A Technical Paper on Best Practices in Crowd Sourced Testing

Contents

Note:

1. This paper is with specific reference to sourcing the crowd for Software Testing. In most places, I have explicitly called this out, but in some places, where I may have just used the words “crowd” or “crowd sourcing”, I still mean the crowd being sourced for testing purposes.
2. The initial sections of this paper have been written without any direct reference to an individual. But as I get into the discussion, especially around sections on “Best Practices in Implementing Crowd Sourced Testing” I have referred to the reader as “you” to make the content more engaging and direct.

I've been reading "The Wisdom of Crowds" by James Surowiecki over the last few weeks and am enjoying every piece of it. Back in July 2011, when I submitted my proposal to Star East to speak on "Best Practices in Crowd Sourced Testing", I was very driven to come and talk to the audience on this new and evolving trend, based on varied applications of it that I had a chance to be a part of. I guess I've been fortunate to gain practical experience in this area which I am now able to additionally validate through all the material I've been reading and professional interactions I've been exposed to, these last 6 months.

So, where does all of this really start and how is it relevant for us in Software Quality Assurance? The premise is that the crowds when leveraged for the right problems and directed with a driven goal often even exceed the "experts" in their performance. We are not so bothered about whether or not they surpass experts, for our discussion around Software Quality. Since the crowds represent a valid base of our end user community for whom software is being developed, their feedback provides an amazing set of inputs to improve product quality, when the right set of practices are adopted. Employing the crowd is not always a straight-forward solution which yields success. Crowd Sourced Testing needs careful forethought, planning and tactful implementation to succeed, which is the core topic of discussion in this technical paper.

To set context and perspective, here is an example from James Surowiecki in his book providing an interesting piece of data about a popular television show. In "Who Wants to be a Millionaire", of the set of life lines available to the participants, apparently the crowds' answer turns out to be right 91% of times compared to "Ask an Expert" which is right 65% of times. Such is the power of the crowds' collective wisdom.

Software development processes and methodologies have evolved lately to accommodate changing user needs and requirements as one of their core drivers for success in the market place. This core driver has brought about several radical changes in how software is developed including:

a. Faster time to market
b. A collective ownership of quality
c. The need to focus on product domain knowledge, understand competing products, creatively emulate end user scenarios and mimic end user environments in test labs

Traditional testing techniques involving a dedicated test team will continue to test and enhance product quality amidst these changing scenarios. The question is, will they alone suffice to ship a quality product? Probably yes; probably no. Rather than analyzing too much on the product test team vs. crowd sourced team, one should look at the value of engaging the crowd and look for means to maximize the Return on Investment (ROI). As we start delving into the implementation aspects of the crowd, let's first look at what a crowd is.

What is a Crowd and Crowd Sourced Testing?

Simply put, for our specific needs of software testing, crowd is a diverse set of people who can be leveraged in testing and providing feedback on a product, because of their end user representation, domain knowledge, subject matter expertise (SME) or software testing experience. Crowd Sourced

Testing is the process of using the crowd and employing a strategy to have them provide formal and informal feedback on the product under test. In my humble opinion, the most value from Crowd Sourced Testing is reaped in the first 3 scenarios than the one on testing experience, because in the first 3, the crowd tester is a more realistic tester than a trained tester. The crowd can be engaged for monetary gains for the work they do; but this is not always the case. Pride of participation, non-monetary rewards, other accolades, sense of satisfaction to quench one's personal or professional aspirations could be other drivers that engage the crowd.

It is often misinterpreted that Crowd Sourced Testing is about engaging / working with a company whose business model is to pool in a crowd of testers, who get paid for valid bugs they report. Well, this is definitely one form of Crowd Sourced Testing, but there are several other more important and creative manifestations to look at, to leverage the crowd's wisdom including:

a. Sourcing relevant people from within one's company although they might not be directly working on the said project
b. Building a pool of end users gradually over time to provide feedback across various phases of product development (be it public beta users, dog-food programs, private betas covering MVPs – Most Valuable Players etc.)
c. Partnering or working with organizations and universities who have domain knowledge and subject matter expertise or are capable of representing the end users

With an idea on possible areas to reach out to the crowd, let's look at the next set of important topics on:

a. Challenges in Crowd Sourced Testing
b. When does the Crowd Succeed?
c. Best Practices – The What, When and How of Crowd Sourced Testing to get a holistic view of what the model has to offer

Challenges in Crowd Sourced Testing:

Given the challenges in product testing, even within the core team, one can imagine the challenges in Crowd Sourced Testing, as the testers are often spread across the globe. Lately, product teams are themselves working in distributed team models either through their own teams distributed across locations or through outsourced vendors that they work with. Some of the core challenges they have around communication, management overhead, stakeholder buy in, keeping everyone in sync on product updates, exist in a crowd sourced team as well, often with a greater magnitude. Some additional challenges inherent in the crowd model are:

a. Ensuring security/confidentiality of the product under development
b. Keeping the crowd motivated and encouraged especially when monetary returns do not exist
c. Analyzing and interpreting the results of the crowd's efforts and tying them back into the product's testing strategy

As a product team that understands the value of the crowd and its associated challenges, there are mitigation strategies and best practices to adopt, that will help reap the model's benefits to the fullest, before which let's take a look at when does a Crowd Sourced Test team succeed?

When does the Crowd Succeed?

Diversity is very important. The more diverse the crowd, the feedback that comes in is invaluable. Let's look at an example here. Say for instance, a social networking application is thrown open to the crowd for feedback. When it is made available to a diverse crowd, across geographies, age groups, culture, gender, infrastructure the more representative their feedback would be which would be very difficult (if not impossible) to get from, within the core product team.

Careful thought needs to be given into whether communication amongst the crowd should be allowed or not. There is no right or wrong answer here and this really depends on each individual product and project. If such open communication will promote faster knowledge sharing and such common knowledge base will elicit better product feedback, it can certainly be encouraged. I've known of several beta discussions in ISV companies such as Microsoft where MVPs get together on regular calls to discuss the product. In some scenarios (such as the social networking application testing presented above), to get the crowd's objective feedback and to keep the communication overhead minimal, the crowd may not be introduced to each other. Both these routes have their own pros and cons that one needs to weigh in before making the decision to introduce the crowd to each other.

In my experience, Crowd Sourced Test efforts are most valuable when the product:

a. Has a global user base
b. May need specific domain knowledge that is difficult to source internally and
c. May need specific user environments that are difficult and complex to set up in the test lab

Let's consider a couple of examples here:

I had attended a Tie Seattle panel discussion in 2011 about the potential of e-learning in which I had asked an executive of a language learning product company, as to how they test for the validity of their content, given the lack of and difficulty in finding language experts. His answer was two-fold. Acknowledging the challenges I raised, he said, on one hand they rely on the experts who created the content in the first place. Secondly and more recently, they have started leveraging the crowd across the globe who not only validate the content but also help validate the content's context specific to the target market. In my opinion, the second is a very smart move, implementing the checks and balances the product needs to succeed in the market place.

Couple of years back in one of my projects at Microsoft, we tested a mobile application specially built for feature phones, that was targeted for the emerging markets like India. We had a team in India dedicated to test the product but if we take a moment here, most software testers in the IT industry are technology savvy people who probably own smart phones. Although they could have added value from a culture context given the geographical presence, they would not have given us a comprehensive view into the end user's needs. With the consent of the product team, we extended the application to our

security guards, cafeteria employees, support personnel who provided very valuable feedback testing on realistic feature phones, signal strengths not to mention the scenarios that a true end user would use the application for. This just called for creativity and proactive thinking, with crowds that we could leverage very easily. What did this give the crowd? A sense of pride and importance that people they look up to, reached out to them to help them do their job.

The last point I want to call out here is something I was recently introduced to, by an executive at Microsoft. This gentleman, Ross Smith, is the Director of Test, for the Microsoft Lync team. He is an avid speaker evangelizing creative and innovative software test techniques across the industry over the last several years and one of his passionate areas is "How Games Transform the Workplace". According to him, a closer look at when employees excel in their work place reveals two kinds of behaviors viz. In-Job Behaviors and Organizational Citizenship Behaviors. He presents an intriguing model that a motivated employee is able to succeed in an assigned task (which may not be directly associated with the project under discussion) when it aligns with his core traits. This refers to the Organizational Citizenship Behaviors where the employee contributes to the project voluntarily. In case of the project that he is directly responsible for (In_Job Behavior), he is able to succeed when he is tasked with challenges that require him to build on his core and unique skills. These are pictorially shown in the diagram below and the two blocks in green are the ones where building games really succeed in transforming the workplace:

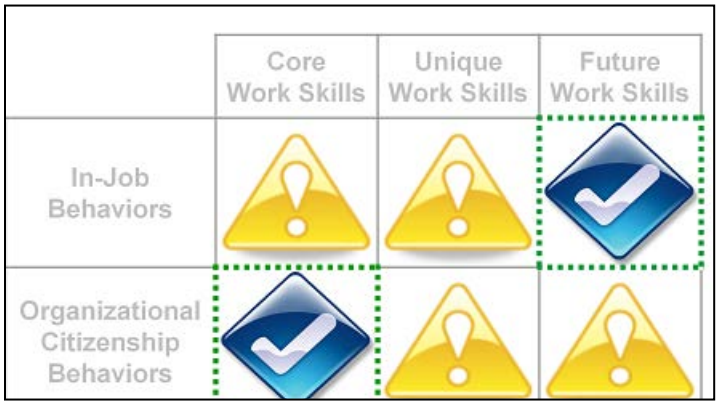

Figure 1: Working successfully with your in-house employees

I discuss this here, because when you leverage the employees in your organization not directly working on your project, you are really using the crowd in testing your product. His model of leveraging the crowd is through employing fun and productivity games to transform the work place, which I've discussed through examples further down in this paper.

To explain this model in slightly more detail:

a. **Core work skills** are skills that a vast majority of the population would have – for e.g. to speak a certain language, to know how to use the internet for regular web browsing

b. **Unique work skills** are skills that an individual possesses, for which he has been tasked with the job at hand. for e.g. experience with a certain test automation tool, experience with a certain programming language
c. **Future work skills** are skills that an individual needs to grow into through formal and informal education, work experience etc. for e.g. learning a new programming language or a tool or a technique

In Ross's model, he talks about how productivity games yield maximum success, when they are built for:

a. In-job behavior working on future work skills
b. Organization citizenship behavior working on core work skills

For the sake of our discussion on crowd, we are more interested in the second scenario of organization citizenship behavior which has been explained through examples further along in this paper; to be fool-proof though, let's look at an example for the in-job behavior as well. Let's take an example of a typist who already types at a certain speed per hour. If a game were to be built for the typist to type at that "speed + x", the game's results are going to be conflicting with the person's core job responsibilities, and thus confusing to the player on what kind of performance to demonstrate. Since, typing is a unique skill for the typist it does not make sense to build productivity games in this space. Rather, if the typist is challenged with a future work skill where he/she does not just type out content but also drafts content in some scenarios, it is a clear avenue for growth; Building a game in this space, will make it both exciting and motivating for the individual with specific outcomes and rewards to expect.

Best Practices in Implementing Crowd Sourced Testing:

As explained earlier, Crowd Sourced Testing is not a hassle free solution. It has its own challenges. However, practice, experience, diligent planning and implementation will definitely help the team succeed in leveraging the crowd. In this section, I want to share best practices from my own experience, what I've heard from practioners in the IT industry, to provide a holistic and readymade set of items which one can then customize to suit his/her own project's needs. I look at best practices from 3 angles – the "What, When and How" of Crowd Sourced Testing.

What to Crowd Source?

Typically user facing features where there is more value in getting the crowd's feedback is where maximum ROI exists. Product design is another area where the crowd is sometimes reached out to. Since design tends to be a sensitive area often with high IP, you may decide to expose it to just a select group (e.g. a private beta) of MVPs who have worked with you in the past. Specialized areas of testing such as compatibility (where users use a diverse set of configurations, platforms, and environments), performance (simulating realistic end user loads using diverse connectivity bandwidths), localization (to accommodate context and culture based verifications) are some golden areas to have the crowds provide feedback on. Content testing, as explained previously, is another area worth using the crowd due to challenges in hiring SMEs.

It is important for a test manager to do a requirements traceability analysis to ensure that between the product team's test efforts and the crowd's areas of focus there is ample test coverage in helping him confidently make the product sign off decision.

In this same context, let's look at what areas do not make sense to crowd source. These are typically:

a. Areas where you suspect a lot of moving pieces requiring in person/close collaboration
b. Products or specific modules that are very IP sensitive and that need to be safeguarded against any leaks
c. Products that have environment dependencies that cannot be simulated outside, as yet

When to Crowd Source?

As you research more in this space, you will soon realize that timing the Crowd Sourced Test effort is very important. You will definitely want your product to work E2E in a reasonable shape (I use this subjective term "reasonable", because you need to decide what E2E means specific to your project and its constraints), unless you are seeking feedback on your design and feature set.

More importantly, your team has to be ready to analyze and work on the crowd's feedback. If this is not the case, you not only have wasted expended effort and the valuable feedback from the crowd but also de-motivated the crowd whose loyalty and association to your product may soon be lost.

Other scenarios include content sources that are ready to be reviewed, absence of an in-house formal test team due to time and/or budget constraints, which necessitate a complete test pass by the crowd. As far as possible it is good to avoid the second scenario, as the test effort here tends to be ad-hoc and informal. ***Crowd Sourced Testing works great as a supplemental test technique rather than as a stand-alone test technique.***

How to Crowd Source?

This section talks about the crowd sourcing process. Firstly, picking the right areas, team and time is important, which we have elaborated in the above sections. These 3 together contribute largely to the effort's success or failure. Internally, you will need to make investments around people, tools and infrastructure to support your crowd. These are required to address the challenges we discussed earlier.

A project manager or a tester who is business savvy, prompt and articulate in communication, who understands the value of end user feedback and who knows the product well to guide the crowd through queries, needs to be dedicated part of full time depending on the scale of the testing effort. To ease the communication process, collaborative tools such as sharepoint, wiki, google docs, open source defect management systems such as BugZilla etc. can be leveraged.

When you think of communication also think about the protocols you want to lay down. They need to encourage the crowd's creativity, need to be open and free flowing at the same time organized, prompt and relevant. Giving this some serious thought upfront at the same time being flexible to accommodate newer needs down the road, definitely calls for the manager to think out-of-box.

If IP is a concern, the team can explore options including a signed NDA, a Virtual Private Cloud to ensure privacy of product and its data. It is also a good idea to get the project's stakeholders' and sponsors' buy in by taking time initially to address their apprehensions, if any. This may even call for a small pilot project demonstrating the value of the crowd. Often times, this is a one-time investment that pays off in the long run, that is worth taking on, even if it means some additional overhead to start off with.

Finally, rather than being bogged by day-to-day tactical issues, it is important to also think about keeping the crowd engaged and motivated in the long run. While new members in the crowd help bring in freshness in the feedback provided, retaining some of your previous crowd has its inherent benefits. Since the crowd has no obligation to stick with you or your product, it is important to look for ways to keep them motivated, challenged, acknowledged and associated with a sense of pride.

Having looked at the Crowd Sourcing model, specific to software testing and also discussed some examples along the way, I would like to conclude the discussion with additional examples and also a section on myths and facts about Crowd Sourced Testing. These examples interestingly show case themselves across the varied crowd sourcing manifestations I presented earlier and are ones that have been tried and tested across leading ISVs like Microsoft as well as an independent test services vendor, QA InfoTech Pvt. Ltd. The bottom line being, once planned and implemented diligently, crowd sourcing can succeed regardless of your scale, business model and project constraints. Some examples I want to present include:

1. **Language Quality Game** – During Windows 7 testing, Microsoft leveraged its diverse set of international employees for localization testing across 36 languages. This was in addition to its core linguistic testing efforts, yielding them a huge success of over 4600 players, 530000+ screens reviewed, 6700+ bugs reported.
2. **Communicate Hope** – Players across Microsoft were leveraged for dog-fooding Office Communicator, with an incentive of contributing to disaster relief agencies through the points they earn
3. **At QA InfoTech,** we work with the Blind Relief Association in India, to bring in the visually challenged crowd for our client's accessibility testing
4. We work with **universities to have professors with SME grade our client's content**

One needs to keep in mind that there are plenty of factors that motivate the crowd. In my experience, when they voluntarily engage in testing a product, the typical motivation factors include:

a. Corporate Social Responsibility (CSR)
b. Sense of pride and acknowledgement
c. Monetary gains

Take time to understand your target crowd's hierarchy of needs. Management theories such as Maslow's hierarchy of needs, Murray's system of needs etc. may come in handy in helping you decide your crowd sourcing engagement plan to truly leverage "The Wisdom of the Crowd".

Crowd Sourced Testing: Myths and Facts:

Myth: Crowd Sourced Testing impacts the core testing team adversely and threatens their positioning in the product team

Fact: Crowd Sourcing is a supplemental test technique not a stand-alone test technique. As presented earlier, there are clear situations where crowd sourcing does not work well. Also given the privacy issues involved, the core product knowledge along with core team's formal testing experience in areas such as test techniques, tools, coverage, automation, test driven development, specialized areas such as performance and security testing, they (be it in-house or an out-sourced team) should not feel threatened; rather they should feel empowered to deliver a better quality product to the end users, with the additional feedback coming in from the crowd.

Myth: Crowd Sourcing is only for software testing. Development is a very technical and specialized area to leverage the crowd for

Fact: Not really. Think of open source software that is developed. Think of Linux for instance. It is a popular example of free and open software collaboration. It is true that it is easier to use the crowd for software testing than for development mainly because the crowd is often the end user base for the product and can provide realistic feedback without needing technical knowledge. That said, most of the best practices and approaches presented in this paper are extendable to Crowd Sourced Development as well. Several companies such as TopCoder.com have crowd sourced development model as their core business driver

Myth: The management overhead is significantly higher in Crowd Sourced Testing. Given the short project deadlines, we do not have the time or resources to manage a crowd sourced test team

Fact: It is true that the management overhead is slightly more in case of crowd sourced test teams when compared to a centralized test team. However, given the global product distribution effects lately, there are several team engagement models such as global virtual teams (both outsourced and internally sourced) that are becoming inevitable. The moment you have distributed teams, you will need to set aside some bandwidth to ensure team collaboration which can otherwise easily go astray given the time zone, culture differences and lack of association with the core team. So given how the situation is changing even in case of core product testing, the Crowd Sourced Test effort does not significantly add any more overhead. Also remember, the crowd once chosen correctly, is a bunch of smart people. I don't mean to say they are experts; in-fact we do not want them to be experts; they are a diverse crowd and as potential end users they are quite self-sufficient and independent in evaluating the product once you give them the basic guidance and inputs.

That brings us to the end of this discussion, which has hopefully given you a holistic view into Crowd Sourced Testing. With the evolutions in the IT industry, this is a test technique which is here to stay for several more years. If you have already thought about this or have implemented and are reaping the benefits of Crowd Sourced Testing, I am sure this discussion would have provided additional reassurance and some valid examples for you to try. If you are an entrant in this space, I am hoping this paper has given you enough insights and scenarios to encourage you to try this in your organization in one of the

many manifestations presented here. For any questions, feedback, comments, I can be reached at rajini.padmanaban@qainfotech.net. Thanks for your time in reading through this material. I am glad I could get this done and am looking forward to presenting this to the Star East audience in April 2012.

About Rajini P:

As Director of Engagement, Rajini Padmanaban, the author of this paper, leads the engagement and relationship management for some of QA InfoTech's largest and most strategic accounts. She has more than ten years of professional experience, primarily in the software quality assurance space. Rajini actively advocates software quality assurance through evangelistic activities including blogging on test trends, technologies and best practices, providing insights on software testing to analyst firms such as Gartner, IDC. Her official blogs are available at: http://www.qainfotech.com/blog/. She lives in Seattle, WA and can be reached at Rajini.padmanaban@qainfotech.net