

Big Visible Testing

Claire Moss
Atlanta, GA, USA
claire@aclairefication.com

Abstract—Software testing is a key part of software development, so it's important that the testers be as effective as possible. Especially in fast-feedback agile teams, software testers must respond to their teammates' need for information with readily available, easy-to-understand testing materials. The novel twist of turning user experience personas inward to focus on the product team members provides deep insights into both what decisions team members need to make and how to present testing information to them to support those decisions. Having established personas as the context for the testing work, testers are poised to spring into action, planning, estimating, and executing both exploratory testing and user-facing automation. Testers then surface the results of this work to the product team in a way that helps them to address concerns in a timely fashion. Big visible charts display testing for co-located teams in a particularly effective form. Testers experiment with a variety of big visible representations throughout the life of the team, retaining these charts only as long as they provide value for decision-making and moving on to other forms as the team learns more about satisfying their information-gathering needs through testing. User personas produce a fresh perspective on product team members to help testers to focus on the value big visible testing artifacts provide to the individuals on a software development team.

Keywords—software testing, agile testing, big visible charts, user personas

I. TEAM BUILDING

I had just taken a new job with a new company. The team I was hoping to serve was quite different from those I'd helped in the past. Now my eight-week-old cross-functional Scrum team included a product manager (PM), a UX designer, and two developers - one a self-professed agilista, or advocate of the Agile Manifesto. I wondered whether the developer's emphasis on writing unit tests to test-drive his code and his limited experience with software testers devoted to his product team would be a point of friction. I'd never seen a UX designer before and didn't even know what user experience was. I had limited experience working for a larger company with a portfolio of products to manage and so didn't have much insight into the work of product management, except that this PM had been doing the testing up until that point. Together, the lead developer, UX designer, and the PM formed a Product Owner (PO) team rather than the single person PO that my ScrumMaster training described.

When I started working with this new team, I thought I was well versed in the art of agile software testing. After all, my development teams have always delivered features in

two-week sprints. My testing activities included reviewing requirements or user stories before the iteration planning meetings to assemble a list of questions and test ideas that I would use to approach the proposed work. I participated in a review before code completion that allowed for some exploratory testing, brief and informal though it may have been at times. In the past couple of years, I also planned and coded test automation. I looked forward to learning more.

Now that we had a fully staffed product team, we decided to revisit our team norms and collaboratively developed a Definition of Done document that we posted on the wall. We had defined the results we want for each story with an emphasis on quality, so the testing information needed to support this value. Three important points in our definition were:

- Team (developers & tester) reviews functional and unit tests are adequate
- Exploratory testing questions are triaged
- Defects are triaged and all urgent defects are fixed and verified

You may notice that my product team wasn't interested in the overall picture of the testing process in the way my testing manager is. They wanted more short-term feedback from testing so that they can do something about it within the sprint. Clearly, I needed to study these new roles more closely: the unfamiliar, the reoriented, and the self-sufficient. And so I began my transformation from a pseudo-agile tester to a true agile tester. Rather than sitting apart from the software developers in my own quality engineering department, I sat in the same room as the other employees from this mix of disciplines who now formed my product delivery team.

II. TRADITIONAL USER PERSONAS

At this point, I was asking myself how I would fit in and looking for new approaches to adding value that could apply to my testing when I found myself fascinated with this mysterious discipline of user experience that had never before been part of my world. We had discussions about the tools that UX folks use to frame their work and I learned the persona approach on the job from two of my teammates. When I walked in the door, my first UX designer had already posted our team's first Big Visible Chart containing several positive personas to represent the customers we would target and one negative persona to represent customers we would not consider to be primary users of our software, potentially even detractors. His personas were based on research conducted before the first sprint's kickoff and helped to

shape how we thought about the value our software product would deliver when we shipped. It answered the question "Who cares?" quite nicely for me - and I really needed that since I had not encountered people doing the work of our customers in my day-to-day life. Discovering these personas kindled a professional awakening for me as a tester, better equipping me to advocate for users.

Having such a clear focus for my work sharply focused my usual testing activities. Story acceptance criteria describe the value of the work for a particular persona. Testing priority relates to the target personas, not just the technical risks and considerations of the work. Testers are in the information business, so I craved a way to ground my work in solid research. As testers, we draw upon a variety of resources including the experience and perspectives of fellow testers and product team members, heuristics, and product history to reach the goal of delivering a product the customer values, focusing especially on the quality aspects of that value. However, our work is never as fruitful as when it centers on our users.

III. AD HOC USER PERSONAS

When our team makeup shifted and another UX designer rotated in, she led a session on designing ad hoc personas for an area of the software that was coming up. The team gathered in front of whiteboards and formed some guesses about what our users would be like based on our own experiences and what we knew of the industry. In some ways, this was the most difficult work we took on because we didn't know what motivated these people and how our software solution could help them to achieve their goals. We named our personas, discussed the perspective each advocated, and got a feel for these companions on our journey.

These ad hoc personas became hypotheses for our usability research. Through UX-facilitated hands-on usability sessions, we gathered evidence that allowed us to judge the accuracy of our guesses. When I participated in those usability testing sessions as a logger, I felt much more connected to the users - the people on the other end of our current software revision or prototype - and appreciated the value of testing with customers early in the process. We discarded one ad hoc persona entirely when usability testing revealed that it was not valid. We refined the remaining ad hoc personas to incorporate our learning of the aggregate needs of our users into more formal personas. This gave us clearer targets for user stories and shaping the future of the product we were building.

IV. BREAKTHROUGH

As our pilot release progressed, we had deeper discussions about how testing serves the product team and I gained insight into their various disciplines, ways of thinking, and preferred ways to communicate. We iterated and I came to see each of my teammates as quite distinct customers of my work. I began to think of them in the way they think of their customers, applying the lessons they taught me to themselves. The clouds parted; a ray of sunlight shone down; I shouted, "Eureka!" and developed user

personas for my teammates corresponding to what value each of them emphasized. I began by focusing on the goals and greatest concerns for each of these roles. Part of the value of user personas is that they can be proposed ad hoc and evaluated through research. I began with a guess based on my past experience and tested that against my current context.

Since this was my first attempt at writing user personas, I decided to start with a template based on the persona workshop our UX designer had facilitated for our team. A user persona is a representation of the goals and behavior of a researched group of users. I made aggregate profiles since I didn't want to focus only on the aspects of these roles that I currently observed but to include my previous experience. As a seasoned software tester who had worked alongside an eXtreme Programming development team and then in a smaller agile team with an iterative process, I had developed working relationships with a variety of business people, software developers, and testers. I wanted to incorporate things I'd learned from those professionals that might point out concerns my current team members had not yet voiced.

Before I could lose my nerve, I presented my persona work to my user experience (UX) designer and explained what I was thinking. I'd been working on this document for a few days, trying to wrap my head around the examples I had seen at work and to incorporate the follow-up reading I'd done online. I recounted how our product's user personas for customers shaped my understanding of user stories and forged ahead into explaining my experiment: making user personas for my agile software team members as a guide for presenting software testing information. As she read over the profiles, she laughed at the details that felt familiar and was very supportive of my approach. We had a great conversation about what I thought I understood about UX and how I was trying to grok this unfamiliar approach. We agreed that testing and UX had a powerful and natural symbiosis for our team.

My teammate personas became targets for me to judge how well my presentation of testing hit the mark. The other members of my product delivery team were from a mix of disciplines but did not have a background in software testing as I know it. I needed to translate my testing information product into a format that would appeal to them. Since I was looking for ways to make testing thinking a part of their day-to-day activities, I proposed these "user" profiles to uncover their motivation. I wanted testing to be a natural fit with their usual responsibilities so that they would get the most benefit from the testing information I provided - and to encourage them to participate directly in the testing activities. With abbreviated time between activities and short feedback loops, I needed to gradually increase the visibility of testing activities through defect backlog organization, exploratory test charter management, and paired exploratory testing with both testers and non-testers.

Let's trace my learning curve of adjusting to their needs through the various experiments I have completed in this context. I wanted to meet them where they were and to collaborate on the testing reporting they needed so that they could schedule, design, and implement stories. My first

attempt to educate my teammates about testing was to create a low-tech defect dashboard or - as I call it - the Bug Board. I started with defects because my teammates already had some understanding of the relationship between software development and that form of testing feedback. I organized them into groupings of those Just Discovered, Big Baddies, Retest (suspected fixed), and Backlog, as in Figure 1. Imagine my delight the morning I arrived at work to find my Bug Board sticky notes rearranged: it had been the focus of fruitful discussion among the members of my Product Owner team, that is the folks making the decisions about what to do, who had prioritized the bug backlog on my board.



Figure 1. Bug Board – Iteration 1

V. PRODUCT TEAM PERSONAS

Recognizing the inherent tensions between my teammates' distinct and various needs, I started working out how to fine-tune the testing format with the only other person on my team who had been tasked with manual testing: my product manager. I realized that he was a secondary persona - not the person I would ultimately optimize my Big Visible Charts to serve and who would be mostly satisfied with the solution for my primary persona - but he was familiar and that made it easier to get started. My first persona was Ben the business advocate, who needs well-rounded information for making business decisions, hence the Product Owner team approach. Since he is calendar-driven, he needs to efficiently answer the most essential questions as described below:

Ben the Business Advocate



Figure 2. Persona 1 – Ben the Business Advocate

Context: Business Analyst background, head of the Product Owner team, CEO of the product, needs well-rounded information for making business decisions

A. About

- Moderate computer skills (word processing, presentations, financial analysis, sales & marketing)
- Subject matter expertise – but from the company's point-of-view (e.g. market segmentation)
- Cares about return on investment
- Daily/weekly/sprintly consumption of test info in standups - and perhaps pairing with the tester
- Collaborates with others in his role for portfolio management of software suite
- Problem-solver, pragmatist, realist
- Older & congenial, has season tickets for a local sports team
- “Chicken” (in the Scrum sense of the word)

B. Feature opportunities

- Severity and priority inform decision on whether and when to address defects
- ROI of testing (e.g. time spent vs. severity of problems found) – possible bias toward numeric measures
- Problems found on or off the beaten path? Obstructing the essential business value? - Find out myself while executing a story's acceptance criteria?

C. Product characteristics

- Ease of reference/efficient gathering of information

For Ben, I rearranged the Bug Board to group bugs by Severity - a familiar characteristic that we were already using in our selected bug tracking tool to describe impact on the user - and discussed that with him, aiming for ease of reference. We revised the representation, as in Figure 3, so that he could efficiently gather information and at a glance know when to follow up with questions to elicit well-rounded information for making business decisions. He wanted to know whether to address defects and when we needed to complete them. When our critical and highest severity buckets were empty, he thinks, "Let's get a return on our investment. Ship it!"

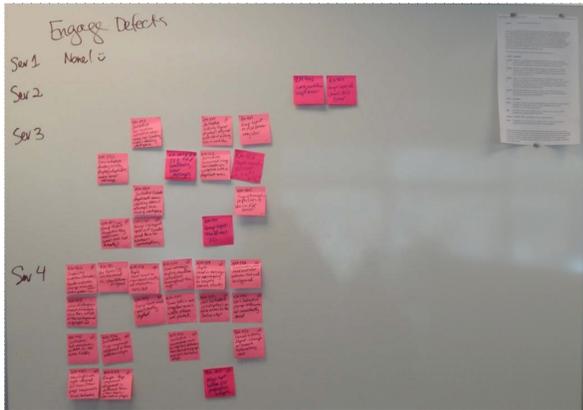


Figure 3. Bug Board – Iteration 2 – Severity Buckets

While this Bug Board format immediately helped us to focus on our “big baddies,” it was a pretty narrow focus and we quickly resolved our worst offending bugs, leaving large ambiguous groupings of lower impact problems that weren’t as meaningful. So I moved on to looking at the other team members to see whether there was an opportunity for improvement.

Although Ben is the CEO of the product, his is not the only perspective to consider. There is an inherent tension between this business motivation and what drives my second persona Ursula, the user experience designer. She balances the team by bringing a sharp focus on the users, whom she knows deeply through broad customer contact, whether through usability testing or contextual inquiry. She sees the system through their eyes - and frequently prefers a visual representation of the impact on the user.

Ursula is a "doer" since she designs the user experience. In Scrum terms, she is a “pig,” or someone totally committed to the project and accountable for its outcome. In UX terms, she is a primary persona: I want to optimize my quality information for her use in problem resolution and her “needs and goals can be completely and happily satisfied by a single interface without disenfranchising any of the other personas” [1]. She takes immediate action on what she sees in my Big Visible Charts. Since Ursula is also a judicious perfectionist with a focus on consistency, she and I have a great relationship and collaborate closely. I use her designs and user research to feed my test design and she uses my granular test output in the larger context of the product.



Figure 4. Persona 2 – Ursula the User Advocate

Context: research (& formal methods) background, usability testing, Product Owner team member, wardrobe remixer/retro fashion/recycled clothing

A. About

- Wants to resolve pain points user will frequently encounter
- Moderate computer skills (visual production, CSS & markup for web apps)
- Doesn't know users' subject matter but expert in harvesting information from user interaction/market research
- Perfectionist/has a focus on consistency?
- Daily/weekly/sprintly consumption of test info in standups and pairing - with tester or developer
- Collaborates with others in her UX role for design review
- Idealist, young & hip, “granola”
- “Pig” (in the Scrum sense of the word)

B. Feature opportunities

- Screenshots of presenting symptoms
- Steps to reproduce problems to see them herself
- Visual representation of bug backlog
- Clustering problems, especially grouping by screen/page but could also indicate affected components of the system or areas of focus

C. Product characteristics

- Ease of learning
- Efficient translation to action (e.g. design changes)
- Accuracy matters

For Ursula, I organized bugs around a web application site map I had already built for brainstorming exploratory test charters. Ursula consumes defects more in clusters than singly, so we revised the Bug Board accordingly to form something more like a site map. I enhanced the Bug Board even more by focusing on efficient translation to immediate action she can take to resolve discovered problems. Ursula evaluates how bugs impact a user's day-to-day life with the software, so I emphasized user scenarios in the problem descriptions to make them more meaningful. Ursula can use this as an input to iterate on the user experience, which encompasses more than just graphic design extending to interactions and patterns of behavior. In turn, I can provide feedback and critique mockups or prototypes, asking helpful questions.



Figure 5. Bug Board – Iteration 3 – Site Map

We maintained this one format, the site map Bug Board in Figure 5, for a long time since it provided deeper insight into our biggest problem areas - and not just for UX! This single Bug Board provided rich information to the whole product team.

While everyone on the team is conversant with software from our own usage, no one has as deep an understanding as Todd the Technologist - sometimes he has to explain his recursion jokes for the rest of us. As the principal "doer" on the team, he is my highest priority customer. Although Ursula has the chops to get into the user interface code, Todd is the one to keep things running smoothly behind the scenes. Test-driven development and automated checks are his first line of defense against bugs, so he may respond with puzzlement to the presence of a bug backlog. Let's translate it for him.

Context: resolves issues reported, advocates TDD (test-driven development) and automation as primary quality drivers

A. About

- Expert computer user (probably builds his own machines or at least cares deeply about their specifications)
- Probably not user-facing/industry subject-matter expert

Todd the Technical Advocate

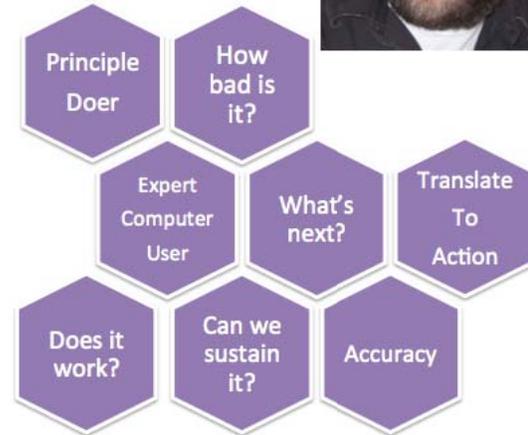


Figure 6. Persona 3 – Todd the Technical Advocate

- Analyzes own code with unit and automated integration testing
- Discovers areas of fragile code/problem concentration through daily/weekly/sprintly consumption of test info in standup, pairing with tester, bug reports, exploratory testing session logs, paired programming real-time code review
- Consumes testing information with some puzzlement/amazement at problems sneaking past TDD/pairing/automation
- Seasoned professional but not necessarily older, probably curses the code more than the testers (which is impressive!)
- Single income earner, loves to geek out in his own way (not yours!), board gamer who has read strategy guides
- "Pig" (in the Scrum sense of the word)

B. Feature opportunities

- Scripted smoke test that covers acceptance criteria
- Test ideas provided before coding begins
- Test heuristics for unit and integration tests
- Given-when-then format for automated tests (checks) at unit and integration levels
- Scenario scripts for user-facing automation (whether at business layer or in UI)

C. Product characteristics

- Efficient translation to action
- Ease of learning
- Reliability (though comfortable with ambiguity)

- Differentiate between symptom and cause
- Environment for clustering defects (e.g. operating system, platform, browser)
- Affects build/affects version to pin down the point in time of the bug's origin
- Cause of defect (after investigation), components in defect reports
- Associating defects with backlog stories coming up in that area

First, we tried scheduling a time box of hours to pay down the technical debt encompassed by the bugs, focusing our most urgent concerns since Todd cares about identifying the highest priority work so he can attack it. We moved the sticky notes from the separate Bug Board to our time-boxed Bug Bucket on the sprint board (i.e. bottom lane of the sprint board in Figure 7). However, Todd is not interested only in an expedient solution. He wants to keep the product sustainable, so he may report issues for the backlog that incorporate technical debt, which may relate to clustering of defects.

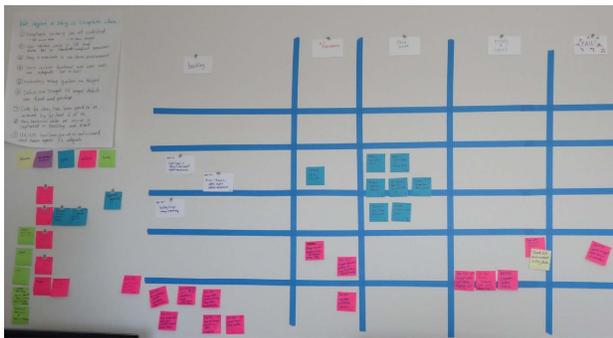


Figure 7. Bug Board – Iteration 4 – Time-boxed Bug Fixing

While that seemed to work at first, since we were reducing the quantity of bugs in our backlog, we realized that pulling in the highest priority bugs didn't always make sense when it was not closely related to the other work completed during the sprint. Ursula isn't the only one who likes a good backstory for bugs, so we shifted to linking bugs to stories already scheduled for sprints since they represented some of our technical debt but were less independent than stories in our backlog. The bugs became story tasks rather than a separate lane on our sprint board. We also included UX tasks such as design into the sprint stories. After all, Todd needs to know the whole picture so he can incorporate refactoring and bug fixing in the estimate along with the feature enhancement that together produce value for the user. He also likes preventive measures such as a scripted smoke test that covers a story's acceptance criteria - just another way for him to know that a story is done.

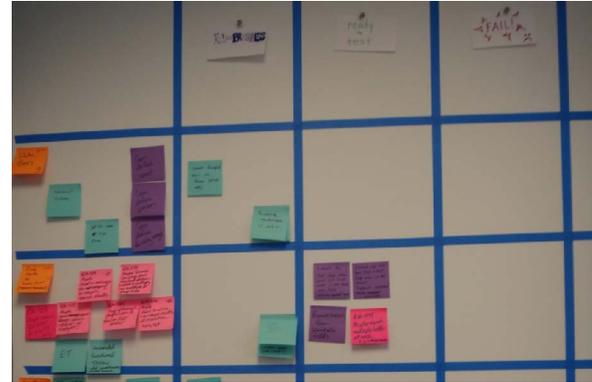


Figure 8. Bug Board – Iteration 5 – Bugs as Story Tasks

VI. GO OR NO GO

That brings us full circle to Ben who makes the ultimate judgment on whether the story is done when he runs the application and evaluates the result against the acceptance criteria. Although Ben (as a Scrum chicken) may be a secondary persona for me - one I want to accommodate but not as directly as my primaries - he benefits from all the testing information along the way since he has confidence when he accepts a story. Our sprint board informs Ben about the current status of our sprint's story progress, including defect resolution and discovery.

VII. BEYOND BUGS

Once the whole team understood defects better, we began breaking down the single testing task for a story into more focused exploratory testing charters so that everyone could see the progress of testing over time and what charters uncovered the found defects. When we found ourselves running up against our story work-in-progress limit, every member of the team could contribute to exploratory testing to some degree and our product's users reaped the benefit of the common understanding. After all, while I may be in the information business as a tester, it is all ultimately in service of our users.

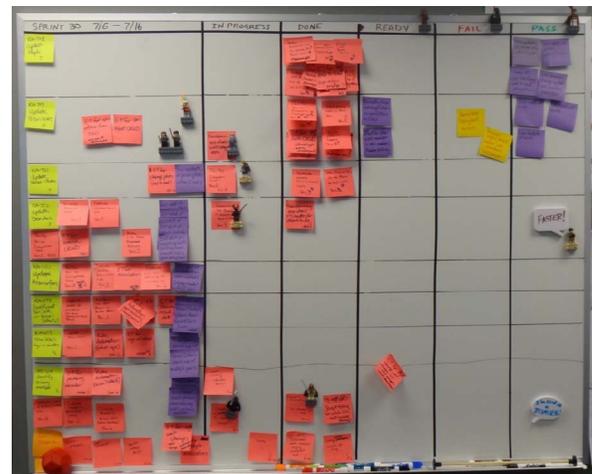


Figure 9. Bug Board – Iteration 6 – Testing within Stories

In the end, the sprint board became a representation of our integrated work and a kind of team building rather than just an information radiator. Team member personas jump-started our progress and kept us focused on working well together. It has been a learning curve for all of us and I hope that you benefit from our experience.

ACKNOWLEDGMENT

Claire thanks the Daxko Atlanta team Will Sansbury, Patrick Boudreaux, Jace Bennett, Steve Guthrie, Astrid Paris, Jason Benton, and James Barrow for their wonderful collaboration and participation in these testing experiments.

REFERENCES

- [1] A. Cooper, *About Face 2.0: The Essentials of Interaction Design*. Wiley, 2003.

ADDITIONAL READING

- [2] A. Cockburn. (2008, June 19). Information radiator. [Online]. Available: <http://alistair.cockburn.us/Information+radiator>
- [3] R. Jeffries. (2004, October 20). Big Visible Charts. [Online]. Available: <http://xprogramming.com/articles/bigvisiblecharts/>
- [4] L. Crispin. (2011, April 26). The Whole Team Approach In Practice. [Online]. Available: <http://lisacrispin.com/2011/04/26/the-whole-team-approach-in-practice/>
- [5] T. Ottinger. (2010, October 8). Heatmap: The new hotness. [Online]. Available: <http://agileotter.blogspot.com/2010/10/heatmap-new-hotness.html>
- [6] A. Savoia. (2004, April 1). eXtreme Feedback For Software Development. [Online]. Available: <http://www.developertesting.com/archives/month200404/20040401-eXtremeFeedbackForSoftwareDevelopment.html>
- [7] M. Bolton. (2012, February 8). A Sticky Situation. [Online]. Available: <http://test.techwell.com/articles/membersub/sticky-situation>
- [8] M. Webb. (2012, April 13). Exploratory testing in Agile environments. [Online]. Available: <http://itknowledgeexchange.techtarget.com/software-quality/exploratory-testing-in-agile-environments/>