Building Your Dream Team

By Hans Buwalda

When I write and speak about testing, it's usually about methods, techniques, and project experiences. But when I'm actually testing, I've found that the testing team itself is a major factor in a project's success or failure. I want to share some experiences and ideas about building a successful QA team. Remember that circumstances and people differ, so you will have to determine which techniques will work best for your team.

stars
16

Testing is often underestimated as a discipline. In an average project, most attention is given to system requirements and programming. Testing is seen as a supporting activity, and not much effort or money is invested in building or upgrading the testing team. I often encounter testers who have not received training in even the most basic testing techniques. This is unfortunate since we're talking about a small investment that can have a substantial impact on quality and productivity.

Testing is a specialized and challenging discipline. To be effective, testers must have a deep understanding of the system and subject matter under test and should be able to think outside the box in order to find the subtler bugs. Testers also need to be able to work well with others, even under stressful project conditions.

There are three paths to becoming a more valuable member of a test team:

- Learn specialized testing techniques and methods.
- Move to test automation, a more technical but interesting and challenging activity. There is a growing need as more and more organizations move to automation.
- Become a test manager, and take on more responsibility for progress and the results of the testing and test automation activities.

What Is Expected of Testers?

In many projects, I am surprised by the sheer complexity of the tasks and responsibilities of the test team. In today's IT projects it is generally not enough to devise and type in some simple test cases. The testers must understand the system under test—often at a deeper level than even the system developers—and come up with sharp, well-designed, and thoughtful tests that can break a system made by skillful developers.

Testers also need to have "soft skills." Projects seldom play out smoothly, to say the least. Testing activities are generally on the receiving end of disruptions and unforeseen problems, since they are dependent on good, up-to-date specifications; working systems; and suitable test environments. Testers are an important source of quality and progress updates for managers and other stakeholders, including developers and business owners. In that respect, testers are often the bearers of bad news—having to report, for example, that a system is still in bad shape and far from being shippable, even though deadlines have come and gone.

The picture gets even more complex when automation is involved. In many projects, management is interested in automation, which promises to shorten lifecycles, allows for better reuse of testing investments, etc. This means that automation skills become part of the mix. Automating a test is not easy. In fact, IT developers will tell you that it is among the most difficult things to do in IT, particularly because the target software usually has not been designed to allow for automation. This introduces uphill challenges like accessing complex user interface (UI) objects and dealing with subtle timing issues.

Forming a Team

When building a team, the best results are obtained by running testing as a team sport. Don't expect every person to have every skill, but rather form a mixed group of people who understand the issues at hand and can stay on top of them.

The most important recommendation I have about test-team building is: Decide what mix of skills, experiences, and personalities you want before the project starts. I like to think of it as "designing the team." This might seem fairly obvious, but a team lacking key skills will result in unnecessary delays and frustrations down the line. Investigate the generic and specific knowledge that is relevant for the project (e.g., "model-based testing" or "What does a bank teller do?"). Estimate the complexity, political context, and involved risks of the project and environment in which team members will have to work. Then determine the following:

- What skills do I ideally need? (a "Utopia" analysis)
- Which people can I get, given factors like environment, timelines, and budget?
- Should I hire people, and what should their skills and personalities be?
- What skills do the available people currently have?
- Can I train them to supplement what I don't have yet?
- And the critical question: How can I divide the work across team members to get an optimal result?

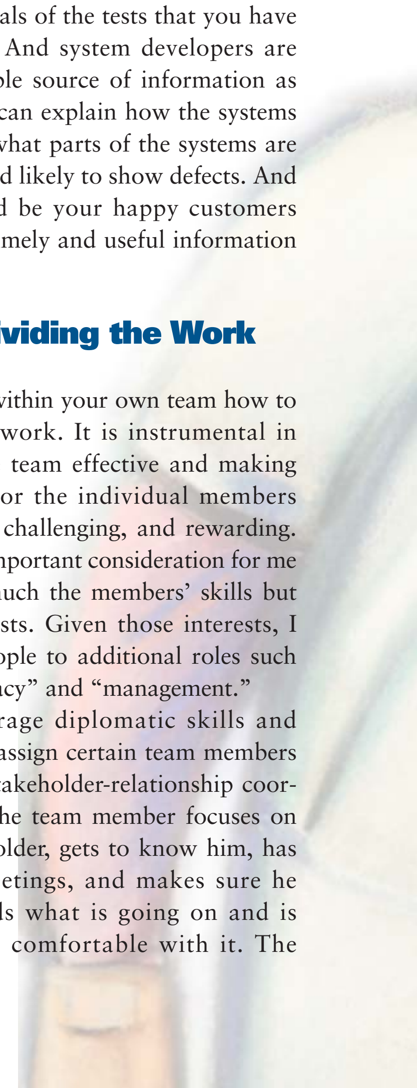
In setting up a test team there are qualitative and quantitative aspects. The qualitative aspect is: Are the knowledge and skills that we will need represented in the team? The key knowledge and skills should also be present in sufficient numbers—the quantitative aspect. As long as the total team size is sufficient for the job at hand, it is often possible to let team members train and coach one another to compensate for any insufficiencies of specific knowledge and skills.

Teaming up extends even beyond the test team. Make sure you have excellent working relationships with other stakeholders in the project. For example, business owners need a short line to supply you with functional knowledge and approvals of the tests that you have developed. And system developers are an invaluable source of information as well. They can explain how the systems work and what parts of the systems are complex and likely to show defects. And they should be your happy customers regarding timely and useful information on defects.

Dividing the Work

Decide within your own team how to divide the work. It is instrumental in making the team effective and making the work for the individual members interesting, challenging, and rewarding. The most important consideration for me is not so much the members' skills but their interests. Given those interests, I appoint people to additional roles such as "diplomacy" and "management."

To leverage diplomatic skills and interests, I assign certain team members roles like stakeholder-relationship coordinators. The team member focuses on one stakeholder, gets to know him, has regular meetings, and makes sure he understands what is going on and is happy and comfortable with it. The



relationship coordinator makes sure the stakeholder is aware of what we expect of him and that it is feasible in the time allotted.

Team members also can be placed in management roles. I commonly ask team members if they are interested in assuming management responsibility, because not everyone aspires to manage. If they do, the test project is a good opportunity to leverage such interest.

Note that this discussion on carrying management responsibilities comes from my view of the difference between a manager and a managed worker. While I don't claim the difference is black and white, I do see an essential difference in responsibility. A manager is responsible for the results of work done by others, so the focus shifts from being responsible for a contribution to being responsible for a result, regardless of how that result is achieved.

If someone on my team wants to take on management responsibility, I will delegate some of my own tasks to him. For example, he can own progress and quality of parts of the project, thus creating an extra job dimension for him and relieving me of some of the project burden. I use the bandwidth that this strategy brings me to anticipate and deal with the unexpected issues in a project, thus obtaining a much higher level of control and more peace of mind.

Many skilled testers, however, are not into diplomacy or management. They want to focus on and be good at the testing work at hand. I try to identify and delegate specific tasks to them to gain maximum results for the project and the people in it. Some tasks you can delegate include:

- Select someone to coordinate the application of the testing and test automation method used in the project. The proper use of a method is an important factor for success.
- Make sure work gets reviewed. One person can take this on, or peers can review each other's work. Make sure that the review and the approval are visible explicitly in the developed test.
- Let someone focus on a particular part of the functionality under test and the specific business know-how that goes with it.

When specializing people, make sure not to foster "guru-ism." Avoid having only one person with extensive and exclusive knowledge of a certain area. I had such an experience in a project, and I found it a very unpleasant surprise to learn late in the game that the person who left the team was the sole owner of knowledge critical to running a large part of the tests and evaluating their results. I now make sure that multiple people share critical system knowledge and that notes are made in the tests explaining what the steps are and why a certain outcome is expected.

Training

There are two ways to increase the skill level of your team: Improve the skills of the current people, and hire additional people.

If you already have good people on your team, some form of training is, in my view, the best choice. My recommendations for effective training are:

- Create and follow a well-structured plan for the training that shows objectives, costs, and timelines.
- Don't stick to a single course, but follow a curriculum that is supplemented with coaching on the job and even periodic assessments of the effects of the training.

Training can be seen as an investment activity. It obviously benefits the people involved, but it should also have clear benefits for the business. The Investors in People Standard, created by the UK government, is a good example of this (see the StickyNotes for more information). It provides a specific framework to make business improvements by systematically investing in individuals. Development plans are agreed upon that will improve employees individually and are driven from a well-defined business perspective.

Following a longer-term program, including multiple training events and follow-up activities with coaching and assessment, allows the subject matter to be put into daily practice, thus increasing the long-term effects of the investment.

When larger groups of people are involved, consider the possibility of staff members—after an initial period of externally obtained training—training newer staff members. This principle, sometimes referred to as "train the trainers," minimizes and leverages training costs over a longer period of time.

Improve your return on investment by allowing the external training organization to study your situation first before giving the training. Many good trainers are capable of tailoring their offerings to your specific requirements: the systems under test, the existing skills and backgrounds of the team members, and other testing requirements.

Other Options to Improve Skills

Training is not the only way to improve team skills, although it can be focused and effective. There are other internal and external resources to consider, each with its own possibilities and limitations.

Testing conferences and public training events offer a variety of topics, both in specific presentations and "tutorial" days with public training classes. These events are less focused than trainings, but they tend to bring new ideas to the organization and to energize and motivate a team.

Before you send one or more team members to a conference, spend some time preparing for the trip. Since there are usually many parallel tracks of presentations and classes, it makes sense to map out in advance which ones to attend. Team members can make notes and discuss their findings with the rest of the team when they return.

One group that is often missing from conferences is managers. Their absence is a sad thing because managers can use conferences as a time to step away from the daily burden and find new ideas and directions. In fact, I make it a priority to send managers, particularly to major conferences.

Another way to improve team skills is self-study. There are many good books, articles, and Web sites out there that contain a wealth of easily accessible information. Magazines, like the one you're reading now, are a low-cost and potentially effective way to increase know-how and awareness. And don't underestimate the value of less-formal channels, such as vendor newsletters and Web sites and blogs from well-known testers.

One of my favorite ways to achieve continuous improvement is internal special interest groups (SIGs). Organizations use SIGs quite a bit for various disciplines such as project management, and they are particularly well suited for testing and test automation. Members can join a SIG and attend regularly organized meetings during which a presentation is given, either by one of the members or an external speaker, and relevant topics are discussed.

You can get an internal SIG started through informal events such as monthly "brown bag" lunches, where you invite members from other teams to share experiences, "war stories," and ideas. If that is successful, you could formalize it into a regularly scheduled, structured event and invite outside speakers. You might even try securing a budget for small, two-hour training sessions on specific topics relevant to your organization.

The focus of a SIG should not be to coordinate operational matters but rather to discuss professional ideas and experiences, thus empowering the members—and through them the organization—in the fascinating field of testing.

Variations on this theme of leveraging internal capabilities are:

- Create and maintain an internal Web site or newsletter (only do this if you are able to keep the information interesting and up to date).
- Organize occasional "visits" in which one group meets with another group or project to discuss experiences and insights. Testing groups are often isolated and thus miss an important means to leverage available company know-how.

For larger organizations, form a central group to coordinate testing efforts. This can easily lead to bureaucracy and competency battles, so take care.

Automation

Many of the principles outlined in this article also apply to test automation. More and more organizations look at automation as a means to leverage and better reuse test investments and shorten lifecycles. However, automation is more difficult than often assumed and places a heavy new burden on the testing team.

The first thing to understand is that test automation is different from testing. Testers who are good at creating test cases and finding bugs aren't necessarily good at writing automation scripts. They don't think like programmers, and you don't want to use them as such. Why make a good tester into a bad programmer? It's better to focus on making good test cases. You don't have to be able to write automation scripts to write good test cases. Automation is a specific and technically oriented discipline that deals with items like user-interface-object accessibility and subtle timing issues, and, in my view, is the realm of specialized engineers.

I see two main ways to embed the automation discipline in the testing organization. The first is to keep it separate from the test teams. Automation can be either outsourced to an external provider or organized as a separate team, potentially part of the development organization. The advantage is that the specialization is taken care of. The potential pitfalls are a lack of necessary interaction with the testers and a potential conflict of priorities. However automation is organized, make sure the automation team does not have a testing task itself. This can lead to the interests of the "other" testing teams being treated secondary, leading to frustration ("We have the need, they have the tools") and a fall back on manual testing.

The second way to embed the automation discipline in the testing organization is to integrate the automation discipline into the test team. In most cases this is my preferred option. It keeps all testing-related activities under one manager and shorter and more effective communication lines with the testers. It also allows for sharing tasks. Even though testing and automation are two different disciplines, it is often possible to let testers help out with automation activities or let the automation engineers assist in test case development.

Remember that testing and test automation goals are seldom achieved by "buying a tool." Using a good methodology and managing the project well are more important—but most of all it is the people involved who can make or break a successful testing effort.

Conclusion

Creating and further developing a successful testing team is more than putting a group of people together and asking them to test something. A testing team has to take on many specialized tasks and responsibilities, often under high project pressures. Understanding the roles that people can play and gathering the right mix of specialties, skills, and interests are the essential first steps toward testing and test automation success. Team members can be assigned tasks and responsibilities—not only testing but also communicating with parties outside the test team and managerial tasks, such as taking ownership of the progress and results of part of the testing project. Resources such as training programs and SIGs can be used to further enhance the capabilities in the team and achieve maximum effectiveness. {end}

Hans Buwalda, now at LogiGear in California (and formerly at LogicaCMG in Europe), is the chief architect for Action Based Testing (ABT). ABT is a comprehensive method based on his ideas for creative and manageable testing and test automation, such as action words and soap opera testing. You can reach Hans at www.logigear.com.

Sticky Notes

For more on the following topics, go to www.StickyMinds.com/bettersoftware

- The Investors in People Standard