Agile Testing Without Automation

John Duarte, Eric Thompson

john.duarte@puppet.com, erict@puppet.com

Abstract

Most research on Agile Testing and QA have requirements on highly automated testing and an Agile or Scrum project-management structure.

How can we iterate towards a more Agile testing process, with all the benefits that entails, when some of the common requirements are missing or undesirable in the near-term?

Drive quality as a core principal through communication and collaboration with QA test "consultants" embedded with development teams: We discuss real-world results from Puppet on feature turn-around time and relevant defect rates

Derive transparency on upcoming features and requirements through quality user stories and acceptance criteria, BDD or otherwise

Risk analysis driven by all teams, particularly Product, QA and Development: We discuss the importance of risk-relative testing, and why it's important in some cases to test less, rather than more.

Whatever is prioritized to test should be transparent to all groups and stakeholders: Everyone knows communication is important, but how can we agree upon the most effective details that need constant discussion with effortless transfer? We discuss methods and motivation on communication: Instances of success and opportunities for improvement at Puppet are presented.

Biography

John Duarte is a QA Engineer at Puppet in Portland, Oregon.

Eric Thompson has been an Electircal Engineer, Software Engineer, and Quality Engineer for 15 years, in organizations of all sizes, leading teams of engineers with varying degrees of expertise and backgrounds.

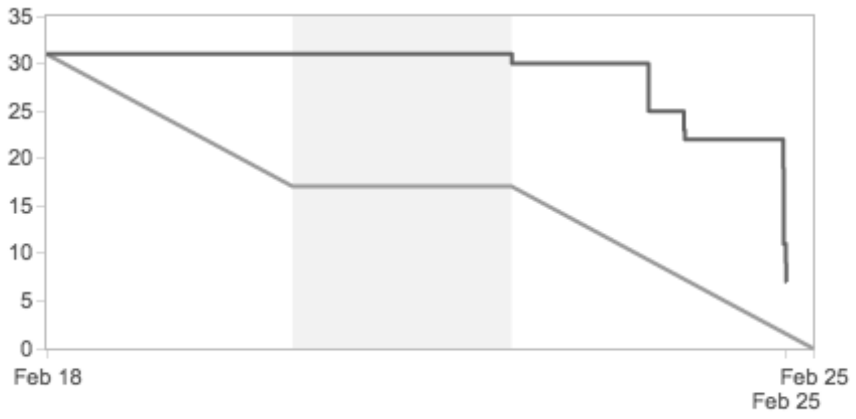
1 Introduction

In the fall of 2014 at Puppet, our QA organization was struggling to transition from waterfall to Agile testing practices. Engineering was practicing Scrum, but QA's involvement was sandwiched at either end of the development cycle. At the beginning of the cycle we were struggling to gather project requirements in order to write test plans. At the end of the cycle, we were frantically trying to validate functionality after it had been implemented. This type of practice is often referred to as *agilefall*.

In the winter of 2014, Puppet adopted an embedded model where each Scrum team had assigned QA resources. This model went a long way toward getting QA out from under the agilefall, but it posed new challenges.

The time to execute went way up for two primary reasons: QA resources were spread thinner by the embedded model and the Scrum teams assumed it was the duty of QA to test *All The Things*. Tickets needed QA approval to close and were being passed to QA late in the sprint without enough time for proper validation. This meant that a high number of committed story points rolled over, which created an undesirable trend in the burn-down chart.

1.1 Poor Burn-Down



2 Risk Driven QA

The purpose of Agile methodologies is to provide value to the customer as quickly as possible. The burn-down metrics were telling us that we were not delivering on that goal.

QA at Puppet needed to re-assess how we could re-align ourselves with the customer goal of delivering valuable features in a timely manner.

2.1 Analysis

In order to increase velocity without sacrificing quality, QA adopted a risk analysis driven approach to testing.

All tickets are evaluated for the risk that they pose to the product and the customer. We prefer to perform this analysis collaboratively with the developers during the sprint planning meetings. This process also forces the team to make sure that acceptance criteria are clearly defined, since the assessment cannot be performed without it. This analysis is transparently communicated to the stakeholders and their input is brought to bear in the final assessment.

Each ticket is evaluated based on the severity and probability of the risk that it poses. These assessments are expressed as simple *High, Medium, or Low* values. Reasons for these selections are added to provide the stakeholders context as to why a particular assessment was given.

The risk values for the severity and probability are then combined into an overall risk assessment.

2.1.1 Overall Risk Matrix

The following matrix provides a guideline for how the two risk vectors are combined at Puppet. It is only a guide and there are occasions where additional information will allow a QA professional to apply a diverging combined value. The matrix accomplishes several goals:

1. It provides a consistent heuristic for all of the QA personnel to apply when calculating the overall risk of an issue.

2. Establishes a clear protocol that allow QA and development professionals to quickly apply when discussing concerns about an issue.

3. Encourages documenting a combined value that deviates from the matrix. Although this is not common, there are certainly occasions where the a value will deviate from the guidelines. For example, a security vulnerability may pose a severity level that is so high that it commands the highest level of risk even if its probability is low.

	Low Probability	**Medium Probability**	**High Probability**
Low Severity	Low Risk	Low Risk	Medium Risk
Medium Severity	Low Risk	Medium Risk	High Risk
High Severity	Medium Risk	High Risk	High Risk

2.2 QA Engagement Protocol

The value of the overall risk assessment translates into a clear engagement strategy by QA:

High

High risk tickets must have automated tests associated with them in order to validate them against the broadest compatibility matrix and to protect against future regression.

Medium

Medium risk tickets will receive manual validation on a sub-set of platforms. Automated regression tests are not expected.

Low

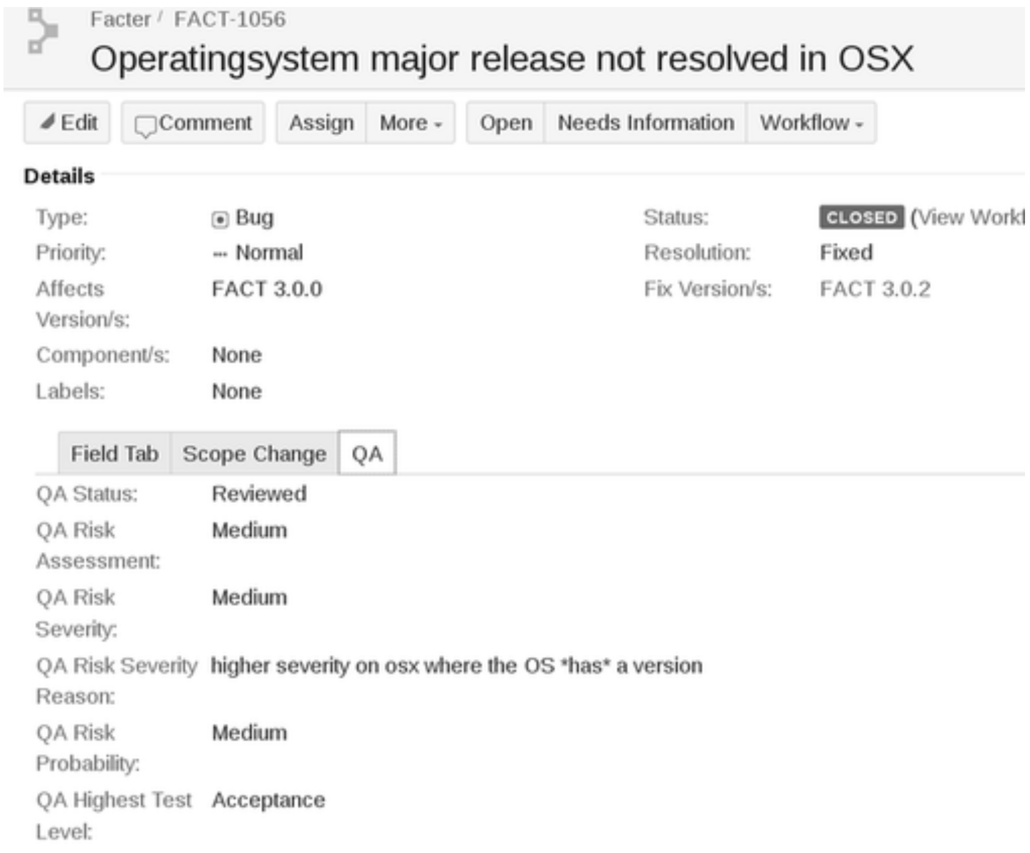
Low risk tickets receive no further QA attention.

In all cases, Developers are responsible for providing adequate unit tests.

2.3 Transparency

All of the risk assessment data is captured in each ticket in our issue ticketing system. We have placed it in a QA tab so that it is easy to find, but is out of the way when not needed. This transparency allows stakeholders to easily infer the level of QA commitment with regards to ticket validation. Any disagreement or lack of clarity can easily be discussed within the comments on the ticket.

2.3.1 Custom QA Fields in our ticketing system



Operatingsystem major release not resolved in OSX

Details

Type:	Bug	Status:	CLOSED (View Workl
Priority:	Normal	Resolution:	Fixed
Affects Version/s:	FACT 3.0.0	Fix Version/s:	FACT 3.0.2
Component/s:	None		
Labels:	None		

QA Status:	Reviewed
QA Risk Assessment:	Medium
QA Risk Severity:	Medium
QA Risk Severity Reason:	higher severity on osx where the OS *has* a version
QA Risk Probability:	Medium
QA Highest Test Level:	Acceptance

3 Results

3.1 Less Automation

Since the QA engagement protocol for the risk matrix only mandates that automation be applied to testing of *High* risk tickets, we are writing fewer automated tests. This has several benefits:

Our people have a more robust understanding of what product risks are covered by unit tests and do not need higher level testing.

The automated tests we write focus on features and regression candidates that have the highest value to the customer.

The cost of our automated test suites is not growing at the same rate as our product code bases.

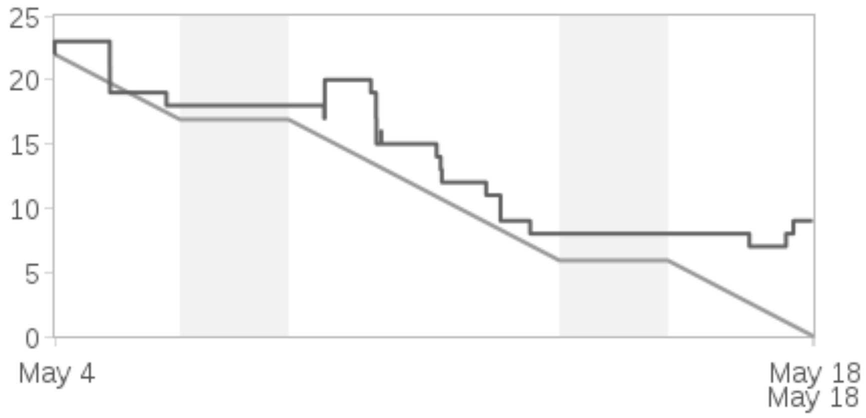
3.2 Relationship

This work-flow has created a healthy relationship with stakeholders, especially developers. QA will openly discuss the validation required for an issue with developers. Clear acceptance criteria are defined. Based on these criteria, it may be determined that the issue is adequately covered by the unit tests written by the engineers. This means that even if a ticket has been risk assessed as *High*, the unit coverage may remove the need for QA to write more automation.

Teams have made this an explicit part of their process which has been very successful. Reviewing tests with developers and determining at which level each test should be automated definitely helped avoid duplication of effort, and ensured that we were getting sufficient coverage of the most important scenarios.

Developers have organized their work to deliver code to QA earlier in the sprint so that both parties can continuously move tickets forward. In the event that QA cannot validate all the tickets by end of sprint then the developers will assist in ticket validation. This has resulted in better burn-down trends in sprints and more predictable velocities for the Scrum teams over time.

3.2.1 Improved Burn-Down rate



3.3 Focus on Higher Value

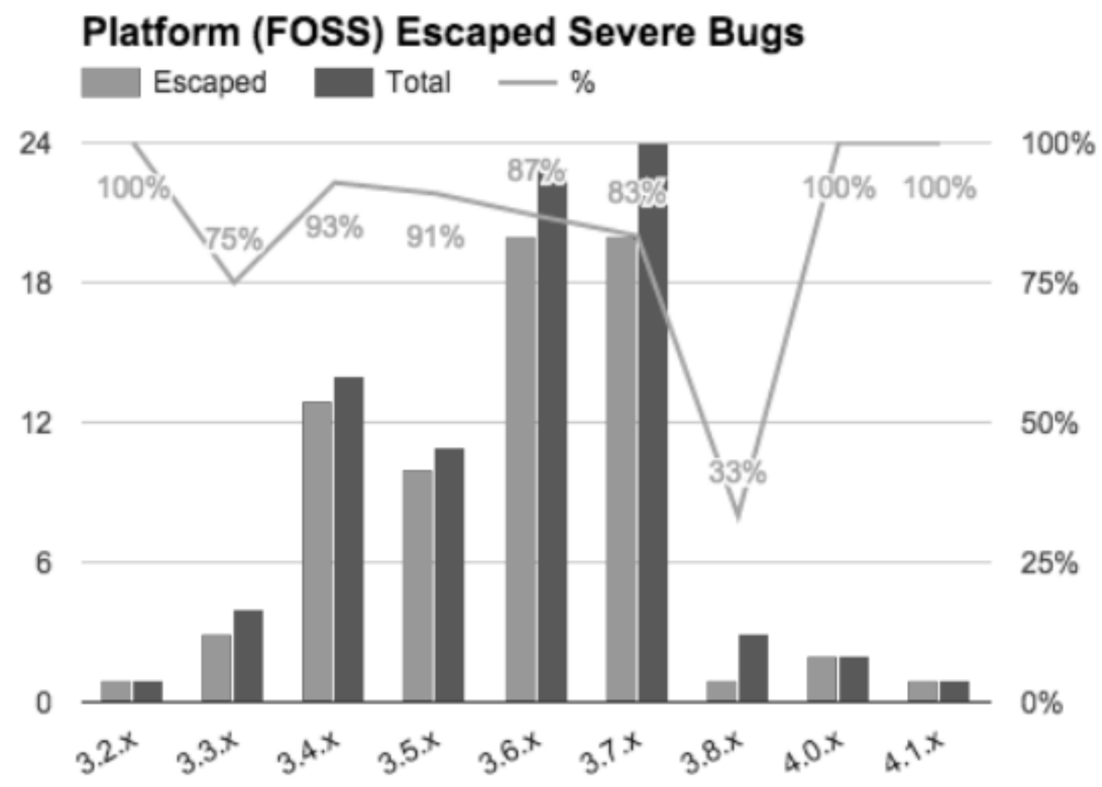
The risk driven method has allowed QA to focus on high value efforts.

QA collaborates on user stories for new features. QA works with product owners and developers in order to capture relevant user stories and define the expectations for error conditions.

Each ticket receives acceptance criteria to minimize ambiguity. Having all of this information centered in the ticketing system allows all of the stakeholders to contribute.

Using a risk driven approach has also resulted in a dramatic reduction in our defect rate when measured against severe and critical issues that have made their way into shipped product.

3.3.1 Defect rate



4 Conclusion

Applying a risk driven approach to testing will allow you to focus on the issues that have the greatest impact on your product and its users. Doing so in a way that is transparent to the organization, and fosters the input of all the stakeholders, throughout the process will result in a better product. It will also result in better relationships between QA and the other groups within the organization.