Author: Stecuk Maksim 1.2

Bibliography

1. Copeland L. A practitioner's guide to software test design. – Artech House, 2004.
2. Patton R. Software Testing. – Sams, 2005.

Terminological dictionary

Most of the definitions are taken according to standards of ISTQB

Word / phrase	***Transcription***	***Definition***	***Translation***
Black box testing	[blæk bɒks ˈtɛstɪŋ]	Testing that examines the functionality of an application without knowledge of its internal structures or workings.	*Тестирование черного ящика*
White box testing	[waɪt bɒks ˈtɛstɪŋ]	Testing technique where the internal structure and logic of the software are analyzed. Testers have access to the source code and an understanding of the internal workings of the software, allowing them to create tests based on this knowledge.	*Тестирование белого ящика*
Gray box testing	[greɪ bɒks ˈtɛstɪŋ]	Testing approach that combines elements of both white box and black box testing. Testers have limited knowledge of the internal workings of the software and use this knowledge to develop tests without fully disclosing implementation details.	*Тестирование серого ящика*
Acceptance testing	[əkˈsɛptəns ˈtɛstɪŋ]	Testing conducted to determine whether or not a system satisfies its acceptance criteria and to enable the customer to determine whether or not to accept the system.	*Приемочное тестирование*
Ad hoc testing	[æd hɒk ˈtɛstɪŋ]	Testing carried out informally, often without a planned approach or strategy.	*Ад-хок тестирование*
Alpha testing	[ˈælfə ˈtɛstɪŋ]	Testing of an application when development is nearing completion. It is carried out by testers within the organization.	*Альфа-тестирование*
Beta testing	[ˈbeɪtə ˈtɛstɪŋ]	Testing a product in the real environment by real users before its release to identify potential issues.	*Бета-тестирование*
Boundary testing	[ˈbaʊndəri ˈtɛstɪŋ]	Testing the behavior of a software application at the boundaries or extremes of input ranges.	*Тестирование границ*
Compatibility testing	[kəmˌpætəˈbɪlɪti ˈtɛstɪŋ]	Testing to ensure that software can run on different hardware, operating systems, browsers, networks, etc.	*Тестирование совместимости*
End-to-end testing	[ɛnd tuː ɛnd ˈtɛstɪŋ]	Testing a complete application environment from start to finish to simulate real-world usage scenarios.	*Тестирование от начала до конца*
Functional testing	[ˈfʌŋkʃənl ˈtɛstɪŋ]	Testing the functionality of a software application to ensure that it behaves according to specifications.	*Функциональное тестирование*
Integration testing	[ˌɪntɪˈgreɪʃən ˈtɛstɪŋ]	Testing the integration of different components or modules of a software system to verify that they work together as expected.	*Интеграционное тестирование*
Load testing	[ləʊd ˈtɛstɪŋ]	Testing the performance of a system under specific load conditions to assess its scalability and reliability.	*Тестирование нагрузки*
Performance testing	[pəˈfɔːməns ˈtɛstɪŋ]	Testing the speed, responsiveness, and stability of a software application under various conditions.	*Тестирование производительности*
Regression	[rɪˈgrɛʃən	Testing conducted to ensure that changes to the	*Регрессионное*

testing	ˈtɛstɪŋ]	software have not adversely affected existing functionality.	*тестирование*
Smoke testing	[sməʊk ˈtɛstɪŋ]	Preliminary testing to quickly evaluate whether the basic functionalities of a software application work correctly.	*Дымовое тестирование*
Stress testing	[strɛs ˈtɛstɪŋ]	Testing the stability and reliability of a system beyond its normal operational capacity, often to the point of failure.	*Тестирование на устойчивость*
Usability testing	[ˌjuːzəˈbɪlɪti ˈtɛstɪŋ]	Testing conducted to evaluate how user-friendly and intuitive a software application is for end-users.	*Тестирование удобства использования*
Agile testing	[ˈædʒaɪl ˈtɛstɪŋ]	Testing conducted in an agile development environment, focusing on continuous feedback, flexibility, and adaptability.	*Гибкое тестирование*
Agile	[ˋædʒaɪl]	Iterative software development methodology that emphasizes flexibility, collaboration, and customer feedback. It focuses on delivering working software in small, frequent releases, enabling teams to respond quickly to changes in requirements or market conditions.	*Гибкая методология разработки*
Exploratory testing	[ɪkˈsplɒrətəri ˈtɛstɪŋ]	Testing without predefined test cases, allowing testers to explore the application and identify defects.	*Исследовательское тестирование*
Monkey testing	[ˈmʌŋki ˈtɛstɪŋ]	Random testing performed without a specific test plan to discover unexpected errors or defects.	*Тестирование методом "обезьяны"*
Non-functional testing	[nɒnˈfʌŋkʃənl ˈtɛstɪŋ]	Testing aspects of a software application that do not relate to its functionality, such as performance, security, and usability.	*Нефункциональное тестирование*
Functional testing	[ˈfʌŋkʃənl ˈtɛstɪŋ]	Process of verifying that each function of a software application operates in accordance with the requirements specified in the project documentation. It ensures that the system performs its intended tasks correctly, confirming that all features and functionalities work as expected and meet the defined functional requirements.	*Функциональное тестирование*
Debugging	[ˈdiːˌbʌgɪŋ]	The process of identifying and fixing errors, bugs, or defects in software.	*Отладка*
Test case	[tɛst keɪs]	A set of conditions or variables under which a tester will determine whether an application, system, or one of its features is working correctly.	*Тест-кейс*
Test plan	[tɛst plæn]	A document outlining the scope, approach, resources, and schedule for testing activities.	*План тестирования*
Test scenario	[tɛst sɪˈnɑːrɪəʊ]	A detailed outline of a specific test that defines the test environment, actions to be performed, and expected results.	*Тестовый сценарий*
Test automation	[tɛst ɔːtəˈmeɪʃən]	The use of software tools and scripts to automate the execution of tests and comparison of actual outcomes with expected outcomes.	*Автоматизация тестирования*
Test suite	[tɛst swiːt]	A collection of test cases or test scripts intended to be used together for testing a specific aspect of a software application.	*Набор тестов*
Test	[tɛst	The setup and configuration of software and hardware	*Тестовая среда*

environment	ɪnˈvaɪrənmənt]	used for testing, including servers, databases, networks, and tools.	
Test metrics	[tɛst ˈmɛtrɪks]	Quantifiable measures used to assess various aspects of the testing process, such as test coverage, defect density, and test efficiency.	*Метрики тестирования*
Test strategy	[tɛst ˈstrætədʒi]	A plan that outlines the approach to be used for testing a software application, including test objectives, scope, and resources.	*Тестовая стратегия*
Test coverage	[tɛst ˈkʌvərɪdʒ]	A measure of the extent to which a software application has been tested, usually expressed as a percentage of code or requirements covered.	*Покрытие тестами*
Test driven development	[tɛst ˈdrɪvn dɪˈvɛləpmənt]	A development approach where tests are written before the corresponding code, driving the design and implementation process.	*Разработка через тестирование*
Test report	[tɛst rɪˈpɔːt]	A document summarizing the results of testing activities, including test execution status, defects found, and recommendations.	*Отчет о тестировании*
Test validation	[tɛst ˌvælɪˈdeɪʃən]	The process of confirming that the software meets its intended requirements and specifications through testing.	*Тестовая валидация*
Equivalence partitioning	[ɪˈkwɪvələns ˈpɑːtɪʃənɪŋ]	Dividing input data into groups that are likely to behave similarly during testing, thereby reducing the number of test cases needed.	*Разбиение на эквивалентные классы*
Boundary value analysis	[ˈbaʊndəri ˈvæljuː əˈnæləsɪs]	Testing the boundaries or extreme values of input ranges to identify potential errors.	*Анализ граничных значений*
Decision table testing	[dɪˈsɪʒən ˈteɪbəl ˈtɛstɪŋ]	A testing technique where test cases are designed based on different combinations of inputs and conditions, often represented in a tabular format.	*Тестирование таблиц принятия решений*
State transition testing	[steɪt trænˈzɪʃən ˈtɛstɪŋ]	Testing the behavior of a system as it transitions from one state to another, typically used for systems with finite states.	*Тестирование переходов и состояний*
Error guessing	[ˈɛrə ˈgɛsɪŋ]	A testing technique based on the tester's intuition and experience to guess where errors might occur and design test cases accordingly.	*Предположение об ошибке*
Pairwise testing	[ˈpɛərˌwaɪz ˈtɛstɪŋ]	A testing technique that generates test cases covering all possible pairs of input parameters while minimizing the number of tests required.	*Попарное тестирование*
Orthogonal array testing	[ɔːˈθɒgənəl ˈɑːrɪ teɪbəl tɛstɪŋ]	A systematic method of generating test cases using orthogonal arrays to efficiently cover various combinations of input parameters.	*Тестирование с использованием ортогональных массивов*
Domain testing	[dəʊˈmeɪn ˈtɛstɪŋ]	Testing focused on specific input or output domains, ensuring that the software behaves correctly within defined boundaries.	*Тестирование области*
User story testing	[juːzər ˈstɔːri ˈtɛstɪŋ]	Testing based on user stories, which describe specific functionality from the perspective of an end user.	*Тестирование пользовательских историй*
Positive testing	[ˈpɒzətɪv ˈtɛstɪŋ]	Testing aimed at verifying that a system behaves as expected under normal or valid conditions.	*Позитивное тестирование*
Negative testing	[ˈnɛgətɪv ˈtɛstɪŋ]	Testing aimed at uncovering errors or unexpected	*Негативное*

		behavior by providing invalid or unexpected inputs to the system.	*тестирование*
Cause-effect graphing	[kɔːz-ɪˈfɛkt ˈgræfɪŋ]	A graphical technique used to represent the relationship between causes and their associated effects, aiding in the design of test cases.	*Построение графа причин-следствий*
Risk-based testing	[rɪsk-beɪst ˈtɛstɪŋ]	Testing focused on areas of a system that are identified as having a higher risk of failure or defects.	*Тестирование на основе рисков*