Methods for Ensuring the Quality of Functioning in the Life Cycle of Automated Information Systems of a Modern Enterprise

Boris Pozin[1,2] [0000-0002-0012-2230]

[1]National Research University Higher School of Economics,
20 Myasnitskaya St., 101000, Moscow, Russia
[2]EC-leasing, Building 1, 125 Warshavskoye Shosse, 117587, Moscow, Russia
bpozin@ec-leasing.ru

Abstract. An approach to ensuring the quality of functioning of automated enterprise systems based on the concept of the ISO/IEC 15026 standard-providing the necessary level of system integrity in its life cycle. The problem statement and methods suitable for solving the problem are considered.

Keywords: integrity level, ensuring the quality of functioning, life cycle, automated information system, release management.

1. Introduction

The life cycle of an enterprise's automated information system (AIS) is usually several years (or even tens of years). During this period, the system functionality changes: new functional requirements are implemented in the system due to changes in the business. The architecture of modern systems is designed with a focus on minimizing the cost of making functional changes. This has led to the emergence of change-resistant solutions for a whole class of systems, such as a microservice architecture, which allows you to implement and make changes to the system based on Continuous technologies [1] in the form of a set of services (microservices) included in the system. This partly solves the problem of reducing the cost of making changes.

Over the course of operation, the number of end users of such systems and the intensity of requests to the Central part of the system increase, and corporate databases and data warehouses fill up. As a result, the average system response time

Proceedings of the of the XXIII International Conference "Enterprise Engineering and Knowledge Management" (EEKM 2020), Moscow, Russia, December 8-9, 2020.

to a user request increases, and system performance decreases. You need to optimize the computing process and / or upgrade the system, including hardware.

The quality of the system operation is usually characterized by the probability of residual errors, especially in the Central part of the system (critical applications: economic subsystems, filling of corporate storage, supply chains, etc.).

As the owner of the system develops automation capabilities, and the developers of the as subject area, these as are continuously improved based on the business needs formulated by the owner of the system as the customer. The effectiveness of a system is determined by how much it can improve the quality of the business, that is, improve the ability to meet the needs of the system owner and his customers. Most often this means: reduce the complexity and duration of solving business tasks of the owner organization, as well as increase labor productivity when performing business processes of the company within the specified time limits.

The AIS includes not only hardware, software, and information support, but also people- personnel and end-users of the system. It is people, the speed of mastering the changes they make, that determine the restrictions on the timing of putting new system functionality into operation, and the actual changes in automated business processes that are being put into operation. New features of the system are introduced at a pace in which the system personnel is able to master and apply in practice (in business) the changes made. This is why Continuous-technology methods alone are not enough to make changes to the AIS.

Personnel who operate and maintain the system must have criteria, tools, and procedures that allow them to:

- evaluate the state of the system's behavior and the quality of its operation;
- identify cases of regression, that is, the impact of changes made on the unchanged part of the system, as well as manifestations of such errors in the functioning of the system;
- make changes to the system without affecting its performance;
- define a set of activities of personnel responsible for maintaining the integrity of the system during its operation, maintenance and development.

2. Ensuring the integrity and release management

2.1 Integrity.

It is extremely relevant for systems used for managing targeted activities to develop a single measure to assess the quality of the system's functioning in the current period of time to meet the needs of the system owner, including a set of activities of personnel responsible for maintaining the integrity of the system during its operation, maintenance and development. Especially in cases where the performance of the system significantly affects the characteristics of the business.

In our opinion, the approach developed in the international standard [4], which considers a model based on systematic consideration of system integrity levels in risk analysis, is the most suitable for setting such a task. This approach reflects the state of

the system during operation and development, as well as the owner's point of view on the system. "The integrity level refers to specifying the range of object property values required to keep the system risk within acceptable limits. For objects whose failure may lead to a threat to the system, this property is a limit on the frequency of such failure" [4]. With this approach, the uneven implementation of programs that implement new functional requirements for the system, possible changes in acceptable risk levels, and the need for guaranteed confirmation of non-functional requirements that ensure business continuity can be compensated by using test planning.

Almost immediately after the system starts operating, the task of controlling the level of system integrity when making changes to it arises. As a rule, in the process of developing a system, many integrity control tasks are not yet realized, primarily because there is not enough information and experience in using the future system in the owner's business processes to set and solve this problem at the development stage.

In the approach used in [4], ensuring and controlling the preservation of the system integrity level is the goal at all stages of the system lifecycle: during operation, maintenance, development and modernization. The intensity of requests for changes is very high, usually there are several hundred or thousands of requests per year. Daily changes to the system – in its hardware, software, changing the role functions of personnel is impossible technologically, and can lead to a significant decrease in the reliability of the system and its other operational characteristics. To maintain a high level of system integrity, its development is usually carried out in releases.

Each release should be characterized by an appropriate level of integrity, in which it implements the main planned functions, and the use of the system under the planned load does not lead to violations of business continuity (established business regulations) when performing the most critical business functions.

In essence, this means that each release of the system (including the release of the system's software) must be monitored for maintaining the integrity level. Violation of integrity leads to the implementation of systemic risks, that is, to the disruption of business continuity, and consequently to losses for the business – both material and reputational. However, different types of risks may have different consequences for the business. It is unacceptable that the AIS negatively affects the business continuity of the owner, and the level of business continuity must be formulated. For example, "when the system is running in 7*24 mode, the recovery time after a possible failure should not exceed T minutes." The time T is chosen taking into account the prevention of material or reputational losses of the owner and should be regulated in the internal documents of the owner company. It is this criterion of continuity that becomes the basis for a rough, but technologically acceptable assessment by personnel of the quality of the system's functioning during the operation of its release.

2.2 Releases of the system.

Step-by-step study and implementation of business needs by developers and available capabilities by AIS owners leads to the fact that the AIS (including application and system software, hardware) is developed in stages by releases that are budgeted,

formed and put into operation on a regular basis (once a month, quarter, etc.) in accordance with the maintenance and development plan (usually annual).

The new release differs in that it is in relation to the previous one:

- Functional subsystems are modified, or the operating environment of their execution and / or parameters that affect the performance of functions by subsystems are configured so that the system performance characteristics are within the specified limits;
- Reasonable modernization of telecommunications equipment and facilities was carried out;
- Added software subsystems/services with new functionality to the system and / or changed the functionality of existing subsystems;
- Fixed detected errors in the software (in volume - according to the agreed plan).

When developing a release, you need to:

- test the system comprehensively to identify regression facts (negative impact of changes made on parts of the AIS software that were not changed);
- test the system under load, that is, with a prospective change in the number of requests of various business processes by both end users and messages from related systems;
- train end-users of the system and personnel who perform operation and maintenance of the new capabilities of the automated control system;
- make changes to the system documentation due to changes made to the AIS.

In this way, the system lifecycle includes release planning in all aspects (planning for functionality, quality, integrity, budgeting for releases, and allocating resources to implement releases planned for a medium-term period – such as a year).

2.3 Activities to ensure integrity.

During the operation, maintenance and development of the system, it is necessary to organize systematic quality control and assurance activities. This means that two main processes must be set up on a regular basis:

- measurement and recording of quality and integrity metrics.
- systematic load testing of the system release under development until it is put into operation, which should be carried out in parallel.

The first task should be solved by the system operation service based on measuring the frequency of execution of various business chains that implement automated business processes. This will make it possible to clarify the quantitative characteristics of the system functioning in different modes. Recording system crushes, defects, and failures will not only allow you to evaluate the achieved characteristics and the level of system integrity, but also determine the requirements for improving its characteristics. If the BPMS model of the system is used as a base for system integration, it will also be useful for solving these problems.

The presence of such a" digitized" model provides the basis for planning load testing of a new release of the system and refining the models [2] used for load testing. It is the regulation and load models that can be used as the basis for integrity control. Systematic measurements help to predict the prospects for the development of the load when it changes sequentially, that is, when it does not change dramatically. With significant, drastic changes in the load, the use of measurements makes it possible to predict the presence of mutual influence of the intensity of individual tasks and groups of tasks, which improves the quality of load testing.

2.4 AIS infrastructure to provide a level of integrity.

Direct use of AIS resources to solve the problem of ensuring a given level of integrity during operation and maintenance of the system will lead to deterioration of its operational characteristics. Therefore, to support integrity processes, it is necessary to create a separate maintenance infrastructure that does not take resources from the operating AIS to perform the specified work, and also allows you to parallelize the work on operating the system and ensuring its integrity when preparing new releases. Having a separate infrastructure for quality assurance reduces the risk of business continuity disruption. this is a significant part of the price paid by the system owner for maintaining its quality during operation and for increasing the system's lifetime.

The experience of implementing systems for ensuring the life cycle of the AIS [5,12,15,16] has shown that if there is a maintenance infrastructure, the actual infrastructure of the AIS is not monolithic, it is usually structured into at least three components (areas):

- Development area - for receiving the results of development of the release components;
- Maintenance area - for integration of components into the release and comprehensive testing of the release;
- Operation area - for the operation of the output as part of the system.

The architecture of these areas depends on the information security (IS) concept used in the AIS. Accordingly, the levels of IS areas differ both in the test data used (open, depersonalized, real), in the requirements for the degree of readiness and security of software components used and available to the staff of IS tools, and in the possibility of using tools in each area. For example, compilers and tools for making changes can only be placed in the development area. In other areas, you can only use tools that are not related to working with the source code of programs. In the field of operation, the use of any tools is generally prohibited. Naturally, each area has requirements for safe and effective actions of personnel.

3. Statement of the problem of supervision the system integrity level

In our opinion, the approach developed in the international standard [4], which considers a model based on a systematic analysis of system integrity levels in risk analysis, seems to be the most suitable for setting such a task. It is this approach that reflects the state of the system during operation and development, as well as the owner's point of view on the system. With this approach, it is possible to compensate for factors that potentially violate the integrity of the system: uneven commissioning of programs that implement new functional requirements for the system, the difficulty of confirming non-functional requirements that ensure business continuity, etc.

Compensation of the influence of these factors can be achieved by

- testing planning, in which it is possible to take into account the potential change from release to release of acceptable risk levels that occurs due to the quality of the changing components of the application software of the system or the system as a whole being put into operation as part of the release;
- implementing the plan in the course of preparing the release for putting into operation.

Almost immediately after the system starts to operate, the task of supervision the level of integrity of the system when making changes to it arises. As a rule, in the process of developing a system, many integrity control tasks are not yet recognized, primarily because there is not enough information and experience in using the future system in the owner's business processes to formulate and solve this problem at the development stage.

Typically, developers by the functional integrity of systems understand the degree of completeness of the implementation of the planned changes in functionality. The most often considered as non-functional characteristics of systems are their performance (the number of realized requests to the system - on average, or by the types of processed requests with some background load), the speed of query execution under the background load (on average or by types of requests), etc. In practice, this approach is far from always acceptable. An automated system is just a tool of the owner company to improve its business processes: to improve the quality of customer service or to reduce costs. Typically, the company establishes regulations the execution of certain business objectives, taking into account the real possibilities of the staff and the level of automation of the company and its contractors. These regulations, among other things, establish acceptable time intervals for solving business problems. To solve a business problem of a certain type, it may be necessary to perform several queries to the system, including more than once. Failure to complete sets of business operations, including those supported by automation tools, within acceptable time intervals may mean loss of income and / or reputation for the business.

From the point of view of the owner of the system, supervision the integrity level means not only controlling the degree of completeness of the planned changes in

functionality, but also the ability of the system to provide unconditional fulfillment of regulations at a given load: solving complex tasks for a business at time intervals established by regulations. Under the load refers to the number of requests for solving business problems per unit of time.

Violation of regulations may be critical for business [5] or have weaker levels of criticality. The ranges of changes in criticality levels are set in each company taking into account its business interests based on an analysis of regulations. Assessment of the integrity level should include all the described components of the implementation of the regulations: on the functionality of the system and on non-functional characteristics, taking into account the criticality of the company's business processes.

Thus, in essence, the following model of the integrity level control process is considered:

Let the set {M} of automated business processes of a company be given, each of which over a period of time T is performed with probability $p_i(t)$, i={1,M}. Let each business process be implemented by one or several chains of j subsystems, j={1,J};

In this regulation the fulfillment of a business process is set so that

$$\forall j \ \max_j t_j \ \leq \ \tau_i, \tag{1}$$

where

tj – the execution time of the chain j,

τ_i – is the regulatory restriction on the execution time of business process i.

Each chain j consists of software, which may contain a residual critical error inherent in this software in the current release. This residual error may occur when running the release with probability qj. The manifestation of residual error characterizes certain level of integrity for solving the problem of a particular business process, rather, execution of the corresponding chain j. The probability of the realization of risk p_r in this case is

$$p_{ri=} \ p_i * q_j \tag{2}$$

In this case, the risk level is characterized qualitatively (Lysunets and Pozin, 2013) by the value of φ_i so that for the entire release the probability of occurrence of risk is $p_R = \bigcup_i p_{ri} * \varphi_i$.

If φ_i = 1 at a critical risk level and 0 in other cases, then for critical chains:

$$p_R = \sum p_{ri} \tag{3}$$

The task of supervision the integrity level is reduced to the procedures:

- determining the expected risk levels for the release of the system;
- describing the functionality of the release, corresponding to the occurrence of risk in planning the monitoring of the integrity level;
- carrying out comprehensive functional testing of the release to identify residual system-plan errors (for example, those associated with incompatibility between the data or the boundaries of the domains of the variables of new components with unchanged components).;

- carrying out regression load testing to detect cases of residual errors;
- carrying out regression load testing to detect cases of potential violation of regulations (in terms of performance or efficiency of solving problems of chain j).

The absence of residual errors or violations of non-functional requirements for the release of the system during comprehensive testing of the release allows us to conclude that the release is verified. The release is put into operation and its validation is carried out within 1-2 weeks (depending on the technology applied by the owner). According to the available experience, the release works stably after the control period during which no more than 0-4 critical errors appear (and are eliminated) [5,12].

4. Release test planning

The need to control the level of integrity using the described approach leads to the organization of testing processes as iterative, regression, ensuring the fulfillment of tasks by the planned date of commissioning of a new release of the system to operation based on a typical comprehensive testing plan.

Testing planning is based on achieving the goals of creating a release, taking into account the general technology for integrating the release on time, and ensuring acceptable risks that are manifested through monitoring the integrity level using integrated testing methods. The risk level model, which contains a qualitative assessment of the risk level of each chain, is quite stable in general for a system whose structure does not change dramatically from release to release. Only for a few new release chains (one or more) can the risk level be specified. For example, if the release is associated with changes in regulatory documents, the priority of certain business tasks may change, which will change the risk level of the chains that automate these tasks. The risk level model is approved by the system owner or an authorized person.

Thus, the integrity control of each new release is performed by the following actions: determining the status of the integrity level of the changed chains (critical/ non-critical); monitoring the functional integrity of the changed chains from the system's point of view; checking how the integrity violation (manifestation of a residual error) can affect the integrity level of the entire system.

The goal is achieved using a single technology that is understood and approved by the system owner, since it is the owner's representatives who make the final decision on whether the system release meets the specified quality level. If there are resource or time constraints on testing the release by the planned date, the test plan is refined, taking into account the priority of the chains being finalized by integrity levels. The priority for testing in the current release are those chains that have higher risks.

5. Planning a comprehensive verification functional testing of the release

The main purpose of complex verification functional testing in the maintenance and operation processes is to control (after making changes) the functional integrity of software of the system release, as well as to control the presence of regression, that is, the appearance of errors in previously tested and correctly working parts of the software release.

In contrast to functional testing of software components, planning for complex verification testing of an AIS release is carried out along functional chains of interacting subsystems (services) in accordance with the plan for making changes. At the same time, the testing plan takes into account the need to check the most critical chains for operation. This approach allows you to test the functionality of the AIS release from a business perspective, and the coverage criteria are aimed at checking the quality of functionality implementation in terms of business, i.e. business processes, and not in the understanding of programmers.

Control over the completeness of the implementation of the functions of the AIS release is carried out to check the quality of implementation of all planned business functions. In addition, in the process of comprehensive verification functional testing of the release, input variables are monitored for each planned chain over the entire range of their changes.

This type of testing is characterized by a high complexity of preparing test data, so to reduce the cost of this type of testing, there may be different strategies for conducting it, taking into account the availability of resources and time to complete it.

To conduct functional testing, a functional testing plan is formed based on the functional requirements for the release of the AIS. The basis of this plan is a set of checks for each functional requirement in various business processes supported by the AIS. Each functional requirement is associated with one or more chains that implement this requirement. To test each test requirement, you need to plan and develop appropriate test cases, the expected test results and criteria for completeness of testing and its completion, as well as the composition of system-wide data on which to test the system: the necessary databases and their tables filled with data, normative reference data used in the automated business process.

The test plan will be implemented over a series of checks and multiple times, so it is usually valid for several months. The most representative test data and test cases can be accumulated and used during this period. In the case of automated functional testing, the formation and accumulation of test plans is carried out in automation tools that are selected based on this need.

The test plan will be implemented over a series of checks and multiple times, so it is usually valid for several months. The most representative test data and test cases can be accumulated and used during this period. In the case of automated functional testing, the formation and accumulation of test plans is carried out in automation tools that are selected based on this need.

6. Planning comprehensive verification load testing of the release

System load testing is used to assess the ability to meet business production regulations using a new release of the system, i.e. business continuity, as well as to assess the ability to achieve the necessary operational (non-functional) characteristics of the system when changing data processing technology, load, or when scaling the system. Among the non-functional characteristics, the most important are indicators of the system's purpose, such as performance, system response time, and so on.

Load testing of systems during operation and maintenance requires a special approach that differs from the methods used in building tools for load testing, such as [7-11, 14,17] and in general methods used at different stages of program development and maintenance [6,13].

Load testing is aimed at reducing the risks of violating regulations and disrupting business continuity in the organization that owns the system, when making agreed changes to it. The purpose of each of the stress experiments are based on the customer's need for assessment or prediction of operational characteristics of the information system and relevant non-functional requirements and based on requirements for periodic monitoring of degradation of operational characteristics of AIS. Note that all the characteristics of the system are evaluated by the methods of probability theory and mathematical statistics as a result of an experiment that provides as many measurements as possible for a reliable assessment of these characteristics.

For load testing formed the test plan, which is a feature that maintainer – tester must work with the operator and the representative of the organization system, to develop scenarios of functioning as adequate (in the opinion of the members of their development) reflect the method of use as in the organization – the owner to solve business problems. In this case, scenarios must contain statistically reliable input data of the system, in the number and frequency characteristics sufficient to form statistically valid estimates of the results: the operational characteristics of the AIS.

Key aspects of experiment planning are:

- statement of the problem, defining the objective of the experiment must be linked to the system requirements;
- clear definition of the boundaries of the test object;
- a scenario for the functioning of the AIS during a semi-natural experiment, which reflects the features of intensive use of the business processes of the owner organization;
- conditions for receiving the maximum load on the AIS during the experiment period, based on the features of the scenario;
- a set of necessary characteristics and indicators on the basis of which are determined by the test results.

The authors developed four metamodels [2], which are used to set the task of a load experiment to select the necessary characteristics, indicators, and measured values that adequately characterize the functioning of the tested information system:

Requirements Metamodel – characterizes the type of system under test, the composition of non-functional requirements (business rules and technical requirements) of the tested information system.

System Metamodel-describes the system structure as a network of Queuing systems (including the composition of elements of the "resource" type»);

Load Metamodel-is a description of the number and types of service requirements for the system, the law of distribution of service requirements over the time of the experiment, the rules for entering service requirements into the system, the entry points of service requirements into the system (logical level);

Metamodel of Measurements-defines the composition of the collected characteristics, indicators and values, the interface for entering requirements into the system, the method of their collection and conversion algorithms, as well as criteria for evaluating the results obtained.

When planning a new load experiment using metamodels, models of requirements, systems, loads, and measurements are formed by selecting meta-concepts and determining their values based on the properties of the information system under test and the goals of the load experiment. By using metamodels when planning a new load experiment, the completeness and integrity of the generated models is ensured. The specified models may differ for different types of load testing and different systems.

7. Conclusion

1. The paper proposes an approach to ensuring the quality of functioning in the life cycle of automated enterprise systems, based on maintaining the level of business continuity and integrity of the system during its operation, maintenance and development. The approach is based on the ability to plan changes and assess the level of integrity both during the operation of the system, and during the preparation and after the introduction of system releases, that is, a set of planned changes.

2. It shows the need for changes to be made by planned releases that differ in functionality and / or change in the technical characteristics of the system, but do not change the business continuity.

3. A model has been developed to assess the level of integrity in the design, operation, and load testing of new system releases.

4. It is shown that it is necessary to develop the design and creation of infrastructure for managing AIS releases and separate budgeting of these works in order to reduce the risks of business continuity disruption, the investment of the system owner in maintaining its quality during operation and in increasing the system lifetime. This infrastructure, combined with release management processes, allows you to make changes to the system without disrupting its performance.

5. Methods of planning complex verification functional and load testing based on the development experience are proposed.

References

1. Balalaie, A.; Heydarnoori, A.; Jamshidi, P. Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture. IEEE Software. 2016; 33(3):42-52. doi: 10.1109/MS.2016.64.
2. B.A.Pozin and I.V.Galakhov. Models in Performance Testing. Programming and Computer Software, 2011, Vol. 37, No. 1, pp. 15–25.
3. ISO/IEC/IEEE 12207:2017 "System and software engineering - Software life cycle processes"
4. ISO/IEC 15026-2:2011 Systems and software engineering - Systems and software assurance - Part 2: Assurance case
5. Lysunets A.S., Pozin B.A. Planning for integrity assurance of complex automated banking system methods of functional and load testing. Software engineering. 2013; (3):8-14.
6. ISO/IEC/IEEE 29119-4:2015(E) Software and systems engineering - Software testing - Part 4: Test techniques.
7. Zhen Ming Jiang; Ahmed E. Hassan, A Survey on Load Testing of Large-Scale Software Systems. IEEE Transactions on Software Engineering. 2015; 41(11):1091-1118.
8. Korotkov A.O., Galakhov I.V., Matyuhin K.A. Testing automation based on equivalence class partition for system processing electronic documents. Proceedings of the 4th Conference “Actual problems of System and Software Engineering”, Moscow, 2015, p.p.86-87
9. Blokdyk, G. (2018). Software performance testing A Complete Guide, 5starcooks.
10. Dürr, G. (2010). Testing the Performance of Complex System Simulations, VDM Verlag.
11. Glavich, P., Farrell, C., Massey, C. (Ed.) (2010). .NET Performance Testing and Optimization - the Complete Guide, Red Gate Books.
12. Gotsutsov, S. (2017) Testing environment - technological platform for AIS PFR-2 life cycle management (Russian), Actual Problems of System and Software Engineering 2017, CEUR-WS, vol.1989, 2017, pp.209-214
13. ISO/IEC 14764:2006 “Software Engineering - Software Life Cycle Processes – Maintenance”
14. Klemm, B. (2013). Application Performance Testing: A Universal Performance Testing Methodology, pp.113, Kindle Series.
15. Pozin B. Life cycle ensuring and sustainment: goals, differences, results. 2019 Actual Problems of Systems and Software Engineering APSSE 2019 (Invited Papers), IEEE Computer Society, pp. 99-102
16. Pozin B.A. The Principles of Life Cycle Supporting System for Mission-Critical Systems. Trudy ISP RAN/Proc. ISP RAS, vol.30, issue 1, 2018, pp. 103-114. doi: 10.15514/ISPRAS-2018-30 (1) -7
17. Molyneaux, I. (2014). The Art of Application Performance Testing, 2nd ed., O′Reilly.