

STC 2011: Saga of Unsung Heroes

Raj Kamal, rajkamal@microsoft.com, Microsoft, Hyderabad,

Gunjan Jain, gunjain@microsoft.com, Microsoft, Hyderabad

This paper is written by the Testers, for the Tester. The Testers, who save millions of dollars by finding bugs, still stay unheard & unsung. These are the ordinary men doing extraordinary deeds by living in a software world that poses several challenges making the 'Quality' as a job tougher than ever.

This paper is a tribute to all the testers in this world doing a thankless job, those who have spent long hours & many years of their life with a strong belief that every bug they find makes the product better. This paper tries to address #3 major challenges of their (our) life and proposes some fresh solutions to very old problems based on our learning that can reinstate the same belief in new generations of testers

Audience & Takeaways:

Learning can be instrumental for the new generation of testers to position themselves better when found in these common yet difficult situations and can be worthwhile for the experienced testers & leads to think of more elegant ways to handle the same.

The three challenges which every tester must have faced at some point of time in the midst of their career are:



Challenge: Can we Sign-Off?

The Inevitable & Recurring Ghost

Expectations from Test:

As the release date starts approaching, all eyes set on you (tester) to say the anticipated answer i.e. "YES". Even when the product is not yet ready to be delivered (unstable & sometime not even well tested)

Challenge:

There is no such thing as "NO" as on-time delivery often takes priority over Quality. The final accountability of the QUALITY is forced on the TESTERS INSTINCT or GUT FEELING where the role of metrics like *(no of open bugs, test cases not executed/failed)* diminishes. If anything goes wrong, we know who to blame for signing-off.

Dare if you say "NO", all impossible solutions are proposed & imposed which in brief compromise some or the other aspects of the quality

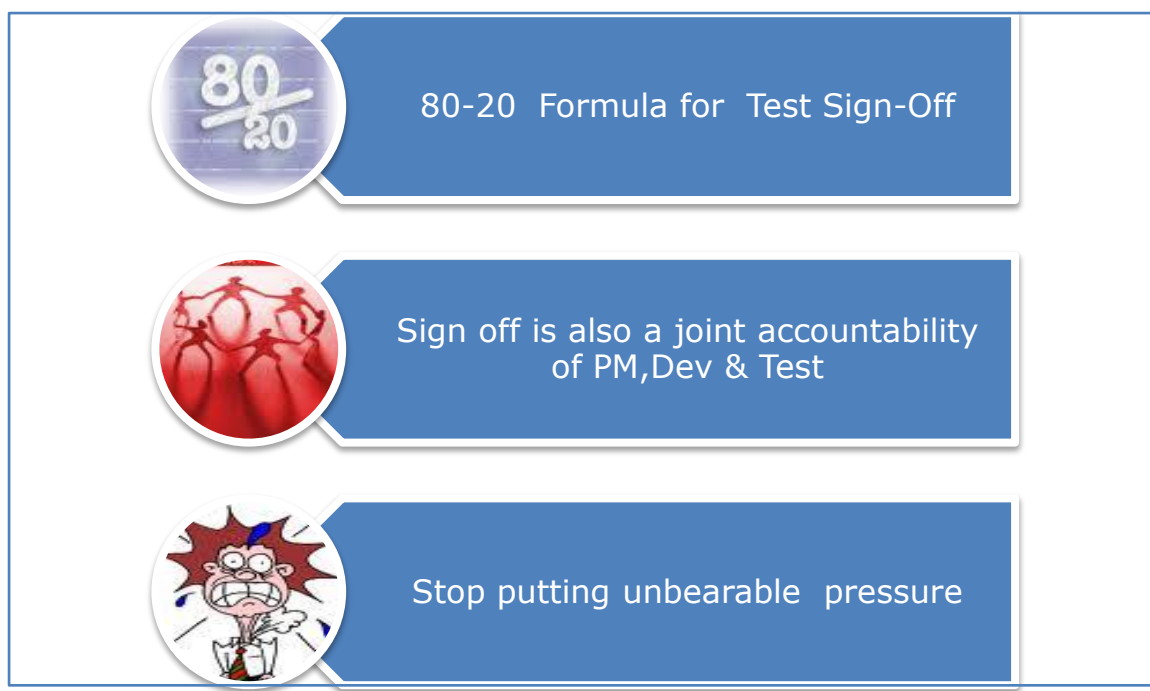
#1: Automate the suite and save time when you have 4 days left for sign-off

#2; Run just P1 test cases. Call known issues in the release notes. We will give patches for P2/P3 issues but won't move the timeline

#3: Stretch or Take help from other test teams who barely know the domain/application

When in dilemma, in order to meet the timelines, the 'quality accountability' is forced on the test team where team's gut feeling / confidence generally overrides the supporting metrics & when things go wrong post SIT, Test teams are often blamed for poor quality and being not rigid

Thing we can do differently



The infographic consists of three blue callout boxes arranged vertically. Each box has a circular icon on the left and text on the right. The first box has an icon of the fraction 80/20 and the text '80-20 Formula for Test Sign-Off'. The second box has an icon of three people holding hands in a circle and the text 'Sign off is also a joint accountability of PM, Dev & Test'. The third box has an icon of a stressed cartoon character and the text 'Stop putting unbearable pressure'.

1) 80-20 Formula for Test Sign-Off

One thing which is missing now is a formula or a structured way to assess objectively if the product is ready to be SIGNED OFF. We are definitely putting a lot on stake and risking things by relying on subjective methods like Tester's confidence as a major factor.

80-20 Formula for test sign-off states:

80 % of the decision should be based on the data

+

20 % based on the Tester's Emotional Quotient

i) 80 % (Intelligent Data Decision)

*= No. of Open Bugs X Weightage based on Severity & Priority +
No. of not tested features X Weightage based on Severity +
Priority*

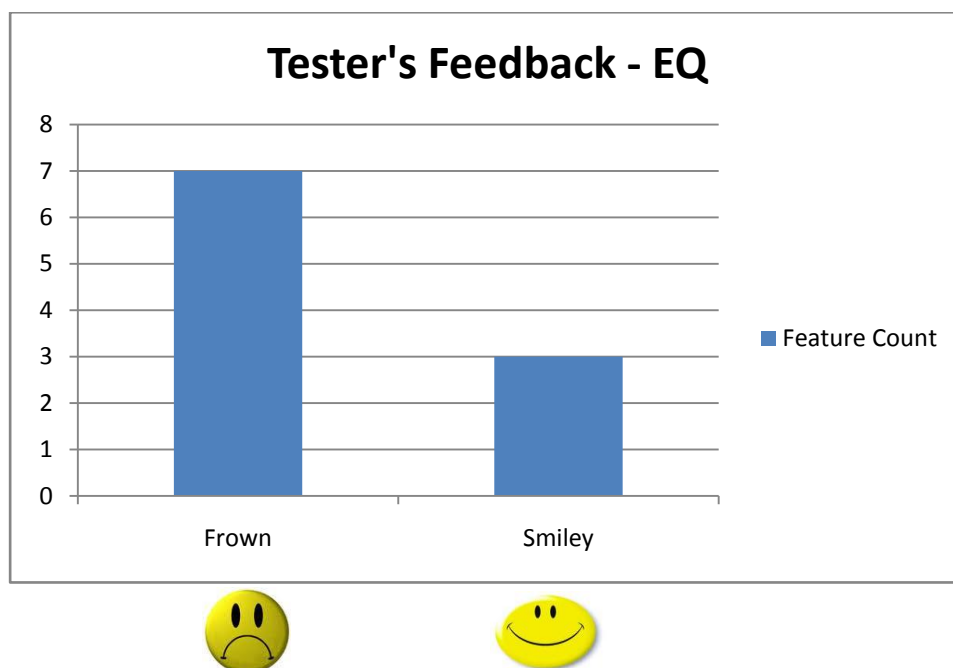
ii) 20 % (Human Decision) – Emotional Quotient

This method suggests introducing "feedback" method (as used by Office 2010 team for Dogfooding) where TESTERS

and other internal audience can send a SMILEY  or

FROWN  for each feature

A graph can be depicted before test sign-off to see the overall experience of the testers as users. *For ex. If they send frown for 7 out of 10 features (either due to performance, usability, and stability issues) then, Tester feel that it's not ready for shipping.*

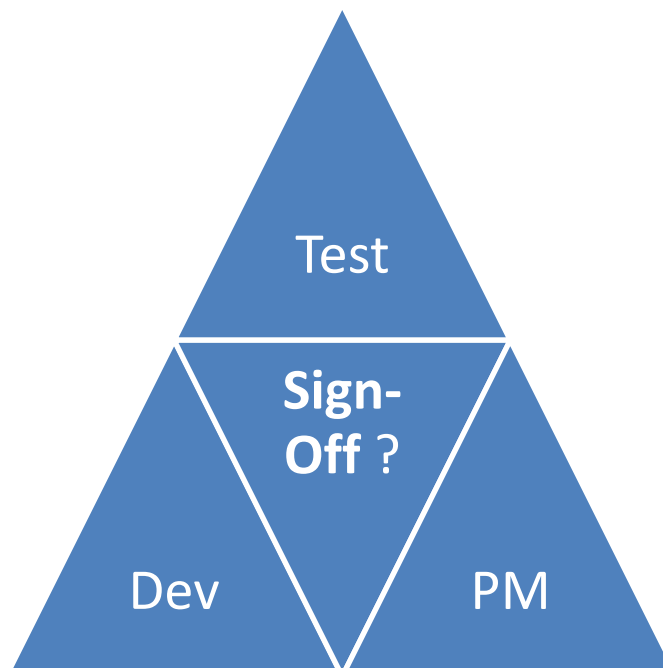


2) Sign off is also a joint accountability of PM, Dev & Test

As Quality is everyone's responsibility, so does "Sign-Off"

If it's true then why only Test Signs off on the quality when product is delivered, we can only expect the high quality when Sign-Off is a shared accountability between PM, Dev and Test.

Each discipline knows the best about their artefacts and should contribute to overall decision of "Go/No-Go live decision"



3) Stop putting unbearable pressure

We understand that tracking dates (planned vs. actual schedule) is important but putting unnecessary pressure by asking the same question "Can we sign off" everyday will make your testers nervous and vulnerable to failure.

If we want to realize the maximum potential of the test team, its key that they are not petrified, terrorized & pressurized by being subjected to this very question too many times. It becomes counter-productive otherwise

It often works in other direction where either the test team say YES because we Indians are not so good at saying NO to when it comes to elders/seniors OR

They start underperforming and doing silly mistakes due to unbearable pressure.



Challenge: Don't shoot the Messenger

Software Testers don't break software; it's broken when we get it.

Test is often considered the show-spoilers (party poopers). They are often stereotyped as the messenger of the bad news by finding bugs and perceived as pessimist. We all know to err is human; to find the errors requires a tester.

Dev, PMs and Leads many times forget that Testers job is to expect the failure and not to prove that it's working fine to please stakeholders when actually it's not.

The above challenge results into:

- i) **No country for 'Testers'** - Fresher (New talent) don't see testing as a long term career opportunity. Senior testers seek opportunities to move into Dev / PM as they start thinking testing is inferior to Dev, PM and other disciplines.
- ii) **Bugs leak to UAT/Production:** When bugs are seen as an entity which should not have happened in the first place, testers start feeling that they will get penalised if for instance they find a bug which should have been found in 'Inspection' phase or 'the humiliation/rejection from Dev for finding a bug that not coming from requirement'

Expectations from Test:

Test team is expected to find all the bugs right at the defect injection phase (be it requirement, design, coding etc).

For instance: if requirement bug is detected in SIT Phase, test team is blasted for being ineffective in inspection.

Challenge:

Ground reality is that when the test team find the bugs they are not appreciated instead blamed for delivering a bad news by Dev, PM, Solution Delivery and Leads. It won't be a surprise that you might hear PM saying "Give me good news"? Or "Tell me its working fine" and when Test team finds a feature working, PM and Leads beam positively and say "That's what I wanted to hear"

Affects the morale of the test team esp. the young testers & ultimately kills the passion in them to find the bugs. Test teams start feeling like 'Second class citizens' who are doing a job which is not welcome by everyone.

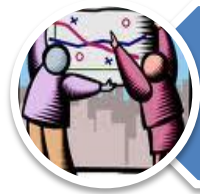
Thing we can do differently



Celebrate Bugs



Don't Punish the Testers



Don't abuse the metrics

1) Celebrate Bugs

It's a behavioural issue with humans that we don't like criticism yet if we want to be the best we need to be critical about ourselves and need to keep improving.

Following the analogy, bugs are criticism for the software applications & products and each stakeholder should welcome every bug undiscovered as an opportunity to improve rather than panicking or creating a big fuss over why it was introduced in the first place.

Of course, Post mortem meetings should be held to discuss ways to avoid such bugs from occurring in the first place BUT when the bug is found we should feel happy that at least it didn't break at customer's machine or caused him loss & dissatisfaction. It's equally applicable to all the stakeholders of the project who interact/deal with Testers.

"Drop by Drop forms the ocean": "Eliminating one bug at a time forms the product perfect".

2) Don't Punish the Testers

For:

- i) **Discovering late bugs:** We might sound bit out of data when we say this in an era where "*bringing quality upstream*" and "*do it right the first time*" are the mantras of success but we do believe that the old adage "***better late than never***" still holds higher importance.

How stakeholder (Dev/PM/Management) reacts when a tester comes with S1/P1 bug few days before Sign-off?

Yes, agreed that the bug should have been caught early or at least in the first or second test pass BUT there could be a valid reason for that to come up late in the cycle and Tester's should be given a benefit of doubt and more importantly encouraged for saving the team from the embarrassment of breaking into production and finally ending up with QFE and Hot fix

- ii) **Finding invalid bugs**

It's not always about finding bugs which have to be fixed. Testers should never be encouraged to make assumptions or miss an important bug because it was wrongly clarified by Dev or PM over email/chat/verbally.

The cost of raising couple of invalid bugs is much lesser than the cost of missing one bug perceiving that as an invalid bug by the tester.

3) Don't Abuse the Metrics

With the pressure from the management, Dev are expected to deliver "*Bug free code*" and so does Testers. It's now officially part of the commitments now 😊 and lot of things are on stake.

It has done more damage than good recently when PM, Developers become defensive to accept a bug as its going to make them look ugly when the metrics are taken. This leads to friction between the disciplines and make

people less tolerant to each other and bigger problems start creeping in.

Ideally the authors of each artefact be it Requirement, Design, Code, Test Harness, should be encouraged to comply to the process to make it fail-proof and metrics should be used as an indicator to measure the improvement and NOT to set hard and unrealistic targets with deadlines.



Challenge: Sandwich Problem

Clock is ticking; waiting for the blast

It is believed that a true professional shines like a diamond, talent is unearthed when one has to deal with the pressure and challenging situations. The important question here is: What do we mean by a challenge? It could be 'a Presentation on challenges faced by Testers' or 'find the bugs contest'.

Well the statistics of the survey done on testers across the industry suggests that majority of our community is suffering from the 'Sandwich Syndrome: *Too Much Work – Too Little Time*'.

Challenge:

- Arbitrary Schedule and Change Requests affects the Scope late in the cycle.
- Release Dates are sacrosanct – a compromise has to be done on the estimations /planned schedules.
- Developers eat up portion of our time.
- Environment issue uncovered just before the SIT, adding to the misery.

Known Wisdom:

- Push back on the Delivery Date.
- No Scope changes after baseline sign off. Only after CCB evaluation
- Change Control Board to evaluate the CR.
- Execute P1 test cases.
- BVT and Automation Framework is the solution.

Thing we can do differently



Work Smart, Not Hard



Vigilance Testing



Internal Ice-Creaming

1) Work Smart, Not Hard:

i. Story Script

To achieve more in little time, instead of the text book approach of executing Test Cases, testers can follow a Script outlining the use cases or Scenarios.

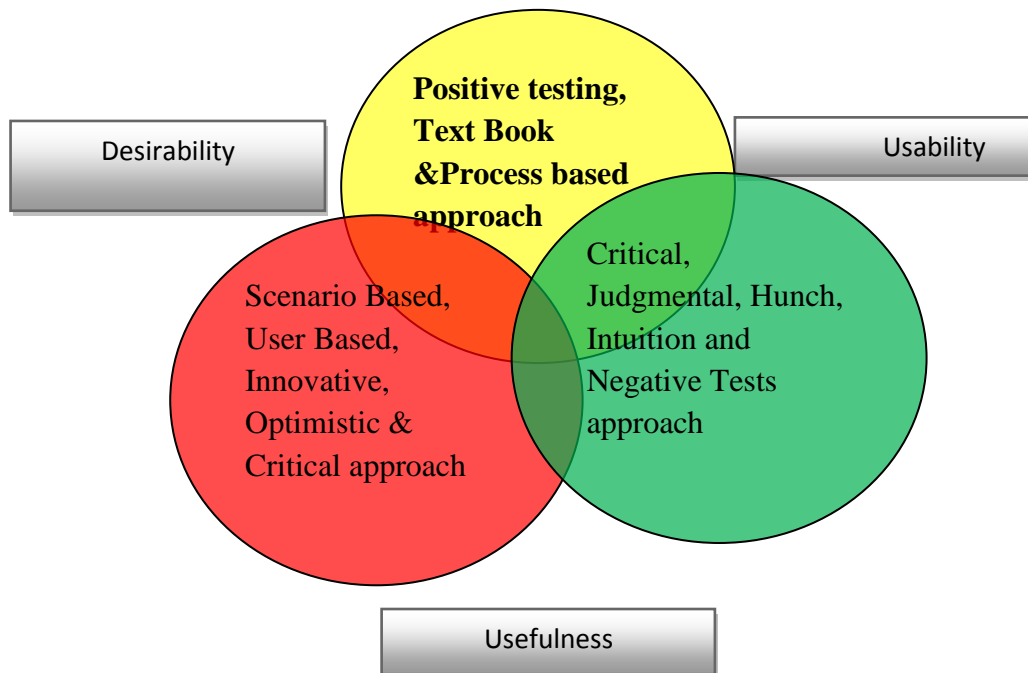
At the time of documenting scripts, the tester should consider scenarios more from the end-user or customer point of view. As described by *CemKaner*, The ideal scenario test has five key characteristics. It is (a) a story that is (b) motivating (c) credible (d) complex and (e) easy to evaluate.

ii. Discover Your Backbone

Each Team member possesses a unique knack of testing the software. Knowing the testers persona works magic in the crunch time. The common traits are:

- Logical text book approach, banking on facts and baseline documents
- Innovative / creative Approach.

- Intuition / Hunch / Out of the box test scenarios.
- User / Scenario based approach.
- Critical and Judgmental.
- Self-Motivated and Optimistic.



In the crunch time and pressure cooker type situations, the testers can be assigned a role based on the traits possessed. The theory works well as it ensures the both in depth and width wise coverage of the functionality under test.

2) Vigilance Testing

It is believed that Testers and Developers think differently, but they both work for the shared Goal and with the same aim. At the time of crunch, another approach towards saving and delivering on-time are:

i. Duet Integration Testing

In order to reduce the bugs in the build delivered to the test team, the testers can work with the developers for performing the Unit and Integration testing. This will not only help in uncovering the major issues early, but the regression testing effort is also reduced for testing Code Complete Build. In the duet mode – testers can

create and executes the Unit Test cases, whereas the dev team in turn can run the Integration test cases documented by the test team.

ii. **In House UAT**

The Dev / Test / PM team should frequently get together for a Bug Bash or In-House UAT. This will help in taking off the pressure from all the 3 teams:

- Test: P1/S1 issues are identified early.
- Requirements-based execution.
- Business process (workflow) (or, user scenario) based execution.
- Data driven testing gets carried out.
- Dev: Gets enough time to fix the issue encountered in Pre UAT.
- PM: gets to feel the stability of the application.

3) Internal Ice creaming

The recent survey done in the industry reveals that the employees are emotionally linked with the projects / software's they have worked in their previous assignment

65 % professional's feels still attached to the application they had worked on before.

Another very useful technique is derived from the above listed survey. An environment can be created and shared with the following people:

- **X Stake holders**: These are the people who was part of the team during some point of time in past. Most of the time they love to contribute in their extra time.
- **Community members**: The teams who are working on similar applications / domain. These include business stake holders, analysts, Group PM, Dev or Test Leads.
- **Technology hungry**: These are the people who hate dull, repetitive work. They are curious to work on new and upcoming technologies

Author's Biography

Raj Kamal is a Senior Test consultant with 9 yrs. of experience, specializing in different types of testing techniques, test automation and testability in different domains like Manufacturing, Healthcare and Higher Education. He holds an APICS certification in Supply Chain Management. Familiar with Rational, Mercury & Microsoft testing tools, he has helped teams develop test automation strategies and architectures for such companies as Cognizant Technology Solutions, Oracle Corporation & Microsoft. He also provides training in automated testing architectures and design. He is QAI(CSTE) & ISTQB Certified. He has a master's degree in Computer Applications. He is currently working as a Test Lead at Microsoft, India, Business Intelligence domain. His passion: <http://geektester.blogspot.com/>

Past presentations:

- Presented a webcast at Microsoft worldwide customers for [Data quality Testing](#)
- Represented Microsoft at QAI Software Testing International conference, Florida as a speaker "Application and Script Independent Automation Framework" (<http://www.qaiworldwide.org/qai.html>)
- Represent Oracle Corporation as Speaker on using "Mapping Rational Unified Process with Rational Testing Tools" at International Forum for Software Testing Professionals <http://ifstep.org/>
- Represented Microsoft as a speaker at [Test 2008 International Conference](#)
- Automation framework designed is accepted for Presentation and Publishing at [International Conference on Information Technology: New Generations](#), Las Vegas & IEEE Digital Library

Publications:

- Published a paper on [Resurrecting the Prodigal Son-- Data Quality \(Stickyminds\)](#)
- Published a paper on ["Adventure of BI/DW Testing" \(Stickyminds\)](#)