology of stakeholder quality perception*. It organizes gained knowledge and can be used for elaborating interaction procedures. This knowledge can be useful by itself to better understand people's needs with respect to the assessments of numerical software qualities.

Interaction models for quality assessments. Based on this knowledge, we can address the question *RQ-2.2* by formalizing the ways of making stakeholders experience and assess numerical quality values by means of process models for quality visualization and end-user interactive assessment. For every quality characteristic the corresponding ***i***nteractive ***a***ssessment ***m***odel IAM needs to be elaborated. It describes the *IAE mechanism* which:

1. receives the numerical quality value as an input (the mechanism does not depend on its source);
2. makes the stakeholder experience this quality according to this value: (a) visually e.g. by displaying the control board with a number of visual failure alerts to represent particular level of reliability; (b) by other means e.g. by making the stakeholder wait for the time related to the particular latency;
3. involves the stakeholder in an interaction related to an assessment of his/her experience; in this case: (a) appropriate research will be performed to choose scales for these assessments [13, 15]; (b) trade-off dependencies among qualities can be used as assessment hints, e.g. the request for the latency assessment could include the hint that improving the latency negatively affects reliability;
4. receives the assessment value from the stakeholder and returns this value as an output.

We aim at facilitating reuse of such models by establishing the IAML library storing IAMs for different quality characteristics. The modeling techniques to be used for these models include conceptual user interface modeling [58] and model-based user interface design (MB-UID) [59].

Parameter-dependent simulation of particular service qualities. We address the research question *RQ-3* by establishing the procedures for parameterized simulation of service qualities. These procedures are responsible for composition and execution of simulation models.

Simulation parameters. To address the question *RQ-3.1* it is necessary to investigate the factors influencing service qualities (we define the notion of *simulation parameters* for these factors). As the parameters are related to the real-world problem being simulated, we establish the *parameter ontology*, where all the knowledge about the factors influencing service qualities is collected (e.g. their types, interdependencies, character of the influence on the model etc.). This ontology needs to be consulted while developing parameter-related simulation modules following ontology-based simulation modeling technique [5]. To address the question *RQ-3.2*, it is necessary to create a *parameter slot* in a simulation model for every necessary parameter; prior to the execution of this model, the *parameter value* is expected to be specified for every slot; these values influence the execution. It is also necessary to take into account possible fuzziness of the parameter values to reflect incomplete knowledge of the factors.

Main simulation structure. The implementation of this structure follows the research on simulation modeling of service-oriented architectures [3, 74, 77] in that we start from the simulation conceptual model [62] for this structure with entities representing different parts of the SOA infrastructure (processing and storage hardware, network and software infrastructure). We consider agent-oriented simulation technique [57] and its applications to SOA simulation [74] as a means of instantiating this model as the set of communicating agents representing the above entities. Another research alternative is discrete-event simulation ([42], its application to SOA is in e.g. [16]); these two techniques could also complement each other. These approaches are better suited for modeling software systems than continuous techniques such as system dynamics [49].

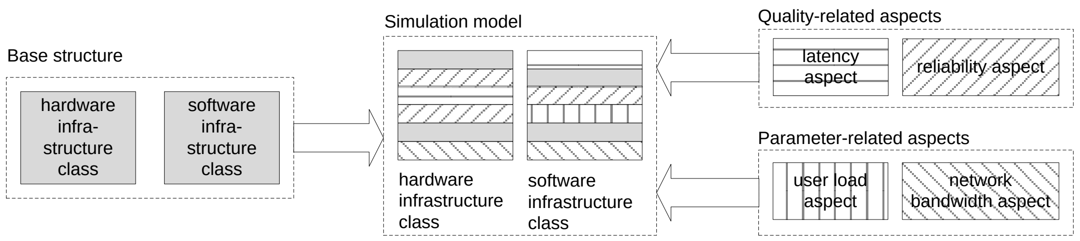
Composing the simulation models. Simulating particular service quality characteristic requires the particular set of solutions to be introduced into the simulation model. On the other hand, making simulations depend on the particular parameter also requires specific set of solutions to be implemented. To address research question *RQ-3.3*, for every selected quality characteristics and parameter, we elaborate the set of simulation model solutions best suited for its support and package it into the *simulation module* (obtaining the set of quality-related and parameter-related modules).

Separate research directions need to be pursued to establish simulation models for performance and reliability. For *performance*, it is necessary to make the module implement the set of modeling procedures implementing actions specific to service performance [6, 74, 89]. They should rely on a formal representation of performance (we plan to investigate Colored Petri Nets [85] or their extension of Queuing Petri Nets [43] for this purpose) and utilize software performance ontology (e.g. [46]). For *reliability*, the module needs to implement the set of fault injection procedures [48] (taking into account error propagation [60] and other reliability-influencing issues) depending on selected fault and failure models [11]; these procedures should rely on service dependability ontology [47].

After the simulation modules are available, we can develop the rules managing their composition to form service quality simulations. Implementing this composition can be complicated as the particular parameter could influence the simulation of several qualities whereas the particular quality simulation could depend on several parameters. In addition, dependencies inside the sets of qualities and parameters as expressed in QAPM-S and parameter ontology need to be considered as well (for example, simulating the reliability can depend on the simulated latency).

Such complicated *crosscutting* relationships among the modules could be the case for the separation of concerns inside the simulation conceptual model according to the aspect-oriented paradigm (*aspect-oriented simulation modeling* - AOSM). This way, both support for the quality characteristic and parameter dependency can be viewed as separate simulation concerns. The corresponding quality-related and parameter-related modules can be independently developed as *aspects* representing the separated concerns (latency simulation aspect, reliability simulation aspect, network bandwidth parameterization aspect etc.). For example, fault injection procedures implemented by reliability simulation aspect could contain "hooks" for possible extensions by other aspects without revealing any information about the character of such aspects (following the obliviousness principle of aspect-orientation).

We choose to follow the asymmetric aspect-oriented modeling approach [67] where the aspects crosscut the main simulation structure which is preserved as the structure of the resulting simulation. The weaving of the simulation aspects into this structure is shown on Fig.1. For example, the composition rule could connect fault injection procedures implemented by the reliability aspect to particular external variables defined by aspects responsible for user load and latency (populating the "callback" hooks defined for all these three aspects) and then weave the resulting load-dependent and latency-dependent fault injection procedures into the simulation models for the hardware and software infrastructure. The application of the aspect-oriented approach to the service quality modeling is novel; existing AOSM techniques are specific to other domains [34, 79].



Base structure
hardware infra-structure class
software infra-structure class
Simulation model
hardware infrastructure class
software infrastructure class
Quality-related aspects
latency aspect
reliability aspect
Parameter-related aspects
user load aspect
network bandwidth aspect

Fig. 1. Weaving the simulation model

This investigation follows the research on assessment interaction models, so we can immediately test simulation results via the assessment interfaces. During the execution, every service quality simulation accepts the set of values for all the parameters it depends on, and produces the value (or the set of values) for the simulated quality, so we just replace mock-up quality sources with simulation modules.

To facilitate reuse of simulation solutions according to the research question *RQ-3.3* we aim at creating QAL and PAL aspect libraries storing simulation aspects corresponding to selected qualities and parameters and the library of infrastructure simulation solutions (ISL). These *simulation support libraries* are supposed to be used for implementing simulation of the complete services (described below). To validate the results of the research at this stage (i.e. simulation models), we follow the research on simulation models verification and validation (e.g. [66]).

To implement proof-of-concept software support for quality simulations and for service-level execution mechanisms, we use Java API for the state-of-the art simulation tools (AnyLogic (*http://www.xjtek.com*) being the main choice and OMNet++ Java extension (*http://www.omnetpp.org*) the main alternative).

Standalone assessment of simulated service qualities. We address research question *RQ-4* by elaborating *IAS mechanisms* (***i**nteractive **a**ssessment of **s**ervices*) aimed at an assessment of simulated service qualities at the level of the *particular service*. Fig.2 depicts the outline of IASC (composition) and IASE (execution) mechanisms.

Composing the service simulation. IASC inputs include QAPM-S representation of the set of *qualities of interest* to be simulated and assessed (it can be a subset of the available qualities) and the set of *necessary parameters*. To get the integrated ***q**uality **s**imulation **m**odel QSM*, we perform a composition of the simulation aspects corresponding to the qualities of interest and the necessary parameters on top of the base simulation structure (all obtained from simulation support libraries). The parameter slots are created for all the necessary parameters. Also, we select the set of interactive assessment models from IAML for the qualities of interest and integrate them with QSM. The resulting service-level simulation and assessment model *IASM* becomes the IASC output. At this stage of the project, it is transferred to IASE for standalone execution (as shown on Fig. 2).

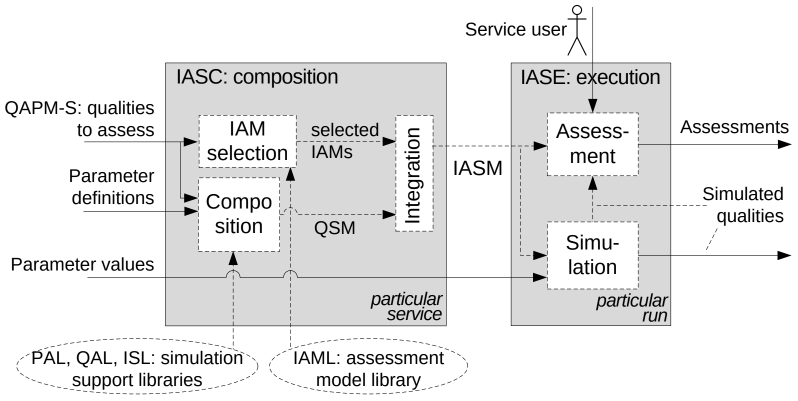


Fig. 2. Mechanisms for interactive assessment of simulated service qualities

Executing and assessing the service simulation. 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 IASM parameter slots. 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 IAE mechanisms described by interaction models integrated into IASM. The IASE outputs are this set of simulated qualities and the set of assessment results.

Service-level assessment example. Suppose we develop the CheckOrder service and plan to make users as sess its latency and reliability which depend on network bandwidth and user load. In this case IASC inputs are the qualities of *latency* and *reliability* and the parameters of *bandwidth* and *load*. The resulting model IASM(CheckOrder) is composed from simulation model QSM(CheckOrder) (based on simulation aspects for latency, reliability, bandwidth and load) and the interaction models for latency and reliability. IASE accepts IASM as the model to execute and, for its run, obtains the parameter values for bandwidth and load. The outputs for the run are the values for simulated latency (e.g. 0,5 sec) and reliability (e.g. 99.4%) and their assessments (e.g. the latency score of 8 out of 10).

IASC and IASE can be useful by themselves if the analyst just wants to ask stakeholders to assess the qualities of the particular service or the list of services without burdening oneself with establishing detailed usage processes.

Process-driven assessment of simulated service qualities. We address research question *RQ-5* by elaborating *IAP mechanisms* (***i***nteractive ***a***ssessment of ***p***rocesses) aiming at interactive assessment of simulated service qualities in context of usage processes at the level of the *particular process*. Fig.3 depicts the outline of IAPC (composition) and IAPE (execution) mechanisms. They rely on IASC and IASE dealing with individual services.

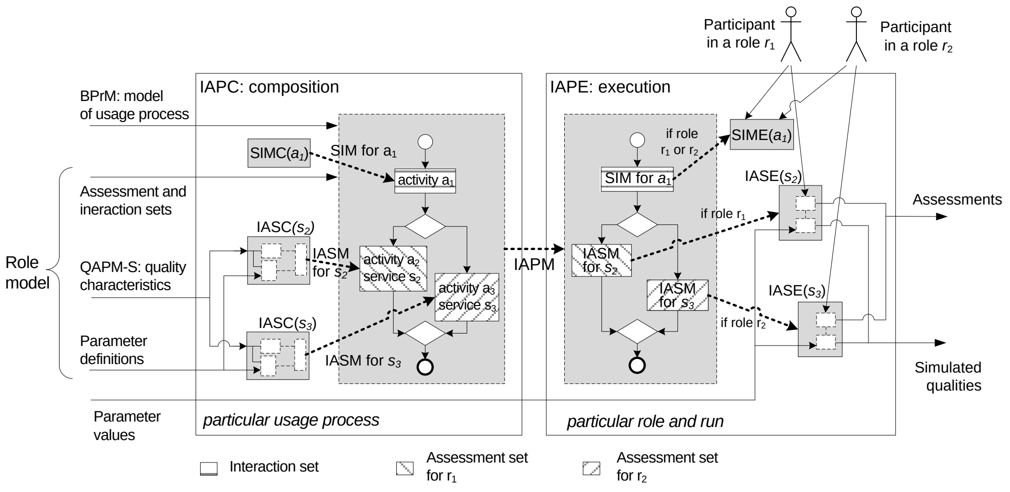


Fig. 3. Interactive assessment of simulated service qualities in context of usage processes

Composing the process simulation. IAPC forms the simulation model of the usage process making it ready for interactive assessment of service qualities. It accepts the following inputs:

1. The *control flow model* (CFM) for the usage process conforming to the network BPM notation such as BPMN, process chains [80], Colored Petri Nets etc. (in the scope of this project we restrict the accepted notations to those supported by the chosen BPM simulation tool). It is possible to use existing process model repositories such as APROMORE [63] for storing and retrieving such models.
2. The *role model* for the usage process which includes: (a) the set of roles defined for process participants (clerk, manager etc); (b) different sets of *interaction activities* for different roles; such activities make participants affect the state of the process simulation (e.g. by selecting execution paths, specifying values to be used for subsequent decisions etc); these interactions advance the model time; (c) different sets of *assessment activities* for different roles; such activities correspond to the *services of interest* to be simulated and assessed by stakeholders; assessments do not change the state of the process simulation (e.g. the model time has to remain the same as the simulation needs to be "paused" while the stakeholder thinks over the assessment); (d) the sets of qualities of interest and necessary parameters defined for every service of interest.
3. The *flow simulation model* (FSM) including all the extensions necessary to implement the process simulation beyond those presented in a role model (according to e.g. the techniques presented in [65, 82]); to simplify this model, all except interaction and assessment activities are treated as black boxes.

While composing the integrated model IAPM for the process, IASC mechanism is invoked for every service of interest. It creates the IASM model for this service which is then integrated into IAPM (this is the case for a_2 and a_3 activities on Fig.4). For every interaction activity, *SIMC mechanism* for constructing the interaction model is invoked and the resulting *SIM model* (depending on the type of activity) is also integrated into IAPM (see the activity a_1 on Fig. 4). SIMC prototype handling several basic interaction types will be established alongside IAPC.

The resulting model contains (1) the simulation logic defined by CFM and FSM (for the process) and simulation submodels of different IASM models (for the services of interest); (2) the assessment logic defined by interaction submodels of these models; (3) the interaction logic defined by the SIM models for all interaction activities.

Executing and assessing the process simulation. The IAPM is executed by IAPE. Every run is presented to the stakeholder belonging to the particular role (on Fig.3 two stakeholders of the roles r_1 and r_2 are shown interacting with IAPE; in the scope of the project, we restrict ourselves with a single-user mode when every run involves exactly one stakeholder; the multi-user mode can be investigated as a project extension). The input for every run of IAPE is the set of parameter values for all the parameter slots defined for the services of interest.

During the run, the following actions are performed:

1. The basic simulation flow is managed by the model derived from the usage process CFM;
2. 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 (on Fig.3, this is the case for a_2 and a_3 representing s_2 and s_3). IASE inputs are parameter values for all the slots of this service.
3. When this logic requires interacting with the simulation, the logic of this interaction is handled by the *SIME mechanism* invoked for the corresponding SIM (this is the case for a_1 on Fig.3). SIME prototype performing several basic interactions is expected to be elaborated alongside IAPE; we plan it to utilize available interaction modeling techniques similar to those used for IAM.

The selection of actions could be different for different roles (on Fig.3, role r_1 interacts with activities a_1 and a_2, whereas role r_2 interacts with a_1 and a_3). 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.

Process-level assessment example. Suppose the task is to make order and bookkeeping clerks assess the services of the order processing system in the context of order processing. In the CFM for the ProcessOrder usage process CheckOrder activity (with performance as the quality of interest) in accessible to order clerks whereas CheckPayment activity (for it, we are interested in reliability) is accessible to bookkeeping clerks; both these activities are backed by services of interest. These activities are assessment activities for the respective roles. On the other hand, AskForCanceling activity accessible to order clerks is not backed by a service and serves only to drive the simulation; it is an interaction activity for this role. IAPC forms the IAPM by combining CFM(ProcessOrder) and FSM(ProcessOrder) with IASM(CheckOrder), IASM(CheckPayment) and SIM(AskForCanceling) and transfers it to IAPE for execution. When the simulation logic for the *order clerk*-driven run passes through CheckOrder, IASE executes the corresponding IASM and obtains the performance assessment from the clerk as described in an example for this mechanism; the same happens to reliability of CheckPayment during the run driven by *bookkeeping clerk*. In case of passing through AskForCanceling its SIME asks the order clerk to decide what to do next.

Implementing the process simulation. We address research question *RQ-5.1* by investigating an approach to simulation which can be implemented by following existing simulation solutions defined for network-based BPM (such as BPMN-supporting OXProS [23] or CPN Tools [37]). We propose to evaluate several possible ways to implement this integration. One possibility is to implement solution in spirit of [3] which uses an implementation of process chains for the control flow simulation and OMNet++ code to implement service simulations with a resulting model deployed on OMNet++ engine. This way, IAPE could run process chains simulation code while IASE – service simulation code (both backed by OMNet++ or AnyLogic). Another approach is to use an idea of [23] which performs template-based conversion of BPMN diagram into Colored Petri Nets (CPN), this way, BPMN-based CFM can be converted into CPN representation, augmented with Java-based service simulation code and run via CPN tools or OXProS engine.

Iterative assessment mechanism. We address research question *RQ-6* with the iterative *IIA mechanism* (***i****terative* ***i****nteractive* ***a****ssessment*) which explores different variants of usage processes, roles and runs. The invariants for all iterations are the set of services under development (corresponding to the chosen variant of SOA) and the QAPM-S instance defined for this set. The iterations are as follows: the outer (composition) loop is *by process* (selecting different usage processes for the services and composing simulation models with IAPC), the inner (execution) loops are *by role* (selecting different roles and, as a result, exploring different subsets of services accessible for the role) and then *by simulation run* (making different runs for different users acting in the chosen role). The results are obtained on the every iteration of the inner loop. As a result, for every service of interest, we obtain simulated quality values and their assessments in all its key usage contexts.

Cost management. Modular structure of the solutions allows for flexible management of the implementation and deployment costs as different level of detail can be selected depending on the permissible cost. *At the service level*, the cheapest option is to build simulation models out of the predefined aspects and interaction models according to the selected set of qualities and influencing factors (the configuration is reduced to specifying qualities and factors of interest). If the cost is not an issue, however, it is possible to prepare custom quality simulations for every service of interest or combine predefined and custom simulation aspects. *At the process level*, the configurations vary from making stakeholders work directly with services, to introducing simple "task list" interfaces for service simulations, then to relying to available predefined processes (e.g. from APROMORE repository [63]) and to building custom processes for the problem at hand. This flexibility offers cost management advantages compared to other solutions. For example even the most complete process-based solutions such as [20, 21] are built in top-down fashion, as a result, complete process model needs to be built in any case; other techniques are even less flexible. Another kind of cost advantages are related to the choice of simulation over other quality modeling techniques such as prototyping [7, 55].

5. Conclusions and future research

In this paper, we elaborated the concepts for the mechanisms of tool-supported stakeholder assessment of simulated service performance and reliability in service usage contexts represented by simulated business process models. We presented service-level and process-level mechanisms, examples of their usage, and possible directions of their implementation. The development of these mechanisms should rely on research in such fields as human-computer interaction, social sciences, information visualization, discrete and agent-based simulation of performance and reliability.

Novelty of the proposed approach. After comparison with state-of-the-art stakeholder involvement techniques we state that the proposed mechanisms possess the unique combination of features:

1. They rely on simulation models able to express quantified quality attributes; to form service simulations, such models are integrated using novel aspect-oriented approach;
2. They allow interactive participation in simulations by business stakeholders; such participation is based on formal interaction models elaborated as a result of studies of real interactions in field settings;
3. They integrate the simulations of service qualities (in particular, performance and reliability) into the simulations of service usage processes and make these simulations accessible to stakeholders acting in particular roles so that a set of services available for assessment depends on the role.

Future research directions. In this paper, we limited ourselves to the description of the assessment mechanisms. These mechanisms form the foundations for the higher-level policies intended to address particular problems of requirements engineering and architectural design. In particular, requirements elicitation policy to get the threshold values for quality requirements out of stakeholder assessments and simulated quality values, requirements verification policy to compare external requirements with requirements elicited via assessment mechanisms, requirements negotiation policy inspired by systemwise optimization by V.M.Glushkov [87] to mutually adjust the resources of the organization and the needs of stakeholders. These policies will be the subject of future research.

The ultimate future goal of the proposed method is to establish a *lifecycle simulation support* by asking the stakeholders to make assessments of simulated qualities reflecting the current state of the software under development as the development progresses with a purpose to make these assessments drive the development.

1. *Amyot, D., Eberlein, A.* An Evaluation of Scenario Notations and Construction Approaches for Telecommunication Systems Development // Telecommunication Systems. – 2003. – Vol. 24. – N 1. – P. 61-94.
2. *Barbacci, M., Ellison, R., Lattanze, A., et al.* Quality Attribute Workshops (QAWs), Third Edition. – Carnegie Mellon University. – 2003.
3. *Bause, F., Buchholz, P., Kriege, J., Vastag, S.* A Framework for Simulation Models of Service-Oriented Architectures // In SIPEW 2008. – LNCS, Vol. 5119. – 2008. – P. 208–227.
4. *Bellin, D., Suchman-Simone, S.* The CRC Card Book. – Addison-Wesley. – 1997.
5. *Benjamin, P., Patki, M., Mayer, R.* Using ontologies for simulation modeling // In WSC'06. – 2006. – P. 1151-1159.
6. *Bertolino, A., De Angelis, G., Polini, A.* A QoS Test-Bed Generator for Web Services // In ICWE'07. – LNCS, Vol. 4607. – Springer. – 2007. – P. 17-31.
7. *Bosch, J.* Design and Use of Software Architectures. – Reading: Addison-Wesley. – 2000.
8. Business Process Model and Notation (BPMN) Version 1.2. –. OMG. – 2009.
9. *Carroll, J.M. (ed.).* Scenario-Based Design. – Wiley. – 1995.
10. *Cooper, A.* The Inmates are Running the Asylum. – Sams. – 2004.
11. *Cortellessa, V., Grassi, V.* Reliability Modeling and Analysis of Service-Oriented Architectures // In Baresi, L., Nitto, E.D. (eds.): Test and Analysis of Web Services. – Springer. – 2007. – P. 339-362.
12. *Cortellessa, V., Pierini, P., Spalazzese, R., Vianale, A.* MOSES: MOdeling Software and platform architEcture in UML 2 for Simulation-based performance analysis // In QoSA 2008. – LNCS, Vol. 5281. – Springer. – 2008. – P. 86-102.
13. *DeVellis, R.F.* Scale Development: Theory and Applications. – Sage Pubs. – 2003.
14. *Diehl, S.* Software visualization: visualizing the structure, behaviour, and evolution of software. – Springer. – 2007.
15. *Drew, P., Raymond, G., Weinberg, D.* Talk and Interaction in Social Research Methods. – Sage Pubs. – 2006.
16. *Driss, M., Jamoussi, Y., Jezequel, J.-M., et al.* A Discrete-Events Simulation Approach for Evaluation of Service-Based Applications // In Proc. ECOWS'08. – IEEE. – 2008. – P. 73-78.
17. *Dumas, M., Van der Aalst, W., ter Hofstede, A. (eds.).* Process-Aware Information Systems. – Wiley-IEEE. – 2005.
18. *Egyed, A.* Dynamic Deployment of Executing and Simulating Software Components // In Component Deployment. – LNCS, Vol. 3083. – Springer. – 2004. – P. 113-128.
19. *Floyd, C.* A Systematic Look at Prototyping // In Approaches to Prototyping. – Springer. – 1984. – P. 1-17.
20. *Fritzsche, M., Gilani, W., Fritzsche, C., et al.* Towards utilizing model-driven engineering of composite applications for business performance analysis // In ECMDA-FA'08. – 2008. – P. 369-380.
21. *Fritzsche, M., Picht, M., Gilani, W., et al.* Extending BPM environments of your choice with performance related decision support // In BPM 2009. – LNCS, Vol. 5701. – 2009. – P. 97-112.
22. *Galle, D., Kop, C., Mayr, H.C.* A Uniform Web Service Description Representation for Different Readers // In Proc. ICDS'08. – IEEE CS Press. – 2008. – P. 123-128.
23. *Garcia-Banuelos, L., Dumas, M.* Towards an Open and Extensible Business Process Simulation Engine // In Proc. CPN'09. – 2009.
24. *Gokhale, S.S., Lyu, M.R., Trivedi, K.S.* Reliability Simulation of Component-Based Software Systems // In Proc. ISSRE'98. – 1998. – P. 192-201.
25. *Graham, I.* Requirements Modelling and Specification for Service Oriented Architecture. – Wiley. – 2009.
26. *Grassi, V., Mirandola, R., Sabetta, A.* Filling the gap between design and performance/reliability models of component-based systems: A model-driven approach // The Journal of Systems and Software. – 2007. – Vol. 80. – P. 528–558.

27. *Gregoriades, A., Sutcliffe, A.* Scenario-Based Assessment of Nonfunctional Requirements // IEEE Trans. Software Eng. – 2005. – Vol. 31. – N 5. – P. 392-409.
28. *Grishikashvili Pereira, E., Pereira, R.* Simulation of fault monitoring and detection of distributed services // Simulation Modelling Practice and Theory. – 2007. – Vol. 15. – P. 492–502.
29. *Harel, D., Politi, M.* Modeling Reactive Systems with Statecharts. – McGraw-Hill. – 1998.
30. *Haumer, P., Heymans, P., Jarke, M., Pohl, K.* Bridging the Gap Between Past and Future in RE: A Scenario-Based Approach // In Proc. RE'99. – IEEE CS Press. – 1999. – P. 66-73.
31. *Hennig, A., Hentschel, A., Tyack, J.* Performance Prototyping - Generating and Simulating a Distributed IT-System from UML Models // In Proc. ESM'2003. – IEEE. – 2003.
32. *Hesse, W.* Ontologies in the Software Engineering process // In Proc. EAI'05. – Ceur-WS.org, Vol. 141. – 2005.
33. *Hesse, W., Mayr, H.C.* Modellierung in der Softwaretechnik: eine Bestandsaufnahme // Informatik Spektrum. – 2008. – Vol. 31. – N 5. – P. 377-393.
34. *Ionescu, T.B., Piater, A., Scheuermann, W., et al.* An Aspect-Oriented Approach for Disaster Prevention Simulation Workflows on Supercomputers, Clusters, and Grids // In Proc.DS-RT 2009. – 2009. – P. 21-33.
35. *ISO.* ISO/IEC 12207:2008, Information technology – Software life cycle processes. –. – 2008.
36. *Jansen-Vullers, M., Netjes, M.* Business process simulation – a tool survey // In CPN Tools Workshop. – 2006.
37. *Jensen, K., Kristensen, L.M.* Coloured Petri Nets. – Springer. – 2009.
38. *Jureta, I.J., Herssens, C., Faulkner, S.* A Comprehensive Quality Model for Service-Oriented Systems // Software Quality Journal. – 2009. – Vol. 17. – N 1. – P. 65-98.
39. *Jwoa, J.-S., Cheng, Y.C.* Pseudo software: A mediating instrument for modeling software requirements // Journal of Systems and Software. – 2009, in press.
40. *Kaschek, R., Kop, C., Shekhovtsov, V.A., Mayr, H.C.* Towards Simulation-Based Quality Requirements Elicitation: A Position Paper // In REFSQ 2008. – LNCS, Vol. 5025. – Springer. – 2008. – P. 135-140.
41. *Kazman, R., Barbacci, M., Klein, M., Carriere, S.J.* Experience with Performing Architecture Tradeoff Analysis // In Proc. ICSE'99. – ACM. – 1999.
42. *Kelton, W.D., Sadowski, R.P., Sadowski, D.A.* Simulation with Arena. – McGraw-Hill. – 2004.
43. *Kounev, S.* Performance Modeling and Evaluation of Distributed Component-Based Systems using Queueing Petri Nets // IEEE Trans Soft Eng. – 2006. – Vol. 32. – N 7. – P. 486-502.
44. *Lassing, N., Bengtsson, P., Bosch, J., Vliet, H.V.* Experience with ALMA: Architecture-Level Modifiability Analysis // Journal of Systems and Software. – 2002. – Vol. 61. – N 1. – P. 47-57.
45. *Lauesen, S.* Software requirements: Styles and techniques. – Addison-Wesley. – 2002.
46. *Lera, I., Sancho, P.P., Juiz, C., et al.* Performance assessment of intelligent distributed systems through software performance ontology engineering (SPOE) // Software Qual J. – 2007. – Vol. 15. – P. 53-67.
47. *Looker, N., Gwynne, B., Xu, J., Munro, M.* An Ontology-Based Approach for Determining the Dependability of Service-Oriented Architectures // In WORDS'05. – 2005. – P. 171- 178.
48. *Looker, N., Xu, J., Munro, M.* Determining the dependability of Service-Oriented Architectures // Intl J of Simulation and Process Modelling. – 2007. – Vol. 3. – N 1-2. – P. 88 - 97.
49. *Mårtensson, F., Jönsson, P., Bengtsson, P., et al.* A Case Against Continuous Simulation for Software Architecture Evaluation // In Proc. ASM'03. – 2003. – P. 97-105.
50. *Marzolla, M., Balsamo, S.* UML-PSI: the UML Performance SImulator // In Proc. QEST'04. – IEEE. – 2004.
51. *Mayr, H.C., Bever, M., Lockemann, P.C.* Prototyping Interactive Application Systems // In Budde, R., Kuhlenkamp, K., Mathiassen, L. (eds.): Approaches to Prototyping. – Berlin: Springer-Verlag. – 1984. – P. 105-121.
52. *Mayr, H.C., Kop, C.* Conceptual Predesign - Bridging the Gap between Requirements and Conceptual Design // In Proc. ICRE '98. – IEEE CS Press. – 1998. – P. 90-100.
53. *McManus, J.* Managing Stakeholders in Software Development Projects. – Butterworth-Heinemann. – 2004.
54. *Memmel, T., Gundelsweiler, F., Reiterer, H.* Prototyping Corporate User Interfaces - Towards a Visual Specification of Interactive Systems // In Proc. IASTED-HCI 2007. – 2007.
55. *Miller, M.J., Pulgar-Vidal, F., Ferrin, D.M.* Achieving Higher Levels of CMMI Maturity Using Simulation // In Proc. WSC'02. – IEEE. – 2002. – P. 1473-1478.
56. *Moreira, A., Araújo, J., Brito, I.* Crosscutting quality attributes for requirements engineering // In SEKE'02. – 2002. – P. 167-174.
57. *North, M.J., Macal, C.M.* Managing Business Complexity: Discovering Strategic Solutions with Agent-Based Modeling and Simulation. – Oxford Univ. Press. – 2007.
58. *Pastor, O., Molina, J.C.* Model-Driven Architecture in Practice. – Springer. – 2007.
59. *Paternò, F.* Model-based Design and Evaluation of Interactive Applications. – Springer. – 2000.
60. *Popic, P., Desovski, D., Abdelmoez, W., Cukic, B.* Error Propagation in the Reliability Analysis of Component based Systems // In Proc. ISSRE'05. – IEEE. – 2005.
61. *Robertson, S., Robertson, J.* Mastering the Requirements Process, 2nd ed. – Addison-Wesley. – 2006.
62. *Robinson, S.* Conceptual modelling for simulation // J Operational Research Society. – 2008. – Vol. 59. – N 3. – P. 278-290.
63. *Rosa, M.L., Reijers, H.A., van der Aalst, W., et al.* APROMORE: An Advanced Process Model Repository. – QUT Reprints. – 2009.
64. *Rozinat, A., Wynn, M., van der Aalst, W., et al.* Workflow simulation for operational decision support // Data & Knowl Eng. – 2009. – Vol. 68. – P. 834–850.
65. *Rücker, B.* Building an Open Source Business Process Simulation Tool with JBoss jBPM. – Stuttgart UAS. – 2008.
66. *Sargent, R.G.* Verification and Validation of Simulation Models // In Proc. WSC'08. – 2008. – P. 157-169.
67. *Schauerhuber, A., Schwinger, W., Kapsammer, E., et al.* Towards a Common Reference Architecture for Aspect-Oriented Modeling // In Proc. 8th Intl. Workshop on AO Modeling. – 2006.
68. *Seffah, A., Gulliksen, J., Desmarais, M.C. (eds.).* Human-Centered Software Engineering. – Springer. – 2005.
69. *Shekhovtsov, V., Kaschek, R., Zlatkin, S.* Constructing POSE: a Tool for Eliciting Quality Requirements // In Proc. ISTA 2007. – LNI, Vol. P-107. – GI. – 2007. – P. 187–199.
70. *Shekhovtsov, V.A., Kaschek, R., Kop, C., Mayr, H.C.* Relational service quality modeling // In 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.
71. *Sindre, G., Opdahl, A.L.* Eliciting security requirements with misuse cases // Req. Eng. – 2005. – Vol. 10. – N 1. – P. 34-44.

72. *Sutcliffe, A., Ryan, M.* Experience with SCRAM: a scenario requirements analysis method // In Proc. ICRE'98. – IEEE CS Press. – 1998. – P. 164-171.
73. *Tewoldeberhan, T., Janssen, M.* Simulation-based experimentation for designing reliable and efficient Web service orchestrations in supply chains // El.Commerce Res. and Apps. – 2008. – Vol. 7. – P. 82–92.
74. *Thanheiser, S., Liu, L., Schmeck, H.* SimSOA: an approach for agent-based simulation and design-time assessment of SOC-based IT systems // In Proc.ACM Symp. on Applied Computing. – ACM. – 2009. – P. 2162-2169.
75. *Tolstedt, J.L.* Prototyping as a means of requirements elicitation // In Proc. SAE International Off-Highway Congress. – SAE Technical Paper Series, Vol. 2002-01-1466. – 2002.
76. *Trætteberg, H., Krogstie, J.* Enhancing the Usability of BPM-Solutions by Combining Process and User-Interface Modelling // In Proc. POeM'08. – LNBIP, Vol. 15. – Springer. – 2008. – P. 86-97.
77. *Tsai, W.T., Cao, Z., Wie, X., et al.* Modeling and Simulation in Service-Oriented Software Development // Simulation. – 2007. – Vol. 83. – N 1. – P. 7-32.
78. *Tullis, T., Albert, W.* Measuring the User Experience. – Morgan Kaufmann. – 2008.
79. *Um, I.-S., Lee, H.-C., Cheon, H.-J.* Determination of Buffer Sizes in Flexible Manufacturing System by using the Aspect-oriented Simulation // In Proc. ICCAS'07. – IEEE. – 2007. – P. 1729-1733.
80. *van der Aalst, W.* Formalization and verification of event-driven process chains // Inf Soft Tech. – 1999. – Vol. 41. – N 10. – P. 639-650
81. *van der Aalst, W., Nakatumba, J., Rozinat, A., Russell, N.* Business process simulation: how to get it right? – TU Eindhoven. – 2008.
82. *Waller, A., Clark, M., Enstone, L.* L-SIM : Simulating BPMN Diagrams With A Purpose Built Engine // In Proc. WSC'06. – 2006. – P. 591-597.
83. *Ware, C.* Information Visualization: Perception for Design. – Morgan Kaufmann. – 2004.
84. Web Services Business Process Execution Language (WS-BPEL) Version 2.0. –. OASIS. – 2007.
85. *Wells, L.* Performance analysis using coloured Petri nets // In MASCOTS'02. – 2002. – P. 217–221.
86. *Williams, L.G., Smith, C.U.* PASA: A Method for the Performance Assessment of Software Architecture // In Proc. 3rd Workshop on Software Performance. – 2002.
87. *Глушков, В.М.* О системной оптимизации // Кибернетика. – 1980. – № 5. – С. 89–91.
88. *Згуровский, М.З., Панкратова, Н.Д.* Системный анализ: Проблемы, методология, приложения. – К: Наук. думка. – 2005. – 743 с.
89. *Менаске, Д., Алмейда, В.* Производительность Web-служб. Анализ, оценка и планирование. – К: Диасофт. – 2003.