Chapter 1. Theory of automated testing.

So, what is automated testing?

Automated testing – an analogue of manual functional testing, which is performed by a robot program, not by a human.

In turn.

Automation of software testing – this verification process that includes the examination of the basic functions and test steps as running, initialization, execution, analysis and results, automatically by specialized tools. Consider the example in more detail.

When we develop software, we certainly test it. If we are talking about a function, we can call it with different arguments, and see what it will return to us. Having created a website or a large portal, we open it in a browser, click links and buttons, check that everything is done correctly. We walk through it on pre - written scripts. We conduct various types of testing (functional, smoke, sanity, etc.) This process is called “manual” testing — a person checks the operation of the program. A reasonable question is whether this process can be shifted to the shoulders of robots? It is usually possible, and this is what is called **automated testing**.

- The speed of execution of test cases can be many times and orders of magnitude superior to human capabilities. If you imagine that a person will have to manually reconcile several files of several tens of megabytes each, the estimate of manual execution time becomes frightening: months or even years. At the same time, 36 tests implemented in the framework of smoke testing by automated scripts are performed in less than five seconds and require only one action from the tester — to run the script.

- There is no influence of the human factor in the process of test cases (fatigue, inattention, etc.). let's Continue the example from the previous paragraph: what is the probability that a person will make a mistake, comparing (symbolically) even two ordinary texts of 100 pages each? And if such texts 10 or 20? And the checks have to be repeated over and over again? We can safely say that a person is guaranteed to make a mistake. The automated script is not wrong.

- Automation tools are able to perform test cases, in principle, impossible for a person due to its complexity, speed or other factors. Again, our example of comparing large texts is relevant: we cannot afford to spend years repeatedly performing an extremely complex routine operation in which we are guaranteed to make mistakes. Another excellent example of test cases that are too much for a person is a performance study, in which it is necessary to perform certain actions at a high speed, as well as to fix the values of a wide range of parameters. Can a person, for example, a hundred

times per second to measure and record the amount of RAM occupied by the application? No. But automation script can.

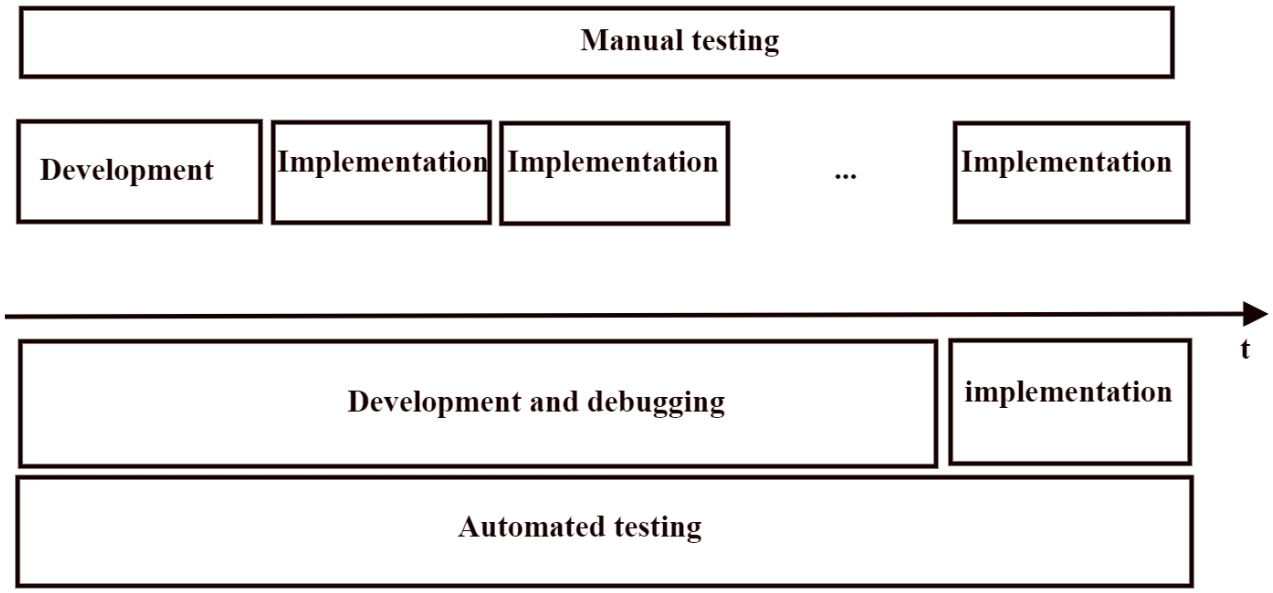
- Automation tools are able to collect, store, analyze, aggregate and present huge amounts of data in a human-readable form. In our smoke-testing example of the "file Converter", the amount of data obtained from the test is small — it can be processed manually. But if you look at real-world design situations, the logs of automated testing systems can take tens of gigabytes for each iteration. It is logical that a person is not able to manually analyze such amounts of data, but a properly configured automation environment will do it itself, providing accurate reports in 2-3 pages, convenient graphs and tables, as well as the ability to dive into details, moving from aggregated data to details, if necessary.

- Automation tools are able to perform low-level actions with the application, operating system, data channels, etc. In one of the previous paragraphs, we mentioned such a task as "a hundred times a second to measure and record the amount of RAM occupied by the application." This task of gathering information about the resources used by the application is a classic example. However, automation can not only collect this information, but also affect the runtime environment of the application or the application itself, emulating typical events (for example, lack of memory or processor time) and fixing the reaction of the application. Even if the tester is qualified enough to perform such operations on his own, he will still need a particular tool — so why not solve this problem immediately at the level of test automation?

So, with the use of automation, we are able to increase the test coverage by:

- execution of test cases, which previously were not worth thinking about.
- multiple repetition of test cases with different input data.
- freeing up time to create new test cases.

But is everything so good with automation test? Unfortunately, not. Very clearly one of the major problems can be represented by figure:



Correlation of development time and execution of test cases in manual and automated testing.

First of all, you should realize that automation does not happen by itself, there is no magic button that solves all problems. Moreover,

a series of serious drawbacks and risks are associated with test automation:

- The need for highly qualified personnel due to the fact that automation is a "project within a project" (with its own requirements, plans, code, etc.). Even if we forget for a moment about the "project within the project", the technical qualification of employees involved in automation, as a rule, should be significantly higher than that of their colleagues involved in manual testing.

- Development and maintenance of both automated test cases and all necessary infrastructure takes a lot of time. The situation is aggravated by the fact that in some cases (with major changes in the project or in the case of errors in the strategy) all the relevant work has to be done again from scratch: in the case of a tangible change in the requirements, the change of the technological domain, the processing of interfaces (both user and software), many test cases become hopelessly outdated and require the creation of anew.

- Automation requires more careful planning and risk management, because otherwise the project can be seriously damaged (see the previous paragraph about the alteration from scratch of all developments).

- Commercial automation tools are significantly expensive, and the available free analogues do not always allow you to effectively solve the tasks. And here again we have to return to the question of errors in planning: if initially a set of technologies and automation tools was chosen incorrectly, it

is necessary not only to redo all the work, but also to buy new automation tools.

- There are a lot of automation tools, which complicates the problem of choosing a particular tool, makes it difficult to plan and define a testing strategy, can entail additional time and financial costs, as well as the need for training or hiring appropriate specialists.

The scope of automation:

First, we look at the list of tasks that automation helps to solve:

• Execution of test cases, unbearable to man.
• Solving routine tasks.
• Speed up test execution.
• Release of human resources for intellectual work.
• Increase test coverage.
• Improvement of the code by increasing the test coverage and the use of special automation techniques.

Testing makes our software more reliable and life easier. But not always. After all, agree, it is better when we ourselves find and fix the error before the release, than when an angry customer or user tells us about the problem.

Firstly, we lose time to correct the defect, sometimes during overtime or on weekends. And, secondly, we are losing business reputation, which negatively affects the business.

Testing is especially useful when developing large applications in a large team, when you can accidentally break some function that

the other person did, and which you did not know. Or, when it is necessary to finalize a previously written complex project.

In large companies, there may be a separate group of people who are engaged only in testing. Usually they are called the testing Department, or Department QA (quality assurance) in this book, I immediately want to separate the concepts of testing and QA.

Testing — the process of product quality assessment, and QA-is the formation of processes that provide high quality software (including development processes, Analytics, documentation).

Basically, when they talk about QA, they mean testing: largely because process control is still not very common in Russia, much is done by intuition.