Orchestrating your Testing Process

Coordinating your manual and automated testing efforts

Joel Montvelisky
joel@practitest.com

Abstract

Due to many historical reasons, most testing and even development organizations, approach their manual and automated testing efforts independently. What's more, when you look closer at these teams, you notice that even within their automation efforts they are using a number of different testing frameworks, running independently and without much thought around coordination, coverage overlaps or functional dependencies.

This approach needs to change. Teams are releasing products faster than ever, and this means that we need to make every testing effort count, including everything from Unit and Integration Tests run by our development teams, Functional and Non Functional automated tests executed by the testing teams, and every manual testing effort encompassing all the Exploratory and Scripted tests run by every member of our teams.

By coordinating the planning, designing, execution, and reporting of our complete testing process we will be able to achieve better visibility and make more accurate decisions faster.

But the road to achieving this goal is not trivial.

This paper will review the objectives of coordinating all your testing efforts, understand common issues and hurdles faced by teams embarking on these efforts and show an approach to coordinate the efforts of your teams in order to generate an orchestrated testing approach.

Biography

Joel Montvelisky is a Co-Founder and Chief Solution Architect at PractiTest. Joel started has been in testing and QA since 1997, working as a tester, QA Manager and Director, and a consultant for companies in Israel, the US and the EU. He is a blogger with the QA Intelligence Blog and is constantly imparting webinars on several testing and Quality Related topics.

In addition, Joel is the founder and Chair of the OnlineTestConf, and the co-founder of the State of Testing survey and report.
1  The testing evolution path is pushing towards more automation

We are all witnesses to the ongoing evolution of the testing practices. They’ve been taking place during these last years and even decades all around us.

There are several factors pushing testing forward, some of them are internal and some external, but all of them point (among other things) to a more extensive and intelligent use of automation and scripting as part of our day-to-day testing tasks.

Among the main factors fueling the evolution of testing we can see:

1. Advances in automation tools - making them more accessible, easier to use, and generating tests are more stable and robust.
2. Changes in the development practices - moving already from Agile towards DevOps and forcing testers to shift our approach to quality and risk in ways that include today concepts such as “testing in production”, “chaos engineering” and more.
3. Changes in the “players” - with more teams integrating testers into Agile Teams, at the same time reducing the number of dedicated testers, and all this while making them both more technical and with roles that can be described at times as coaches or internal testing consultants within their own teams.

The overall effect of these factors, together with some others that are not mentioned here, is the increase both in the number of tests being automated, and in the quality and stability of these automated tests and scripts, in comparison to scripts that used to be written in the past.

Having said that, we are also witnessing how manual testing is not losing its importance, especially as teams are being asked to release products faster, amplifying the need for more focused and almost surgical manual testing efforts by the team prior to releasing our products, and before the automated scripts can be developed.

This new reality means that in order to provide an accurate and complete overview of the product from a testing perspective we need to include inputs from both manual and automated testing efforts.

2  Manual and automated tests are completely different in nature

It is not surprising that manual and automated testing are completely different, but we sometimes forget how these differences can make it difficult for team managers to provide a report based on inputs of these 2 sources of information combined.
Not only are manual and automated testing different in the way they are run, but they are also different in the granularity of the coverage they provide with respect to features.

The nature of the issues found is usually influenced by the type of test that uncovered it, as automated tests tend to be more specific and deeper, while manual tests can be a lot less defined and more extensive.

Finally, there is the aspect of execution. While automated scripts may take minutes (maybe hours) to run, making it seamless for them to have many repetitions, manual tests take a considerable amount of time and human effort to execute and will be run maybe a handful of times during a typical release.

If all this wasn’t enough, we need to understand that the current trend is for organizations to adopt not one automation framework, but between 3 and 7 frameworks in order to cover all the technological and process related needs of the project. Each of these frameworks will vary with regards to all the parameters we mentioned above, and most times will include its own isolated reporting format.

A final but important remark is that multi-product companies will most times give each team the liberty to choose their own tools and processes, almost promising that the standards and metrics used on their local efforts will be uniquely adapted to the team and its constraints.

This makes the job of the team leads and managers close to impossible, when trying to take the outputs of multiple scattered processes, tools and teams and combine them into a coherent and insightful report to assist your company stakeholders in their decision-making process.

3 Test Orchestration Approach for overall coordination of the testing process

The idea behind the Test Orchestration Approach is to plan and manage the testing efforts in a similar way as a Conductor works with an Orchestra, coordinating the different instruments towards a common result while maximising the value provided by each group and type of instrument.

The approach is based on 3 main phases or effort types:

3.1 Master Planning:

1. Definition of the purpose and scope of the overall testing process
2. Agreement on what needs to be tested manually and what should be tested using automation
3. Breaking down the testing needs into groups and if necessary, into atomic tasks
4. Agreement on priorities and estimated timelines
The purpose here is to have a clear definition of the goals of the testing for the project, as well as an agreement on what should be done by what group and based on what priority level. It is obvious that there will be changes during the actual process but having an agreed baseline will make it many times easier to make decisions on the changes being made to the plans.

3.2 On-Going Coordination:

1. Setting up of a communication and coordination process across teams
2. Deployment of a single reporting and visibility framework
3. Integration between the different tools and frameworks (both manual and automated)

The key to this process is to have coordination, integration and transparency. Coordination can be achieved by setting up a Center of Excellence that will meet and make the cross-team decisions. While the integration and transparency will be achieved by having shared and centralized frameworks that will take the results from each team and make it part of the Quality Dashboards and KPIs.

The above-mentioned Center of Excellence will function as a separate team, meeting a limited numbers of times per month or per period, to review specific topics related your testing process in order to make decision or suggestions for the project teams. It should include senior representatives of your teams as well as external architects and project stakeholders, and they will be able to leverage their experience and knowledge for the benefit of the project.

3.3 Value Broadcasting:

1. Delivering value-adding results both locally to the teams, as well as to the centralized dashboards
2. Ensuring information provides value to your business stakeholders
3. Distribution of information via multiple channels for better reach and diffusion.

This third aspect will ensure that this effort is not a one-time-project, by providing value to the rest of the organization, cementing the place of the testing team as the go-to-team when data-driven decisions need to be made.

4 Understand your specific needs and look for coordinated synergies

The approach and the methodology showed above is generic in nature and it can serve as a guideline to follow in order to implement the Test Orchestration process in your organization.

Still, as it is logical to assume, each organization has their own existing structure and process as well as their own challenges and constraints, and this means that as much as we would like to provide a step-by-step guide to follow this would only result in changes that do not provide the desired results and may even be counterproductive in the long run.

The recommendation is to use the points as a generic blueprint to be adapted to your own needs and constraints, in order to bridge between the specific gaps that exist in your organization.
The most important guiding points on the process remain the same, and those are the ones you should keep in mind throughout the process:

1. Coordination is achieved by considering the needs of all the players and providing a solution that is good for most, even if it is not optimal for any of them.
2. Value comes from information and not from the testing actions you are performing, put as many efforts on distributing the knowledge as you in acquiring it.
3. Ideas and innovation will come from the people in the field, a centralized process should not limit individual teams from experimenting and coming up with improvements that can be shared with the rest of the teams.

Finally, just as software development and testing practices have evolved up to our current place, it is only logical they will continue moving forward. It is important to be critical of our methods in order to not only adopt emerging practices but to help develop them as we take part of this journey.