User Acceptance Testing

Finally, Some Validation?

By Trevor Atkins **February 3, 2009**

From the earliest times of political intrigue and assassination, food tasters have "found" employment. Their job was to sample the food and drinks being served to someone else, usually an important person, perhaps even a king or an emperor, who could not afford to completely trust those around them to have their best interests at heart.

The taster would confirm that each meal to be served to this person was of high quality and most critically, did not contain poison and was safe to ingest. The taster would also likely be required to swirl the drink before tasting it, to take samples from different parts of the serving, and to sample some of each serving in combination, trying to cover as many scenarios as possible to mitigate risk of harm to their patron before that august personage took to consuming the meal themselves. In the case of some harmful substance actually being present, it was hoped that the food taster would experience the adverse effects before they could get their chance with the real target. The opportunity to try to rectify the issue would then exist and those found to be at fault identified and punished appropriately.

This direct food tasting is a last-line-of-defense approach to validate that the food and drinks are what they should be; literally one sip away from failing to prevent disaster. However, it is not a very effective approach for dealing with the more complicated situations of slow-acting poisons, accumulative poisons introduced over time, or combinatory poisons brought together from multiple sources. Detection of these poisons obviously cannot rely solely on the food taster to safeguard the king. A more thorough multi-stage evaluation process, involving all the parties working on the assembly, preparation, and evaluation of the meal, would be needed.

WHITEPAPER

What is User Acceptance Testing?

Acceptance testing is both a type of testing and a phase of testing in a given project. As a type of testing, acceptance testing is based on pre-determined criteria that, if passed, allow one project stakeholder to hand-off a deliverable to another stakeholder.

As a phase of testing, acceptance testing can sometimes take place at milestones within the project lifecycle, but most often it is undertaken at the end of a project and can include the following:

- Contract and regulation acceptance testing
- Operational acceptance testing
- Field acceptance testing
- Beta testing
- User acceptance testing

User acceptance testing (UAT) is the one form of acceptance testing that must involve stakeholders outside of the project team; the users. UAT provides a formal means for validating that a new system actually meets the necessary user requirements from the users' or customer's perspective within the users' environment (or as close as possible) before moving the system into production.

UAT is also another one-last-chance to screen for the issues not found and resolved during unit, integration, and system testing where the vast majority of the functional, error, and boundary tests are executed. Of course, when performing acceptance testing there is much more than functionality to evaluate.

> *"Satisfaction with the overall quality of the product and its specific dimensions is usually obtained through various methods of customer surveys. For example, the specific parameters of customer satisfaction in software monitored by IBM include the CUPRIMDSO categories (capability, functionality, usability, performance, reliability, installability, maintainability, documentation / information, service, and overall); for Hewlett-Packard they are FURPS (functionality, usability, reliability, performance, and service)."* – Stephen H. Kan, "Metrics and Models in Software Quality Engineering", 2nd Edition.

However, the central purpose of UAT is to involve the users and business managers in an actual test cycle to help them to gain the necessary confidence that the system will meet their real world business needs.

! There is often confusion regarding the terms 'verification' and 'validation' as they apply to testing and they are frequently used interchangeably or together without distinction.

Barry Boehm provides a simple question for each in "Software Engineering Economics":

Verification: "Are we building the product right?"

Validation: "Are we building the right product?"

UAT is a validation activity rather than a verification activity as the intention is to determine if the system meets the needs of the users in their environment; if it is fit-for-purpose.

Motivations and Challenges of Conducting UAT

User acceptance testing validates the system against the business requirements, uncovers issues that prior testing efforts missed, and is used to evaluate if the system is "done".

The importance of the UAT effort depends on the context, such as the system's purpose, the stakeholders' expectations, and the business environment where:

- Purpose includes how critical the system is to the business, or how critical it is that the system does not fail in production
- Expectations includes that the users will have lower expectations of certain systems and much higher expectations of others

- Business environment includes that getting a system into production sooner can be more important than resolving any remaining issues, no matter how concerning these issues are to some stakeholders

Additionally, the software industry is affected by a number of major trends that have put increased pressure on the final-checking of formalized UAT, including:

- Project outsourcing
- Security accountability
- 3rd party standards compliance

The basic motivation for UAT is to mitigate risk of failure or harm to the business. Depending on the perception of risk or the degree of risk tolerance that the business has for the system, UAT may take one of four main forms as suggested by Michael Bolton in his presentation "User Acceptance Testing" for STAR East 2006:

- **Ceremony:** this is a pro forma ribbon-cutting event. Confidence, or perhaps naiveté, is high. There is nothing left to check, so let's release
- **Demonstration:** a minimal presentation of the system and its functionality working in the customer's environment is given. Everything is expected to work
- **Validation:** concerns of handover and value for payment drive this form of user acceptance test. Testing is performed more so for the purpose of verifying contractual obligations
- **Probing:** there is an elevated concern for failures in the field. Testing is actively trying to uncover problems at this last quality gate before the system goes live in production

It is generally understood that everything cannot be tested as well as it could be if time and money were no object. In the real world of project constraints and concerns of time-to-market, any testing effort needs a strategy that creates an appropriate balance between quality, budget, and schedule.

To accomplish this for user acceptance testing, the contextual challenges faced by the project in regard to this effort must be considered, including the following:

- Lack of a joint understanding up front as to the goals and motivations of the acceptance testing effort
- Lack of agreement early in the project on acceptance criteria with an accompanying plan on how to evaluate the system against those criteria before allowing a given release into production
- Lack of a sufficiently broad understanding by both the project team and customer as to what legal implications there are to accepting the system, or not
- Lack of user involvement until the actual acceptance testing is underway
- The project team creates the acceptance test cases themselves
- Schedule pressures provide the incentive for project team members, users and business managers to pass tests rather than to try to find issues
- Users understand their business but do not understand how to test complex systems
- Tendency to try and re-use test cases or scenarios created for earlier test phases

- Temptation to "throw bodies at it" and have the testing approach consist only of broad participation
- Lack of an understanding as to who the users are, their work environment, and how they perform their tasks
- Lack of using real world data and transactions in the real work environment
- Lack of commitment from one or both of the project team and the customer to participate wholeheartedly in the user acceptance testing effort
- **Starting any kind of testing activity late in the project cycle when there is no longer time or budget to properly address the issues that will be uncovered**

Without due consideration for these challenges and the risks to the business they imply, the context of the user acceptance test cannot be fully understood. Without this understanding, an appropriate plan of action cannot be defined and agreed to early in the project cycle. This forces the parties at the end of a project to work backwards from:

> *"Acceptance testing is any testing done by one party for the purpose of accepting another party's work. It's whatever the acceptor says it is; whatever the key is to open the gate - however secure or ramshackle the lock."* – Michael Bolton, "User Acceptance Testing".

A customer should always do some level of acceptance testing to make sure they are getting value for their money. A software vendor will want some level of acceptance testing to be performed by the customer in order to demonstrate to them that the system delivers what they are paying for. Therefore the customer's users need to be involved in the UAT effort

However, if UAT is left to the end of the project both in terms of planning and execution, you will find yourself, either as a vendor or a customer, unable to complete the project as smoothly or as successfully as you would have liked. Also, this should not be the users' last and only chance to help shape the system that they and their business organization will have to live with.

Example Solution Approach

Until the users are sure that the system meets their business requirements, works as intended and actually helps them accomplish their tasks efficiently and effectively, the project cannot be considered a success.

To avoid falling into the common trap of validating with the users only at the end of the project and consequently most likely failing to succeed with respect to schedule, budget and customer satisfaction, the project team needs to both include and involve the users throughout the project cycle. Working with the users in such a collaborative manner will maximize the use of their knowledge and time to clear the way of potential issues prior to UAT.

> *"It is virtually impossible for a software developer to foresee how the customer will really use a program…When custom software is built for one customer, a series of acceptance tests are conducted to enable the customer to validate all requirements."* – Roger S. Pressman, "Software Engineering: A Practitioner's Approach", Third Edition.

Plan on how you will conduct the UAT from the beginning of the project and plan to cooperate with the users, and bring them into the process of planning and validation from early on.

The following is an outline of some ideas and considerations for getting started in this way.

Decide the Purpose Or Intent of the UAT Effort
Without an upfront agreement on what the motivations and goals of the user acceptance testing effort are to be, the project will only face severe challenges from large volumes of issues of all types uncovered late in the project and the political scrambling to assign blame.

When deciding the purpose of the UAT, make sure to evaluate the initial risk / failure impact analysis versus probable costs and schedule requirements. This will help you to better understand the balance of quality, budget, and schedule that is appropriate for the project.

"Two observations lay the foundation for the enlightened view of testing as an investment. First, like any cost equation in business, we will want to minimize the cost of quality. Second, while it is often cheaper to prevent problems than to repair them, if we must repair problems, internal failures cost less than external failures." – Rex Black, Investing in "Software Testing: The Cost of Software Quality".

Once you have clarified what the purpose of the acceptance test is, it will be much more straightforward within that context to define acceptance criteria, identify the stakeholders that need to be involved, review the estimated budget, and plan the schedule of the UAT.

Identify the Stakeholders
In the case of user acceptance, it would be a limited view to simply consider the end-users of the system as the only stakeholders to involve in the validation effort during the project cycle. Stakeholders can potentially be drawn from the customer's entire organization, such as:

- The members of the business unit for which the system is being built will likely supply the users or their representatives who will want the system to help them do their job and be easy to use
- The customer's IT group who will likely be responsible for supplying infrastructure as well as supporting and maintaining the system after delivery should have a chance to provide input as well
- Business management likewise needs to accept the system as appropriate for the organization's direction in the market and in terms with its goals around efficiency

When a team comprised of representatives from each interested group or department is given a chance to participate in the validation activities and ultimate sign-off of the system, the project will become a much more collaborative effort.

To help identify the stakeholders from the customer's side, first make a list of people or groups who might have an interest in directly reviewing or testing the system, then add those who are affected directly or indirectly by the system, and finally add those that may directly or indirectly have an impact on development or deployment of the system.

! The project team developing or enhancing the system may in fact belong to the business organization's IT department and the customer is a particular business unit to which the users belong.

Some example questions to ask could be:

- Who are the people that have asked for and/or are funding the system?
- Who are the people who have asked for the acceptance testing?
- Who are the people accepting it?
- Who are the people defining the system?

- Who are the people doing the validation reviews and acceptance testing?
- Who is the person that makes the final decision as to acceptance or not?

For each of those people or groups identified above, typically a subset or even a single representative stakeholder will be appointed for the purposes of a specific project or system roll-out. Some example customer stakeholders could include:

- Executive sponsor
- Program/project manager
- Acceptance test manager/coordinator
- Business analysts
- IT infrastructure and maintenance representatives
- Help desk representatives
- User trainer representatives
- Business management representatives
- User archetypes ("Joe the …") representatives

! In his book "How to Break Software" James Whittaker talks about a four-part user model; the human user, the operating system user, the file system user, and the software user (e.g.: APIs). Consider the value of including these when defining and evaluating your system.

For each stakeholder identified it is important to understand:

- What are the needs of their role, the pressures or demands they face, and from whom?
- What are their motivations or goals for being involved in the acceptance effort?
- How will they participate in each validation activity and to what extent (% involvement)?

For a successful delivery of the system that the customer wanted, the project team needs to collaborate and work with each of these stakeholders to elicit what they want from the system, to educate them as to the project cycle processes, to inform them of progress and issues in real-time, and of course define with them how the acceptance activities will be integrated with the project.

Set Clear Expectations and Responsibilities of Acceptance

Now that you know the goals of the UAT and the stakeholders involved, you can work on setting expectations. Examine the various phases of the project cycle to identify the milestone and final deliverables for acceptance, and the acceptance criteria and schedule for each. Then, get agreement from the stakeholders in the context of the purpose of the UAT as to:

- What will be delivered for validation?
- How will it be validated?
- When will the validation happen?
- Who will be doing the validation?
- How will those performing the validation be supported?
- Who will be providing the support?
- What is needed for the final decision to accept or not?
- Who is the final decision-maker?

Plan the Plan

When laying out the plan for user acceptance testing, do so with the intention to have the fastest and most successful UAT possible. This means looking back through the development methodology being used at the project phases and milestones, and deciding where to integrate preparation and problem

prevention activities.

In the context of the above sections, the project team and the customer will need to review and refine the estimated budget, secure and train resources, acquire/schedule and set-up test environment infrastructure, and actively participate within each phase of the project cycle.

! Be aware that as you learn more regarding risks, costs, and other preparation/execution challenges on your project, you may need to revisit the earlier decision as to the purpose/intent of the UAT and make adjustments to the agreement on motivation and expectations.

Each project cycle will pass through a set of common phases one or more times, regardless of development methodology. This provides the additional opportunity to integrate typical verification and validation activities with the tasks and milestones of each stage such that timely involvement in prevention and detection of issues is accomplished (e.g.: early involvement of the test team).

Some examples outside of the usual testing, reviews, and sign-off activities that should utilize the customer's stakeholders include:

- **Business Requirements:** interview the users, view the users work environment and understand what they do and how they do it, identify key user archetypes or personae, include verification tests with each requirement to be validated during this phase by the customer
- **Specification & Design:** involve the users to facilitate a workflow and user centered design approach that will build the solution not just the features
- **Test Planning:** involve the customer in defining the format and content of the user acceptance test scenarios to be executed by the users, leveraging concepts such as checklists, matrices, and even pictures. (Refer to "Communicating Requirements To Testing – You're Going To Be In Pictures!" by Trevor Atkins)
- **Coding:** have representatives from the IT maintenance team participate in code reviews and other tasks that will help them take on their later responsibilities more effectively
- **System Testing:** have user representatives participate with the test team during this phase to try out the user acceptance test scenarios and to take an independent pass at providing usability feedback
- **Defect Management:** during actual user acceptance testing use the help desk to facilitate collection and communication of issues and their resolution, including a daily report to the project team, users, and other necessary stakeholders
- **Quality Assurance:** include a customer representative in this group to gain insight as to expectations from their own IT processes as well as to better make visible the project's challenges and successes to the customer sponsor
- **Configuration Management:** establish a change control board (CCB) with the customer's stakeholders to help manage changes to the project scope, schedule, and budget
- **Risk Management:** this is another area where including a customer representative can be very helpful in gaining insight into the customer's business and IT processes, participating in risk identification, analysis, and mitigation decisions
- **Project Management:** take advantage of the customer's program/project manager to help acquire information, coordinate resources, facilitate change request review/approvals, and identify, collect and analyze key metrics, and report status to the sponsor.

"Industry data suggests that approximately 50 percent of product defects originate in the requirements. Perhaps 80 percent of the rework effort on a development project can be traced to requirements defects. Anything you can do to prevent requirements errors from propagating downstream will save you time and money." – Karl Wiegers, "Inspecting Requirements".

Effectively, you need to invite the customer to be a part of their project team with you, and actively undertake the effort together.

! Include an aspect of informality in the execution of the UAT to complement the formal test scenarios. E.g.: ask users to "test" the system using only a set of guidelines regarding tasks and a deliberately vague understanding of how to use the system.

Conclusion

Next time you are thinking about reducing project costs, achieving higher quality systems, or increasing customer satisfaction, start by looking at the end of the project first. User acceptance testing can be managed as an in-parallel initiative closely integrated with your development methodology, and driving improvements in the same.

Make the customer realize the project team is really their project team and that for the team to succeed (and ultimately save money) they need to participate, and to do so early in the process. It is much easier to address issues before going live with the system, than after it is in production and numerous people are using it and the business is relying on it.

If instead you choose to leave UAT as just the last step in your project cycle, one day you might poison the king, and then where will you be?

About the Author

Trevor Atkins is Principal Consultant with Silverpath Technologies (www.silverpath.com). Most recently he was a Regional Director of Quality Services with UST Global and before that he was a founder and the Vice-President Operations of QA Labs, the largest independent software testing company in Canada.

After obtaining a degree of Applied Science in Electrical Engineering from the University of British Columbia, Trevor has been involved in all aspects of hundreds of successful software projects for the last 12+ years, and is dedicated to the design and improvement of quality processes for use across projects and organizations.

About Silverpath Technologies Inc.

Silverpath Technologies provides results-centric consulting and training services targeted at improving the effectiveness and efficiency of quality and testing activities across the software development lifecycle. Visit http://www.silverpath.com for more information.