

Agile
Testing

Leadership Lessons for the Test & QA Profession



Vaidyanathan Ramalingam (Vaidya)

Director Engineering (Test)

Huawei Technologies India Pvt. Ltd., Bangalore, INDIA



A green rounded square button with a white border and a slight shadow, containing the text "Agile Testing" in white. The text is arranged with "Agile" on the top line and "Testing" on the bottom line.

Agile
Testing

5 Key Lessons!


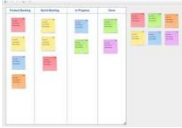








1. Waterfall Testing Vs Agile Testing
2. Testing Checklist – 5 W & 2 H
3. Trade Off Economics in Testing
4. Software Testing Eco System
5. RCA (Root Cause Analysis)





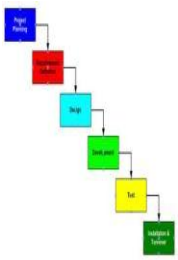
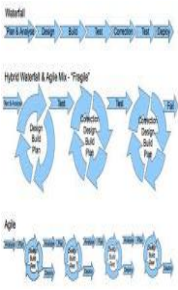




www.huawei.com

Huawei is a global telecommunications solutions provider with long-term partnerships with operators around the world. Huawei actively participates in 91 international standardization organization including ITU, 3GPP, 3GPP2, ETSI, IETF, OMA and IEEE. Huawei holds the world's #1 position in PCT Patent applications (WIPO 2008) and is ranked among the top 3 in LTE essential patents. Huawei's products and solutions have been deployed in over 100 countries and have served 45 of the world's top 50 telecom operators, as well as one third of the world's population.

*The content of this presentation is non commercial and based on self experience, interpretations/view points. The data provided may not be full, accurate and latest. The intention is knowledge sharing & to promote the software testing. The Product names, logos, brands, standard body/institute's names & other trademarks referred within the presentation are the property of their respective trademark holders.

Area	Testing In Waterfall Model		Testing in Agile Model	
(1) Test Requirement		<ul style="list-style-type: none"> •Complete & Base-line (freeze) •Separate Change Request / Enhancement 		<ul style="list-style-type: none"> •Incremental Req. as stories •To accommodate change/dynamic Req. •Req. prioritized based on: business values of customers, early realization & feedback
(2) Test Strategy		<ul style="list-style-type: none"> •Detailed upfront test strategy •Covers e2e test scope 		<ul style="list-style-type: none"> •Initial test strategy • incremental changes during iterations/sprints
(3) Test Team Structure		<ul style="list-style-type: none"> •Independent team /mindset •Test Manager: Project & Resource Management (people & lab) •Defined & Clear role 		<ul style="list-style-type: none"> •Collaborative team (dev & test) ; common goal; Agile mindset •Test Manager: <ul style="list-style-type: none"> •Test Planning & Estimation (input to backlogs) •Resource Management (people & lab) •Scrum Master: Release/Scrum Management •Lack of role clarity & 100% buy-in on agile practices
(4) Test Cases (TCs) Development		<ul style="list-style-type: none"> •One time TC dev. •Longer time •Ver. level contribution (partial & at the end) for acceptance tests 		<ul style="list-style-type: none"> •Iterative/Sprint wise TCs dev. @ story level (Functional & Non Functional) •TCs for inter-relation between stories (within & previous iterations) •Story level acceptance TCs dev. with customers
(5) Test Automation		<ul style="list-style-type: none"> •Automation behind manual test phase •Typically separate manual, automation & performance test team 		<ul style="list-style-type: none"> •Iteration/Sprint based automation & in C.I. •Shortage of time (poor scope) leads to missing/detailed scenarios •Automation suite quality deteriorates - Poor Focus on: <ul style="list-style-type: none"> - Scripting standard/review - SDLC approach - Maintenance (Ver.2 Ver.) - No Synch.: code <-> automation •Lack of attention leads to: Random Failures, Longer time to run, GUI errors, where the issue: defect or automation suite? •Coverage metric

Area	Testing In Waterfall Model	Testing in Agile Model
(6) Continuous Integration (C.I.)	 <ul style="list-style-type: none"> •Not strict •Relatively longer time •Relatively less frequent builds 	 <ul style="list-style-type: none"> •Strictly followed (early feedback/defects); back bone of agile •Efficient CI topology, architecture & timely automation suite are critical •CI pass rate metric
(7) Test Lab / Test bed	 <ul style="list-style-type: none"> •Required just before the testing (due to phase approach) 	 <ul style="list-style-type: none"> •Required from initial iteration onwards
(8) Test Execution	 <ul style="list-style-type: none"> •Longer, dedicated & multiple test cycles •Late product/application visibility •Phase wise test cycles 	 <ul style="list-style-type: none"> •All the stories TCs tested within the iteration/sprint •Continuous product/application visibility to testers •Customer demo at each iteration •Short test cycles for Non Functional Testing @ iteration end •Same Test engineers to iteratively test & be an expert in: <ul style="list-style-type: none"> – customer domain – test design – test automation / scripting – Non functional test – C.I.
(9) Defect Finding	 <ul style="list-style-type: none"> •Zero defect target @ end of Ver. •Relatively Defect backlog is larger •Test cycle wise # defects trend (high-low-high-low) 	 <ul style="list-style-type: none"> •Early & iterative defect finding •Less/zero iterative defect backlog •How much re-work metric •Iteration wise # defects trend (much controlled)

2. Testing Checklist – 5 W & 2 H



Why To TEST? To meet the Req., Architecture, Design, Code, Production Environment, Usability, Interoperability, Migration and their changes



What to TEST? Customer/Domain Requirements, Test Strategy, Weak & Strong Area, Test Iteration/Sprint Scope or Focus, Buggy Module,



When to TEST? Timeline asked for test iterations/sprints/phases/post release tests [R&D or Field tests]



Who will TEST? Functional/Non Functional testers; integration/field testers; ext. certification bodies, customer reps.



Where To TEST? Environment: Typical R&D Env, Integration (Platform/Component/Solution/Multiple Inter-operation Systems)



How to TEST? Test Techniques, Automation, [Script less Automation \(Ref: Qualitia\)](#), [Model Based Testing \(Ref: Conformiq\)](#), [Hypothesis Based Testing \(Ref: STAG S/w\)](#)



How Long to TEST? #Test Cycles, Release quality criteria, Iteration/ Sprints check point exit quality, Defects Trend, deadline/feedback based approach

*Note: The same approach can be applied for Test Automation also.
Example: Why to automate? What to automate? When to automate?....*

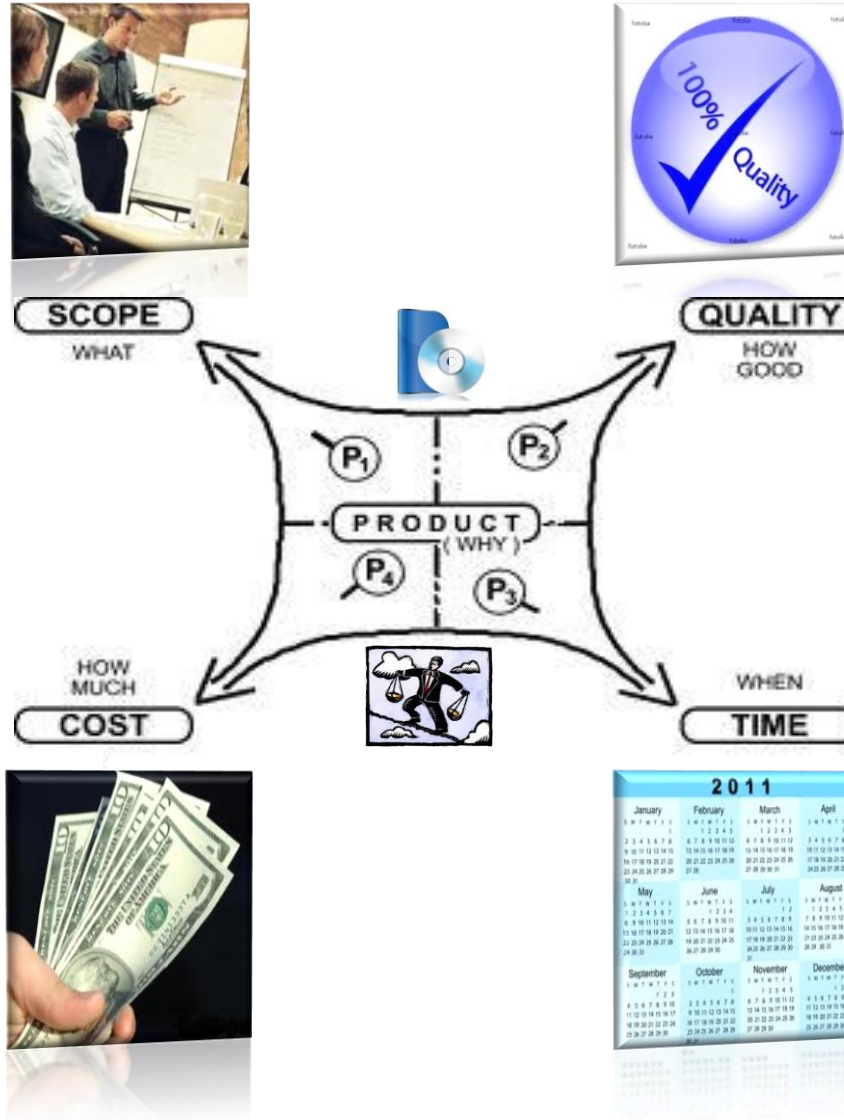
3. Trade Off Economics in Testing

- Test coverage scope
- Iteration/sprint wise plan
- Test priorities
- Automation scope / goal
- Late features / stories
- **Test Scope Risk mitigation**

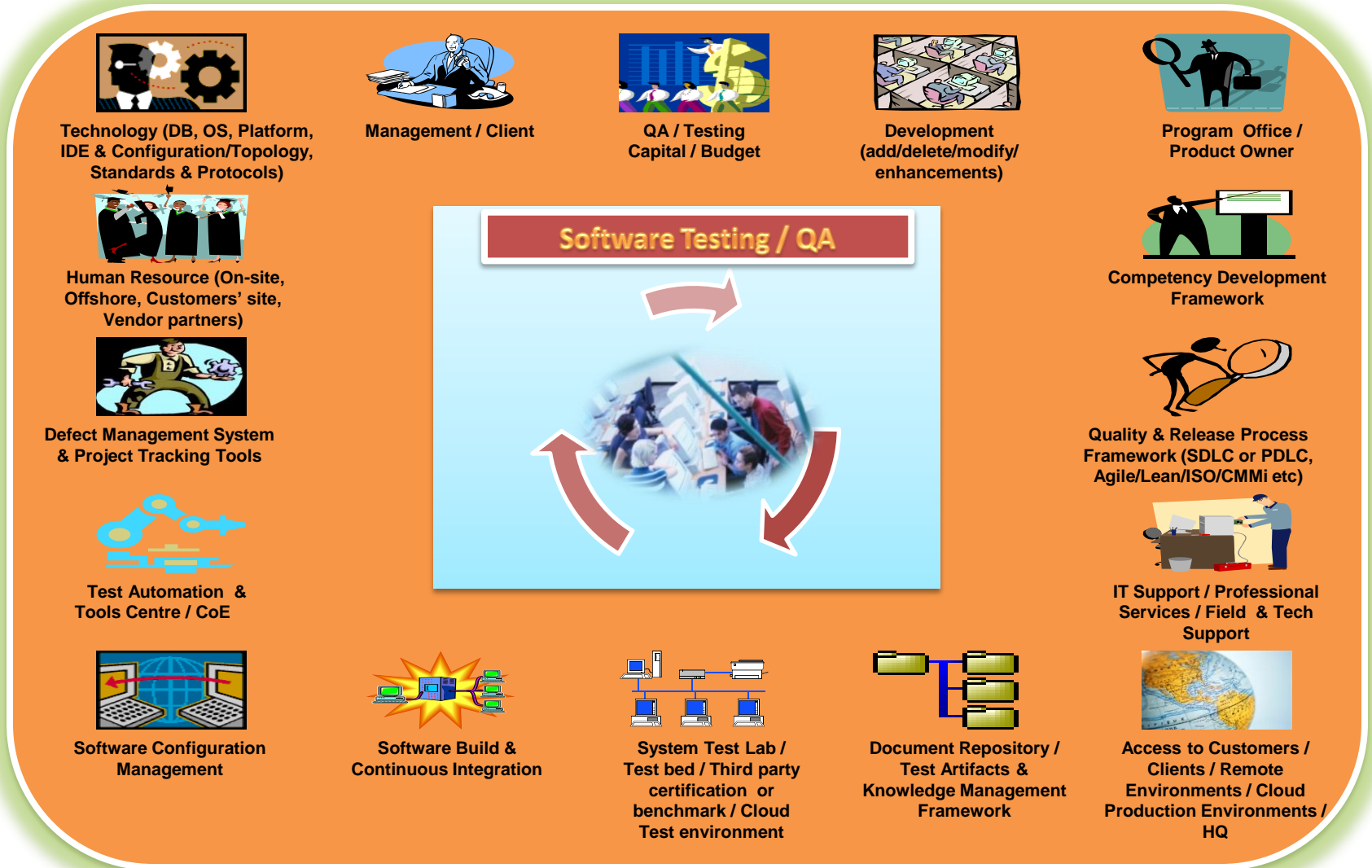
- Test Estimation/planning & resource allocation
- Test bed/lab need
- Quality based (defect trend/backlog) test cycles
- Automation cost
- **Cost Risk Mitigation**

- Iteration/Sprint & Release quality goal
- Design/Code changes
- Continuous reporting to PMO & prioritization
- Defect Trend & Backlog
- **Quality Risk Mitigation**

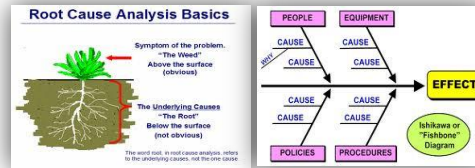
- Test phases/ iterations
- Handover to QA timeline
- Iteration/Sprint/Version level release dates
- Automation & Defect verification timeline
- **Time Risk Mitigation**



4. Software Testing Eco System



*(Needs collaboration with each stakeholders / elements to ensure **QUALITY, COST, TIME-LINE & SCOPE** based on your project need!)*



Focus: within the version ✓

Focus: version to version ✗

RCA on defects must be used like "preventive health check-up" to live longer & healthy (In-Phase Quality)

- The software we build & test to have in-phase RCA at every checkpoints/ iterations/ sprints to ensure no defect slippage to next stage
- Each phase/iteration/sprint to demand quality from previous one and ensure to next one
- Quality to be achieved continuously!

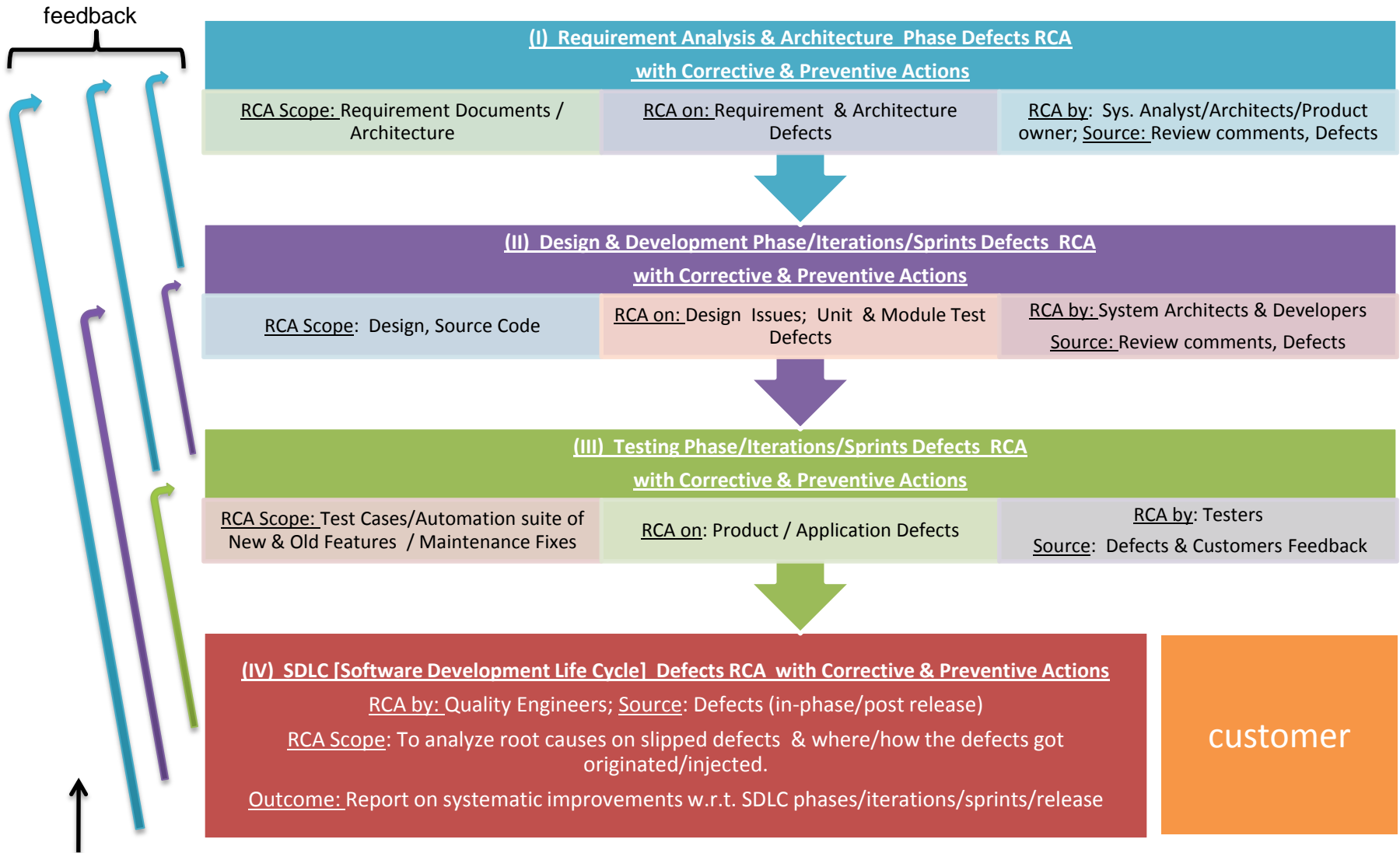


RCA on defects should "not" be like investigating "dead body" during postmortem (Post Release Quality)

- Often (in waterfall model) the RCA is done at the end of the project to improve the "next" release (not before death of the patient)
- In this case the poor quality impact is already made to S/w customers. (RCA learning in this case saves other patients, not the one who is already dead!)



5. RCA (Root Cause Analysis) 2 of 2



Iteration /sprint/release wise feedback to previous SDLC phases. Teams to conduct iterative/in-phase/Post Release RCA. Thus team can improve/prevent on defects slippage in next iteration/sprints/release.

Knowledge Is Power Thank You!



Feedback / Suggestions / Power Point Request to:
rvidya67@hotmail.com

Linked  in.

[Vaidyanathan Ramalingam](#)