Measure Twice, Cut Once: Acceptance Testing

Mitko Mitev

Abstract The challenges it sets are numerous and thus, many project managers face the hard decision of how to organize the process that requires not only technical knowledge but also demands to be able to put themselves in the user's shoes. Being aware of the problems that will be met and relying on the testing performed so far, many decide to skip the need for conducting acceptance testing using both external and internal resources. And what happens next—great loss for the company as the end user is dissatisfied with the software and refuses to use it. The company will face not only profit loss, but what is more important in the modern world: loss of reputation and credibility among partners, competitors and above all—"his/her majesty"—the end user, who is now to judge even harshly as ever, because of all the possibilities he/she has and the ease of switching from one software to another.

That is the reason each project manager must be aware of the main challenges of organizing and conducting the acceptance testing. The benefits from it are numerous and will only lead you and your team to a greater success. Of course, it should be performed correctly and planned really well; otherwise you will have to deal with the losses that follow the release of a lousy product in the eyes of the end user.

Keywords Software testing · Software quality · Software tester · Acceptance testing

1 Acceptance Testing, Part 1: Introduction

Often neglected, one of the most important phases in making sure that the software meets the user requirements is acceptance testing.

The challenges it sets are numerous and, thus, many project managers face the hard decision of how to organize the process that requires not only technical knowledge but also demands to be able to put themselves in the user's shoes. Being

M. Mitev
SEETB, Quality House Bulgaria, Sofia, Bulgaria

aware of the problems that will be met and relying on the testing performed so far, many decide to skip the need for conducting acceptance testing using both external and internal resources. And what happens next—great loss for the company as the end user is dissatisfied with the software and refuses to use it. The company will face not only profit loss, but what is more important in the modern world: loss of reputation and credibility among partners, competitors and above all—“his/her majesty”—the end user, who is now to judge even harshly as ever, because of all the possibilities he/she has and the ease of switching from one software to another.

That is the reason each project manager must be aware of the main challenges of organizing and conducting acceptance testing. The benefits from it are numerous and will only lead you and your team to a greater success. Of course, it should be performed correctly and planned really well; otherwise you will have to deal with the losses that follow the release of a lousy product in the eyes of the end user.

Imagine, you find a severe bug in the acceptance testing phase—what are you supposed to do?

It is a well-known truth that bugs are easy to be fixed at an early stage—not only the bug does not affect the integrated system, but the time and resources; both human resources and financial resources are a lot less compared to fixing a bug later on in the development of the project. Imagine, you find a severe bug in the acceptance testing phase. What are you supposed to do? Ignore it and continue to delivery or try to fix it, but the effect that it will have on the system may postpone the release of the software for months. And why not skip the acceptance testing phase? Because it is the end user who will judge if your product is good or not and will determine its success.

Table 1 shows the status of the IT projects in the last 5 years—CHAOS Report 2015.

Following the notably famous reports, according to the CHAOS Report in 2015, the main reason for bugs are bad requirements—they are not clearly specified, they are incomplete and they are not taken into account seriously. This affects the development of the software, the software test cases created and in the end the conception of the project. Thus, the phase of accepting testing where both the client and the end users are included is the best way to check if all the expectations are met. It is better to postpone the release of the software than release a lousy product. Doing that, in the acceptance phase, you will be able to collect data both from the client (one more time, actually) and from the end user and make everybody involved in the project satisfied. Everyone knows how the business works—only one bad reference can ruin it all, despite having thousands of successful projects behind you.

Table 1 Status of IT projects according to CHAOS Report 2015

	2011 (%)	2012 (%)	2013 (%)	2014 (%)	2015 (%)
Successful	29	27	31	28	29
Challenged	49	56	50	55	52
Failed	22	17	19	17	19

The competitors will not hesitate to mention the bad experience you have had in that single project and gain points before prospective clients that were once yours. In the modern, competitive IT world, such big mistakes as skipping the acceptance test phase are no longer allowed.

But, to make sure that we all need that specific phase, here are some more charts derived from the CHAOS reports.

In the first one, we can see the most common reasons IT projects fail (Table 2).

Then, the second chart shows the factors that make a project successful (Table 3).

Comparing the two tables we can see a lot of familiarities and we can be certain to say that a very important factor for each single project is eliciting correct and not drastically changeable requirements. As we are all aware, good requirements are the basis of acceptance testing as well.

Table 2 The most common reasons IT projects fail

Project impaired factors	% of response
1. Incomplete requirements	13.1
2. Lack of user involvement	12.4
3. Lack of resources	10.6
4. Unrealistic expectations	9.9
5. Lack of executive support	9.3
6. Changing requirements and specifications	8.7
7. Lack of planning	8.1
8. Didn't need it any longer	7.5
9. Lack of IT management	6.2
10. Technology illiteracy	4.3
11. Other	9.9

Table 3 The factors that make a project successful

Project success factors	% of response
1. User involvement	15.9
2. Executive management support	13.9
3. Clear statement of requirements	13.0
4. Proper planning	9.6
5. Realistic expectations	8.2
6. Smaller project milestones	7.7
7. Competent staff	7.2
8. Ownership	5.3
9. Clear vision and objectives	2.9
10. Hard-working, focused staff	2.4
11. Other	13.9

2 Acceptance Testing, Part 2: The Process

How to organize the process so that you make sure that everything goes smoothly? What are the most important steps to execute the “perfect” acceptance testing?

To answer these questions we must start with the definition of acceptance testing as provided by the ISTQB: “formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system. Acceptance testing is also known as User Acceptance Testing (UAT), end user testing, Operational Acceptance Testing (OAT) or field (acceptance) testing”.

2.1 *Types of Acceptance Testing*

There are different types of acceptance testing that should be performed under different circumstances and regulations. For example, the most well known is the *user acceptance testing* where the criteria for “done” are usually created by business customers and expressed in a business domain language. Very popular is also the *operation acceptance testing* where the criteria are defined in terms of functional and non-functional requirements. Also, *contract and regulation acceptance testing* are executed where the criteria are defined by agreements or state laws and regulations. The most well known are the alpha and beta acceptance testing. What do they consist of? Developers of market software products often want to get feedback from potential or existing customers in their market before the software is put for sale commercially. That is when it comes to alpha and beta testing.

2.2 *Alpha Testing*

Alpha testing is performed at the premises of the developing company, not by the developers themselves: usually QAs that were not involved in the project, which provides the certainty of an independent and unaffected testing. This activity is quite often outsourced, which even guarantees greater independence of opinion as you will not harm the reputation and the feelings of your colleagues.

2.3 *Beta Testing*

Beta testing, also known as field testing, is performed by users or potential users at their own location. Many companies, quite often even start-ups consider this

type of testing as a free acceptance testing. You release the software and the customers do the testing job for you—finding and reporting the bugs. This also shows which functionalities are of greatest importance for the end users and where the development efforts should be focused. It sounds great, especially if you lack the budget to perform acceptance testing and you have a tight budget on other types of testing. Yes, the end user may do it for you, but are you sure that every bug they encounter will be reported, are you sure that even the reported bug is clearly described? Think about how much resources you will spend trying to identify a bug poorly described. And, after all, who will use your software if it turns out that your beta release was a really lousy product? Do not ever forget that in the modern world people are communicating very easily and the opinions in the forums for the products really decide what products the mass of the users will download and use. Our advice is to be very careful when making a beta release and making sure you did an alpha testing in advance.

2.4 Main Differences Between Alpha and Beta Testing

To be more precise in distinguishing alpha and beta testing as they are the most common ones, here is a comparative table (Table 4).

3 Acceptance Testing, Part 3: Approaches

While all other types of testing has the initial intent to reveal errors, acceptance testing is actually a crucial step that decides the future of a software, and its outcome provides quality indication for customers to determine whether to accept or reject the software.

Acceptance testing is often considered a validation process, and it does three things:

1. Measures how compliant the system is with business requirements
2. Exposes business logic problems that unit and system testing have missed out since unit and system testing do not focus that much on business logic
3. Provides a means to determine how “done” the system is

The very basic steps to organize an acceptance testing, where managers shall start, are:

1. Define criteria by which the software shall be considered “working”
2. Create a set of acceptance testing test cases
3. Run the tests
4. Record and evaluate

Table 4 Main differences between alpha and beta testing

Alpha test	Beta test
<i>What they do</i>	
Improve the quality of the product and ensure beta readiness	Improve the quality of the product, integrate customer input on the complete product and ensure release
<i>When they happen</i>	
Toward the end of a development process when the product is in a near fully usable state	Just prior to launch, sometimes ending within weeks or even days of final release
<i>How long they last</i>	
Usually very long and see many iterations. It's not uncommon for alpha to last 3–5× the length of beta	Usually only a few weeks (sometimes up to a couple of months) with few major iterations
<i>Who cares about it</i>	
Almost exclusively quality/engineering (bugs, bugs, bugs)	Product marketing, support, docs, quality and engineering—the entire product team
<i>Who participates/tests</i>	
Test Engineers, employees and sometimes “friends and family”; focuses on testing that would emulate 80% of the customers	Tested in the “real world” with “real customers” and the feedback can cover every element of the product
<i>What testers should expect</i>	
Plenty of bugs, crashes, missing docs and features	Some bugs, fewer crashes, most docs, complete features
<i>How they're addressed</i>	
About methodology, efficiency and regiment. A good alpha test sets well-defined benchmarks and measures a product against those benchmarks	About chaos, reality and imagination. Beta tests explore the limits of a product by allowing customers to explore every element of the product in their native environment
<i>When it's over</i>	
You have a decent idea of how a product performs and whether it meets the design criteria and it's “beta ready”	You have a good idea of what your customer thinks about the product and what she/he is likely to experience when they purchase it
<i>What happens next</i>	
Beta test	Release

3.1 Primary Approach to Acceptance Testing

Well, the primary approach to acceptance testing is a big different and that can easily be seen on the pie chart. It is usually done by the QA team or the BA team when it is not skipped or shortened. Quite often, it is not very clear how to set the boundary or the level of the user involvement in it. So, what the pie chart shows us is given in Fig. 1.

Fig. 1 Primary approach to acceptance testing

3.2 Basic Steps to Organize a Well-Done Acceptance Phase

Having seen what the primary approach and being ignorant of it, let's say, you are about to take the basic steps to organize a well-done acceptance phase. Even before taking them, you must make sure that you have in your pocket all the prerequisites needed to start, such as:

1. Business requirements are available and they are not incomplete or ambiguous.
2. The application code is fully developed.
3. All other types of testing such as unit, integration and system testing are completed.
4. There are no show stoppers or high or medium defects in the system integration testing phase.
5. You must have only the so called "cosmetic" errors before acceptance testing.
6. Regression testing should be completed with no major defects.
7. All reported defects are supposed to be fixed and retested.
8. The acceptance testing environment must be ready.
9. Sign-off mail or communication must be evident.

3.3 Approaches to Create Acceptance Test Cases

After you have made sure that your requirements are present, clear and complete, the next step is to start writing the acceptance test cases. There are several approaches to do so, and what is advisable is a mix of them. The approaches that you should

definitely take into account are the test cases that are requirements based (traditional approach), business process/workflow and data driven approach.

Why we advise you to use more than just one approach? Because if you use only the traditional requirements-based approach, then the test cases you create may also carry over the defects of the requirements. In addition to that when incomplete and incorrect requirements are present, then your test cases become incomplete and incorrect. The downside of the business process/workflow test cases is that data-related testing is out of its scope. And the data-driven ones focus heavily on the data and miss out the process and the business side.

Also, never forget that *writing the acceptance test cases is not the last step of system development*. Writing them starts right after the completion and approval of the requirements, data models and application models, and before the development is completed.

The tips we can give you when you start planning the acceptance testing phase are:

1. Identify and involve the key users—they have a deep understanding of the user requirements.
2. Provide not only demo but also a hands-on training of the system to the users.
3. Have the users write their own test cases as well.
4. Ensure that the users will also execute the tests.

More specifically, it is very important to clearly identify and set SMART boundaries of the roles, the type of testing to be executed—in person or self-paced—and the time frames (as you are aware time is never enough to perform a thorough testing), to set the documentation standards and determine the change control process.

3.4 Roles and Responsibilities

The roles are similar to the one in the Scrum team—you should have an Owner/PM who will manage the process and take the final decisions within the team. The Owner will also update the project sponsor on the status. Then, the project sponsor or Business Owner comes. She/he will take care of the requirements and will assist the Owner/PM when testing them. That person will also be solely responsible for the Change Control process. The team resources will actually do the acceptance testing.

4 Acceptance Testing Part 4: Conclusions

4.1 The Execution: What to Avoid When the Acceptance Testing Is Performed?

The execution is to come—it should not be the last step and should be done frequently and also manually when needed. Do not forget to record and evaluate the data, so that you are sure that nothing is missed out. What to avoid when the acceptance testing is performed—you will face numerous pitfalls, the one you should definitely avoid are:

1. Insufficient planning—we are always ready until we are not.
2. Lack of system availability and stability.
3. Lack of resource availability and stability—imagine a massive turnover rate and everybody leaves when the acceptance starts.
4. Poor communications channels—it is no different than any other aspect of the development and the testing process—if you lack the proper communication and the right communication channels, then you are definitely going to fail as there will always be misunderstanding among the team members.
5. Limited record keeping—keep your eyes on the essentials.

A very important aspect that is a real pain is user involvement—this is a highly recommended factor to make your acceptance successful.

4.2 Involving the End Users

Involving end users in the acceptance testing phase is great, but it also sets a few quite important challenges:

1. Users do the acceptance testing in addition to their busy schedules and to avoid that you should have them testing not at the last moment, but as early as possible;
2. As it is the last phase, acceptance testing may be turned to be a “formality” and to avoid that, the users should write their own test cases and let the test alone as they have less or no devotion to the team/project/software;
3. Then comes the challenge to motivate the users to do thorough testing even when they have busy schedules. Well, get to know your users and use different motivational techniques.
4. The worst challenge you may face is the lack of understanding of how the system works—here, the frequently mentioned test cases created by users will help you.

And then, having in mind all the troubles that can be met with the involvement of the end user, you decide to follow the scenario where you entirely rely on the QAs in your company who have worked on the project. The fact is that the QAs

that have tested the software influence the acceptance testing phase too much. That is definitely not good if you want to prove that the system does the things it is supposed to do in the way it is supposed to do them.

4.3 Acceptance Testing Performed by the Internal QA Team

The result of doing the acceptance testing with the QA hired in your company and that were doing the system testing are as follow:

- It only proves that the application works as shown by the previous test stages.
- The coverage is quite small and mainly UI—some even argue that this is what most of the users see, the Pareto principle.
- They do what they do day to day as the corner cases are covered by the Functional/System Testing and is proven correct.
- Acceptance testing is not supposed to find any defects—the things that do not work shall come as change requests, i.e. the primary requirement was at fault.
- Testers should not be involved in acceptance testing as it is a milestone for requirements and their correctness.

4.4 What Are the Main Reasons the Acceptance Testing Phase Often Fails?

Thus, we come to the point to find out what are the main reasons the acceptance testing phase often fails. It is because there is no collaboration and no management buy-in. The other reason is the wrong focus—mostly testers focus on *how* and not *what*. Besides that, acceptance testing is often neglected; it is fully performed by different and sometimes not suitable tools. And then, when the objectives of the team are not aligned and the skill set required is underestimated, there is no possibility to have a successful end to the acceptance testing phase.

Therefore, it is quite understandable to see the fear of acceptance testing when it comes to that point in the testing process. It requires a lot of efforts, a lot of good planning and the ability to be able to adjust quite fast to the new realities. It is also not good to underestimate the requirements of the end users involved in that phase—as most of them will not be very good at IT literacy, you should be careful and patient while explaining the system and the steps for writing their test cases. Yes, that is difficult—you use a set of terms with your team and then the end user comes and is not aware of those terms—you will have to change the way you express yourself and find common words for the terms for the ordinary world. But, the result of that involvement should never be neglected—the ideas the end users can provide and the bugs they can find—usually the one you and your team will definitely have missed out.

Fig. 2 Outsourcing acceptance testing: onsite and offshore tasks

4.5 Best Option for Successful Acceptance Testing

What is the best option to do the acceptance testing phase successful and, at the same time, avoid all the obstacles set before you?

To outsource the acceptance testing to an independent organization that will organize and carry it out for you both with the QAs it has and with end users that they will find and train (Fig. 2).

Yes, despite the model and the strong belief that outsourcing the acceptance testing phase is not a good decision, many nowadays believe the opposite—outsourcing the acceptance testing is one of the great options as you set the requirements and you can guarantee to your client that the results are really independent and show what the opinion of the others of the product is. The outsourcing model works the proper way when both parties are sharing information in a timely manner and that information is correct and not misleading in any way.

4.6 To Outsource Acceptance Testing or Not?

It will not be misleading to say that the acceptance testing phase is not the preferred one to be outsourced as the managers are quite reluctant to lose control; moreover,

they strongly believe that the internal team has a better understanding both of the system and of the requirements of the end user. As other research show, this is not quite true and may even lead to a failure.

A few reasons in support of outsourcing acceptance testing:

1. An external team definitely adds value in terms of completing the test coverage.
2. They have a more objective view of the business scenarios that may occur in that industry.
3. External consultants can help test the performance of the application during peak periods.

Before outsourcing the acceptance testing phase, there are always things to consider. The six main ones are listed below:

1. Establish goals for engaging with the acceptance testing consultants—the vendor should have expertise in the area and also should be engaged quite early in the process.
2. Innovation and customization are key qualities for the vendors—look for vendors with a creative approach to testing.
3. Analyse trends and metrics
4. Encourage cross-functional coordination and interorganizational communication—proper communication and the cooperation between internal and external teams is key to successful outsourcing.
5. Select the right people for the right job.
6. Develop effective tracking and controlling mechanisms—at least at the beginning. After you have established a good relationship with the vendor, you may skip those well-defined and measurable parameters for monitoring and control.

Thus, after carefully planning everything and considering the option of outsourcing, you will have a really good release at the end of the acceptance testing phase. It is not an easy task, but must not be neglected. Neglecting it will definitely cost more than executing the acceptance testing on your own with all the struggles and a good option is outsourcing.

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.