BITS Pilani

Pilani Campus

# BITS Pilani presentation

K.Anantharaman
kanantharaman@wilp.bits-pilani.ac.in

BITS Pilani

Pilani Campus

# Module-10 – Managing Quality and Risks in Agile Project

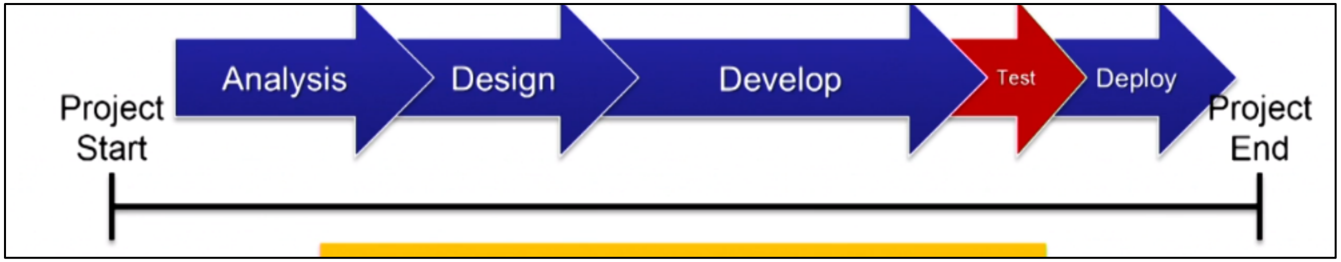
# Key Differences between Agile and Traditional Quality Management

- Integration of Testing with Development
  - Concurrent vs Sequential
- Testing Approach
  - More reactive vs More Proactive
- Responsibility of Quality
  - Overall Team <> QA Team
- Regression testing
  - Frequent (Code Changes), At end after Code stabilizes

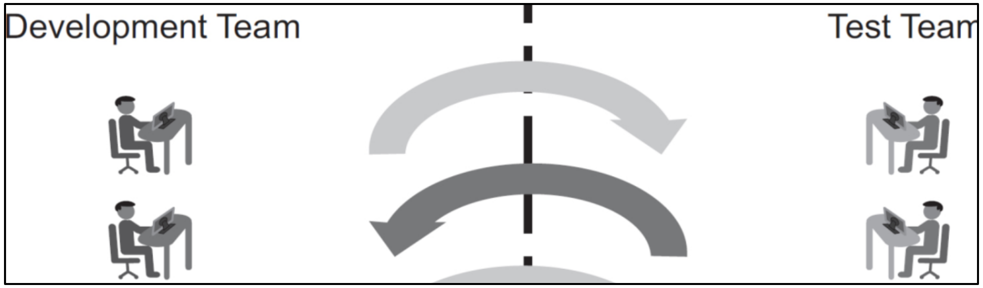
# Agile Development and Testing Practices

- Agile Development Practices
  - Continuous Integration
  - Code Refactoring
  - TDD
  - Pair Programming
- Agile Testing Practices
  - Repeatable Test Automation, Acceptance Drive test development
  - Exploratory testing, Concurrent Testing, Value & Risk based testing

# Agile Approach to Quality



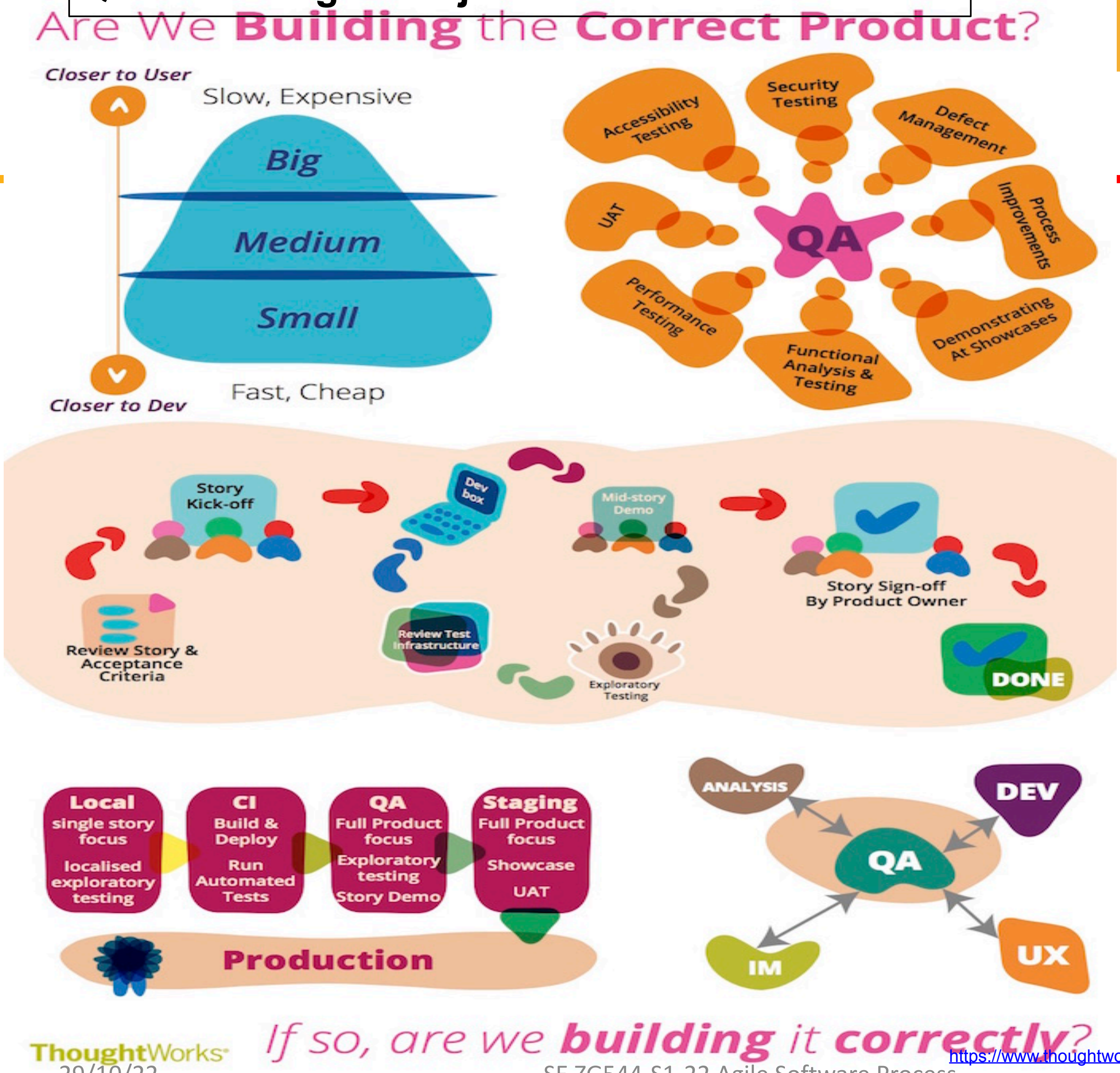
Water fall Project



- Agile approach to building quality product
  - Early delivery & Testing, Sprint Review, Customer feedback
  - Customer collaboration
  - Good Technical practices
  - Whole team participation to Quality
  - Test Automation

Ref: https://www.vivifyscrum.com/insights/qa-agile-project-management

# QA Role in Agile Project

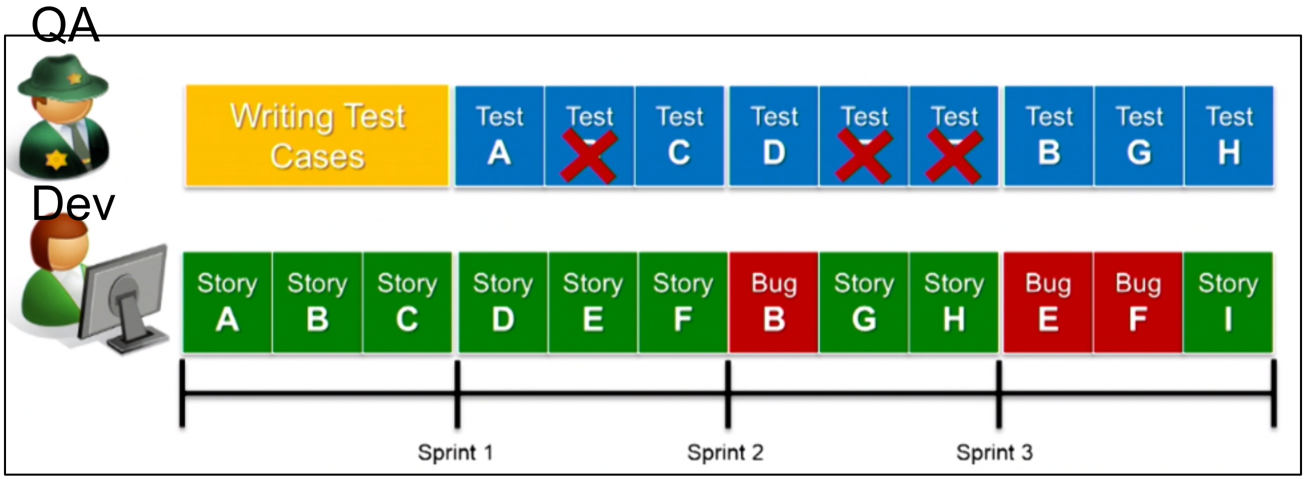


https://www.thoughtworks.com/insights/blog/qa-role-what-it-really

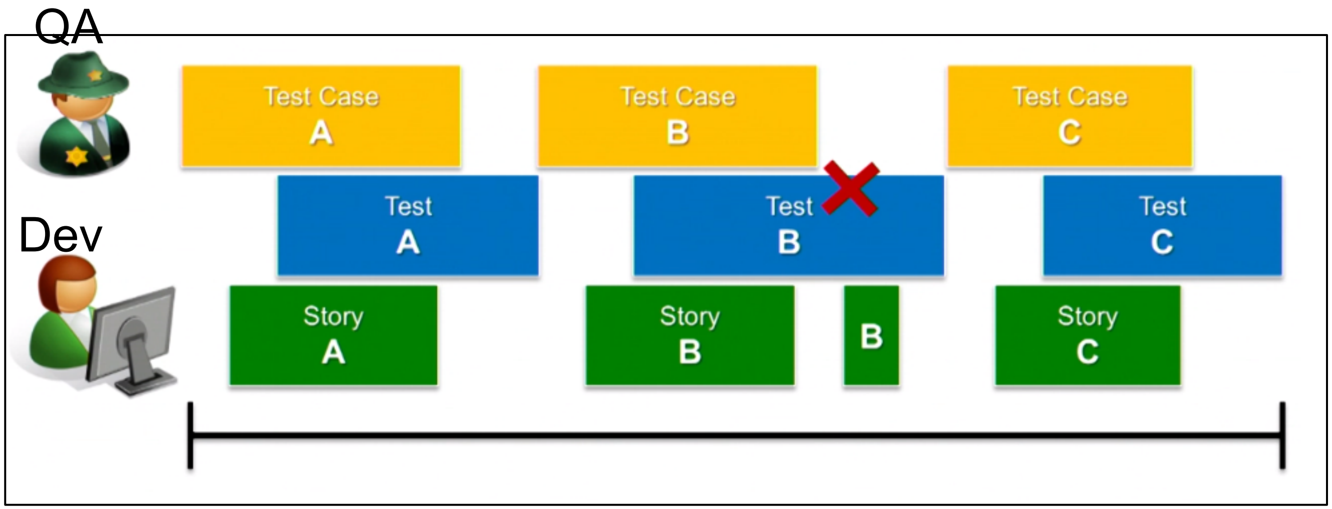
•Agile/Scrum: QA is involved in every aspects of Project/ Product development cycle

QA has a unique mix of all these capabilities.
QA brings the mindset of **"Are we building the correct product and, if so, are we building it correctly**?"

# Testing within Sprint



- QA lagging behind in Testing
- Bugs Snowballing effect



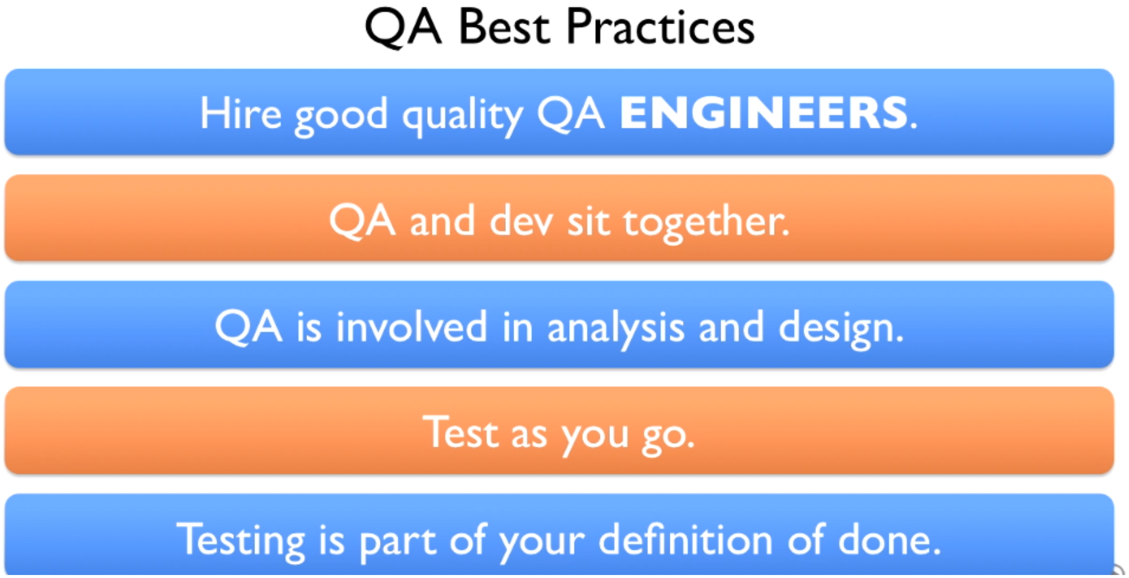
- Collaboratively testing with Dev.
- Fully tested Software
- Minimal Hands-off.

Dealing with bugs:

•Critical, Non-Critical, Enhancements

Source: Scrum Fundamentals and Advanced by Tommy Norman, Published by Addison-Wesley Professional 201

# QA good Practices



QA Best Practices

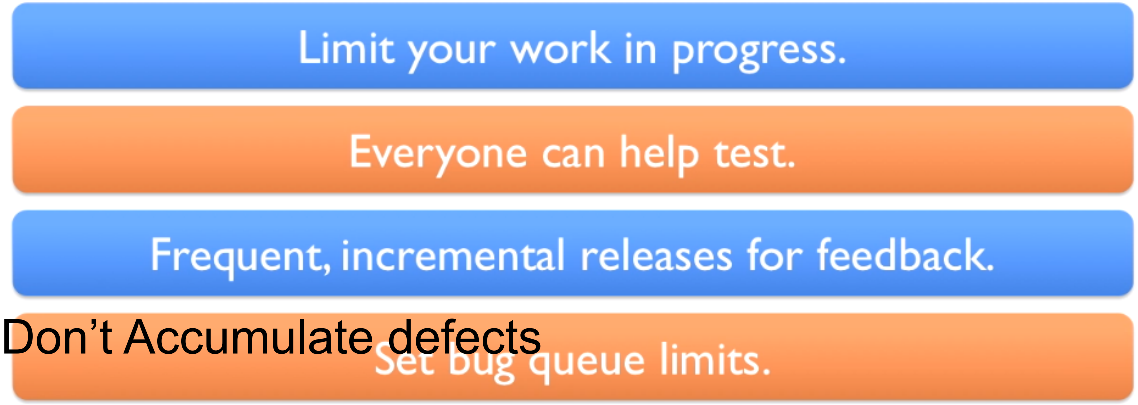
Hire good quality QA **ENGINEERS**.

QA and dev sit together.

QA is involved in analysis and design.

Test as you go.

Testing is part of your definition of done.



Limit your work in progress.

Everyone can help test.

Frequent, incremental releases for feedback.

Set bug queue limits.

Process Quality
Product Quality

Don't Accumulate defects

Source: Scrum Fundamentals and Advanced by Tommy Norman, Published by Addison-Wesley Professional 201

# Quizes

- Q1,Q2,Q3

BITS Pilani

Pilani Campus

# Risk Management in Agile

# Risk management in Agile

- Risks are uncertain event(s)
  - May affect your project positively or negatively
  - Positive Risk: A technology currently being developed that will save you time if released.
  - Negative Risk: Unavailability of Skilled resources.
- Agile methods have a built-in risk mitigation component.
  - Identify, Assess, Prioritize, Mitigate, Communicate
  - Daily meeting, Sprint review, Story Grooming, Retrospective

# Mitigation Strategies for positive risks or opportunities

**Exploit:**

- This strategy ensures that opportunity definitely happens. For example, assigning the most talented resource to your project to reduce the duration of the project.

**Share:**

- Allocating part of the ownership of opportunity to a third party to ensure that the opportunity definitely happens and risk is reduced. For example, going for a joint venture.

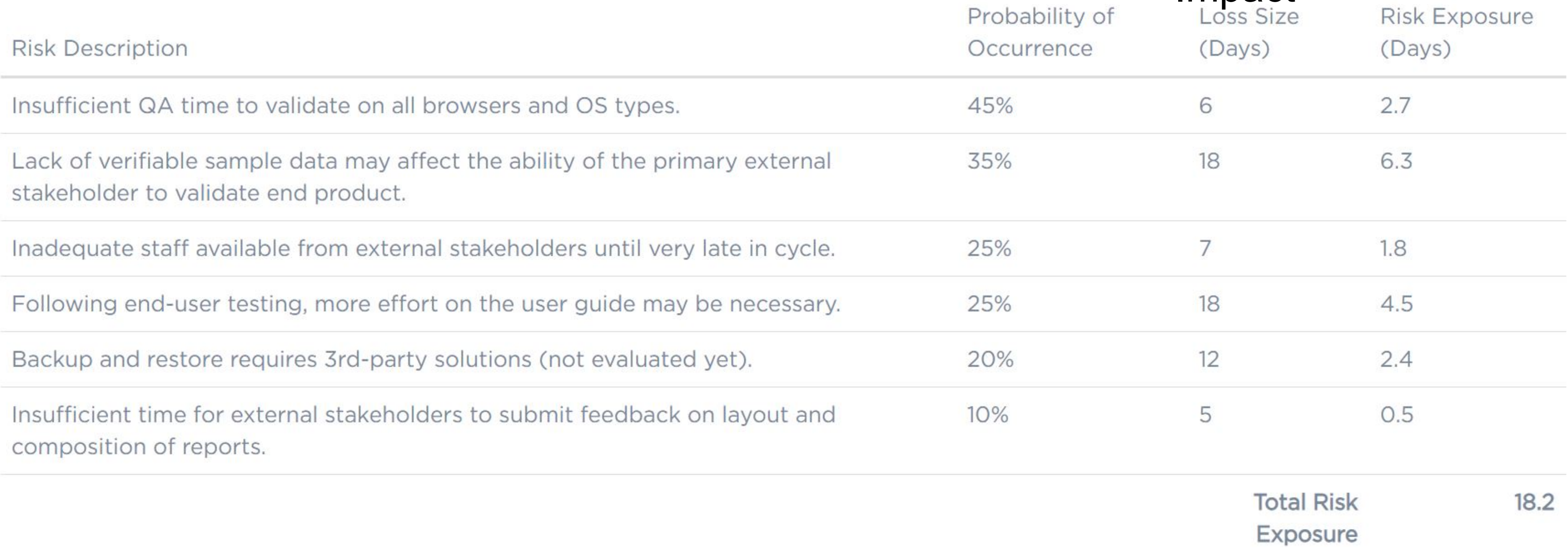
**Enhance:**

- This strategy increases the positive impact of the opportunity. For example, adding more buffer resources to an activity to finish it early.

# Mitigation Strategies for Negative Risks (Threats)

- **Avoidance:**
  - Eliminating a specific threat by eliminating the cause.
  - Use different set of tools/Libraries
- **Transference:**
  - Contracting, insurance warranties, guarantees, outsourcing the work are the examples of risk transfer.
- **Mitigation:**
  - Reducing risk probability and impact
  - Insufficient server resource: Increase CPU/Memory to reduce server crash
- **Accept:**
  - Accept the risk. Do not do anything.
  - Taking a  risky project with potential for future benefits.

# Communicating - Risk Register – An Example

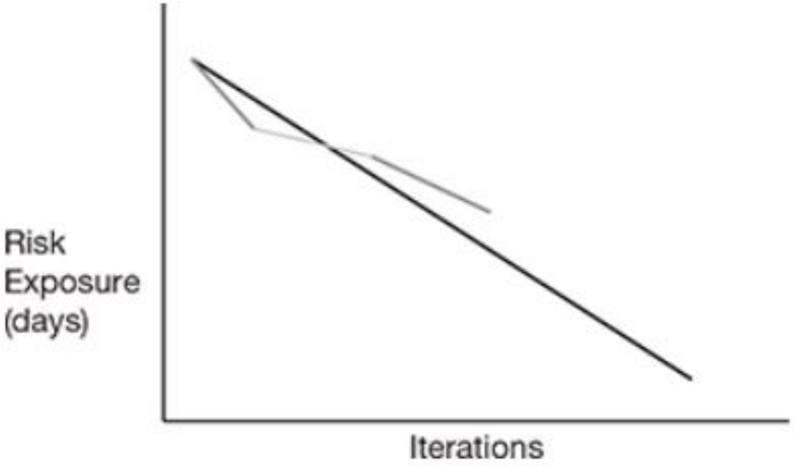


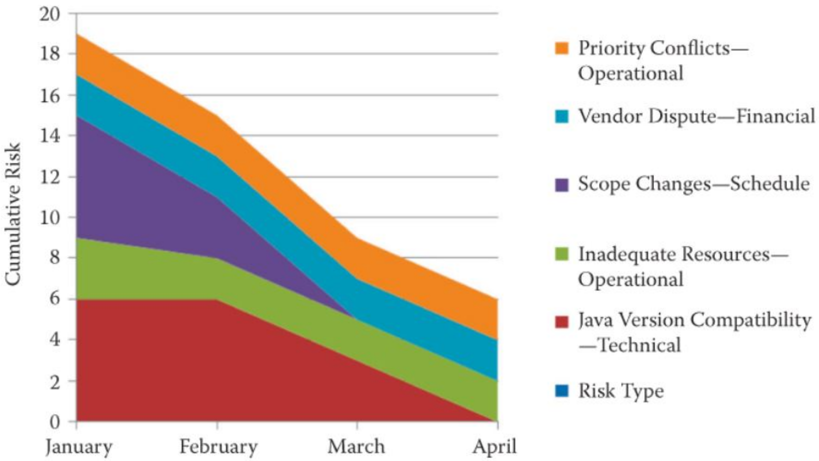
| Risk Description | Probability of Occurrence | Impact<br>Loss Size (Days) | Risk Exposure (Days) |
|---|---|---|---|
| Insufficient QA time to validate on all browsers and OS types. | 45% | 6 | 2.7 |
| Lack of verifiable sample data may affect the ability of the primary external stakeholder to validate end product. | 35% | 18 | 6.3 |
| Inadequate staff available from external stakeholders until very late in cycle. | 25% | 7 | 1.8 |
| Following end-user testing, more effort on the user guide may be necessary. | 25% | 18 | 4.5 |
| Backup and restore requires 3rd-party solutions (not evaluated yet). | 20% | 12 | 2.4 |
| Insufficient time for external stakeholders to submit feedback on layout and composition of reports. | 10% | 5 | 0.5 |
| | | Total Risk Exposure | 18.2 |

- Risk Impact : Measure the negative impact of the risk.
- Risk Impact Objectives: Cost, Time, Quality, Scope
- The could be many other columns in the risk register such  Date, Owner, Status, Priority etc..
- Risk Exposure: Probability * Impact

Source: https://www.castsoftware.com/research-labs/software-development-risk-management-plan-with-examples

# Risk burndown chart





- We can draw risk burn-down chart (graph) which contains iterative cycle number vs risk exposure days. Risks are monitored by the use of information radiators, daily stand-up meetings, and iterative cycle reviews and retrospectives. Y-axis of the risk burn-down chart contains risk exposure days. The X-axis of the risk burn-down chart contains the iterative number.

# Use Risk Management to Make Solid Commitments to executives

- Use Risk Multiplier in your estimation forecast
- Account for common risks - Turnover, Changing Requirements, Work disruption etc..

| | *Risk Multiplier* | | |
|---|---|---|---|
| *Chance* | *Rigorous Process* | *Risky Process* | *Description* |
| 10% | 1 | 1 | Ignore--almost impossible |
| 50% | 1.4 | 2 | Stretch goal--50/50 chance |
| 90% | 1.8 | 4 | Commitment--virtually certain |

**these multipliers are estimates gleaned from DeMarco & Lister's RISKOLOGY simulator and Todd Little's detailed analysis of hundreds of projects. The most accurate approach is to calculate your own risk multipliers from past project history*.

Source: https://www.jamesshore.com/v2/blog/2008/use-risk-management-to-make-solid-commitments

# Using risk multiplier in your estimation

- For example, if you are using a **rigorous approach**, your release is 12 iterations away, your velocity is 14 points, and your **risk exposure is one iteration**.

You would calculate the range of possibilities as:

- Iteration remaining = 12-1 = 11
- Points remaining = 11 × 14 = 154 points
  - Risk Multiplier* = **1,1.4,1.8** for Rigorous Approach, Risky Approach = **1,2,4**
  - 10 % chance: 154/1 = 154 points
  - 50 % chance: 154/1.4 = 110 points
  - 90 % chance: 154/1.8 = 86 points
  - In other words, when it is time to release, you are 90% likely to have finished 86 more points of work, 50% likely to have finished 110 more points, and only 10% likely to have finished 154 more points.

# An Example

Suppose, estimated number of sprints is 10 for a release

Product Backlog at end of Sprint 5 - F1,F2,F3,F4,F5,F6

Assume each feature size = 20, Velocity = 20

Total size of the remaining features = 6*20= 120 points

Sprint Remaining = 5, Risk Multiplier = 1,1,4,1.8

**6th Sprint Commitments:**

10% Chance : Sprint remaining * Velocity - 5*20/1 = 100 points

- 10% chance of delivering F1,F2,F3,F4,F5 (100 points)

50% Chance : 5*20/1.4 = 71.4 points

- 50% chance of delivering F1,F2,F3 (60 points), Stretch = F4

90% Chance : 5*20/1.8 = 55,5 points

- 90% chance of delivering F1,F2 (40 points), Stretch = F3

- Repeat this process after completing Sprint6

# Summary

- Agile Quality Management
  - Adaptive planning, Frequent reviews
  - Concurrent Regression testing
  - Test Automation
  - Proactive testing, Defect handling
  - QA Ownership, QA role is much larger compared waterfall method
- Agile Risk Management
  - Continuous risk assessment: Through daily standup meetings, Scrum planning meeting, release planning meeting, etc.
  - Agile projects have its own inbuilt risk handling mechanism, well aligned with quick risk identification, Ownership, and controlling mechanism.
  - The iterative nature of Agile projects identifies risks earlier in the project execution and also the risk process repeats for each and every iteration, thereby managing it in a better way.

# Quizes

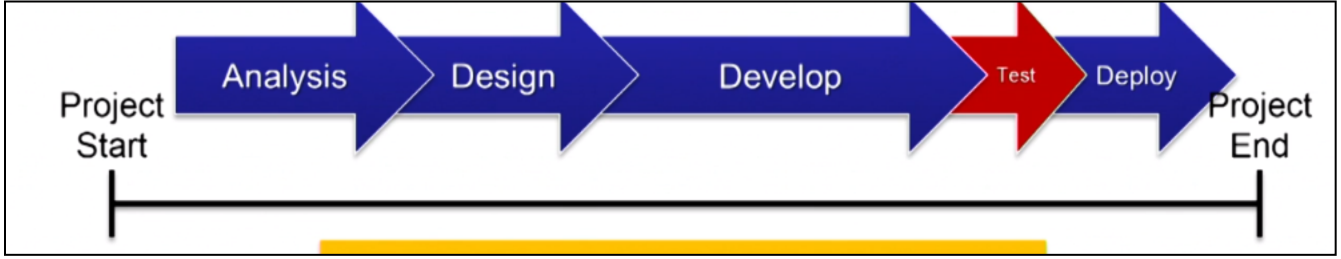
- Q4,Q5,Q6,Q7,Q8

# Additional notes – Quality & Risk management

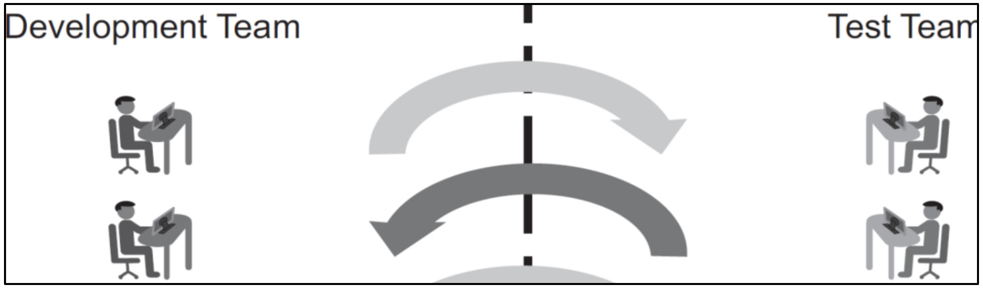
# Issues with Traditional Approaches to Quality Management

## Traditional Approach to QA

1. QA Audit for compliance Review        2. Most QA Testing happens at the end



2. Transfer of responsibility from developer to tester and vice versa



- In traditional sequential,  All of this ‘back and forth’ activity can easily create division within software development and QA teams if not managed correctly.

Ref: https://www.vivifyscrum.com/insights/qa-agile-project-management

# Agile Approach to Quality Management

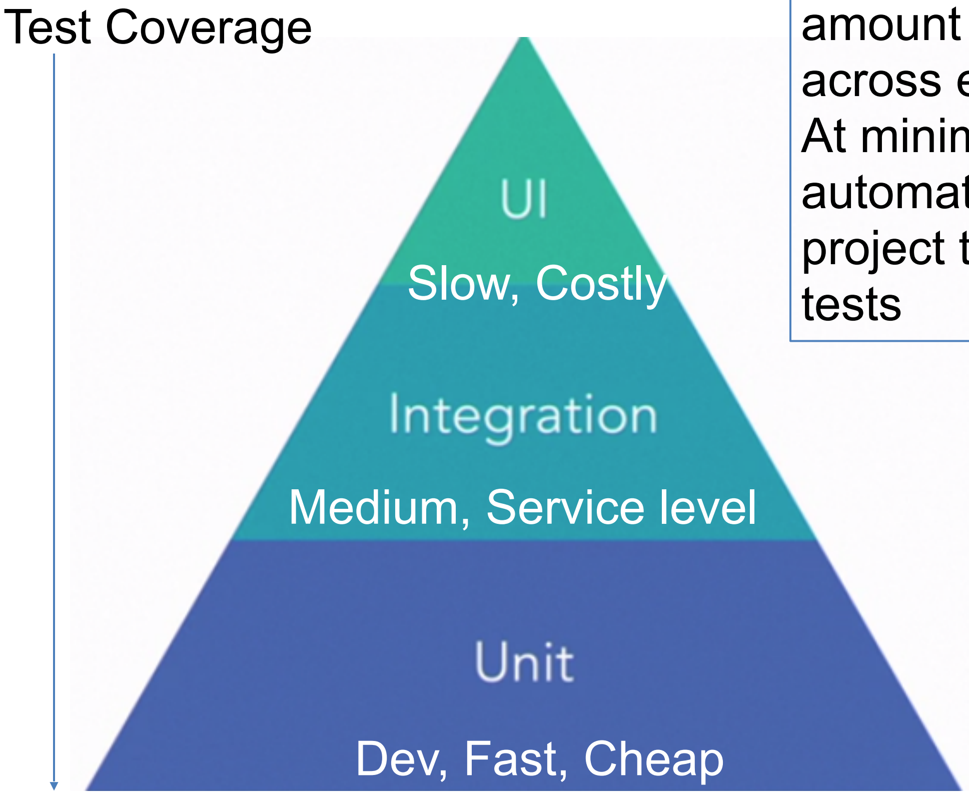
- Agile Manifesto & Agile Principles – Focus on **Building Quality In**
  - **Early delivery & Testing** of working software to customers as quickly as possible.
  - **Customers can also provide early feedback** on features, elements in the product which they like/dislike, and aspects of the solution that they wish to remove or modify.
  - **Agile values promotes collaboration with customer**, Team works with business team on daily basis, Simplicity, Technical excellence, Daily meetings, iteration feedback
  - **Good Technical practices** improves Quality: (Not specific to Agile)
    - TDD, CI, Collective code ownership, Pair programming, Refactoring, exploratory testing, reviews.
  - **Whole team approach** to Quality
  - In this way, Agile development can improve customer satisfaction and produce solutions that more closely meet customer needs.

# Agile Approach to Quality Management ….

- The hand-offs between programmers and testers (if they exist at all) will be so small as not to be noticeable.
  - Team work, Doing a little of everything (designing, coding, testing, and so on) all the time helps teams work together.
  - Tester creates automated tests and the programmer programs. When both are done the results are integrated. Hands-off is insignificant.
- There should be as much test activity on the first day of a sprint as on the last day
  - No distinct analysis, design, coding, or testing phases within a sprint. Testers (and programmers and other specialists) are as busy on the first day of a sprint as they are on the last.
  - For example, testers may be specifying test cases and preparing test data on the first day and then executing automated tests on the last, but they are equally busy throughout.

# Agile Approach to Quality Management …. Automate Tests at Different Levels

- Automation Pyramid



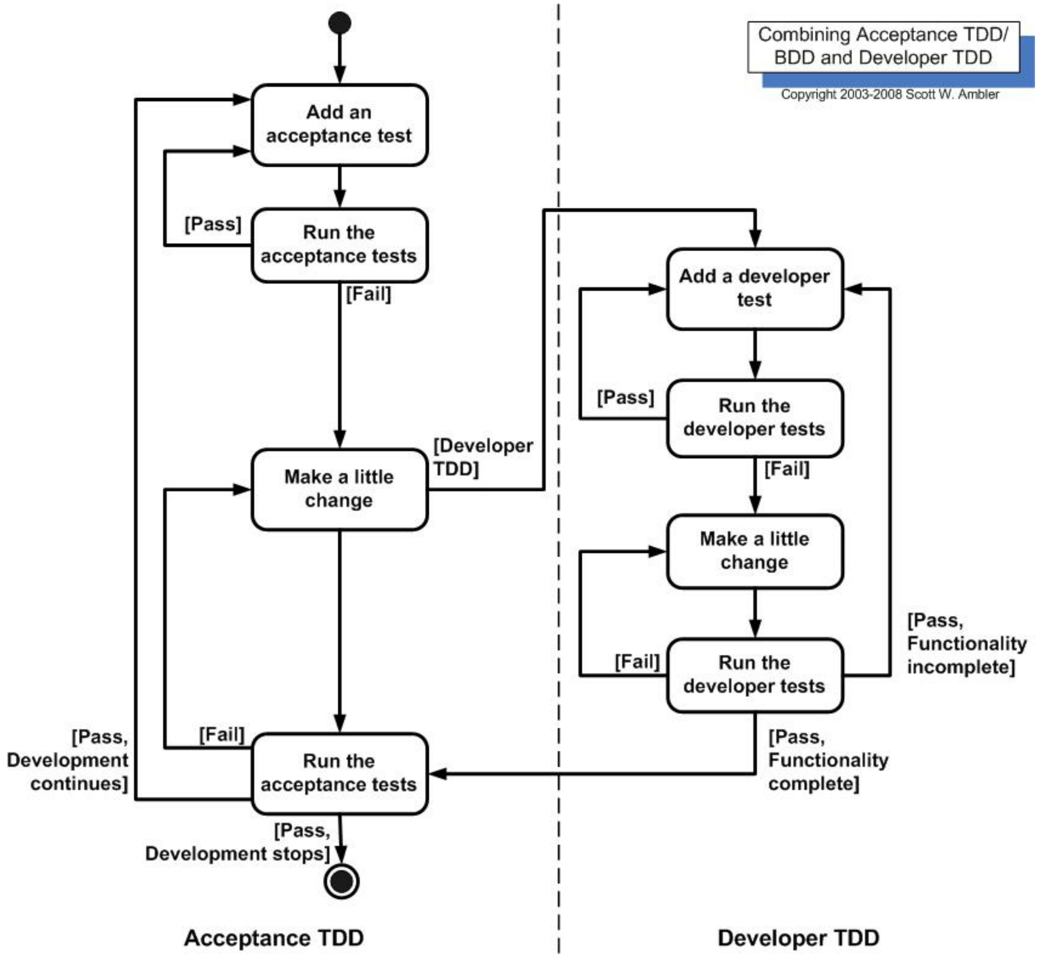
Visual representation of the recommended amount of test coverage that should exist across each type of test.
At minimum we should have three type of automated tests. Depending upon the project type we can have more type of tests

**Uniit Test:** Isolated tests , test functions, Fast, Need greater number of tests.
**Integration tests:** Slower, tests interfaces, databases, file system, other applications.
**UI Tests:** Tests end-end work flow. Much slower.

# The Role of Manual Testing

- It is impossible to fully automate all tests for all environments. Further, some tests are prohibitively expensive to automate. Many tests that we cannot or choose not to automate involve hardware or integration to external systems.
- Exploratory testing
  - Free form manual testing, Quick Test planning, test design test execution sessions.
  - Can identify missing test cases
  - Exploratory testing can uncover ideas that are missing from the user story as initially understood.
- Automate within sprint (Automation not optional)
- Pay off Technical debt

# Do Acceptance Test Driven Development



Analogous to test-driven development, Acceptance Test Driven Development (ATDD) involves team members with different perspectives (customer, development, testing) collaborating to **write acceptance tests in advance of implementing the corresponding functionality.**

# What is Risk?

- A risk is considered to be an uncertain event(s) that has the potential to contribute to the success or failure of a project.
- Positive risks are defined as opportunities and threats are risks that can affect the project in a negative way.
  - Examples:
  - Positive Risk: A technology currently being developed that will save you time if released.
  - Negative Risk: Unavailability of Skilled resources.
- Risk Management
  - Identify, Assess, Prioritize, Mitigate, Communicate
- Agile methods have a built-in risk mitigation component.
- Risk Burndown Chart – For communicating the risks

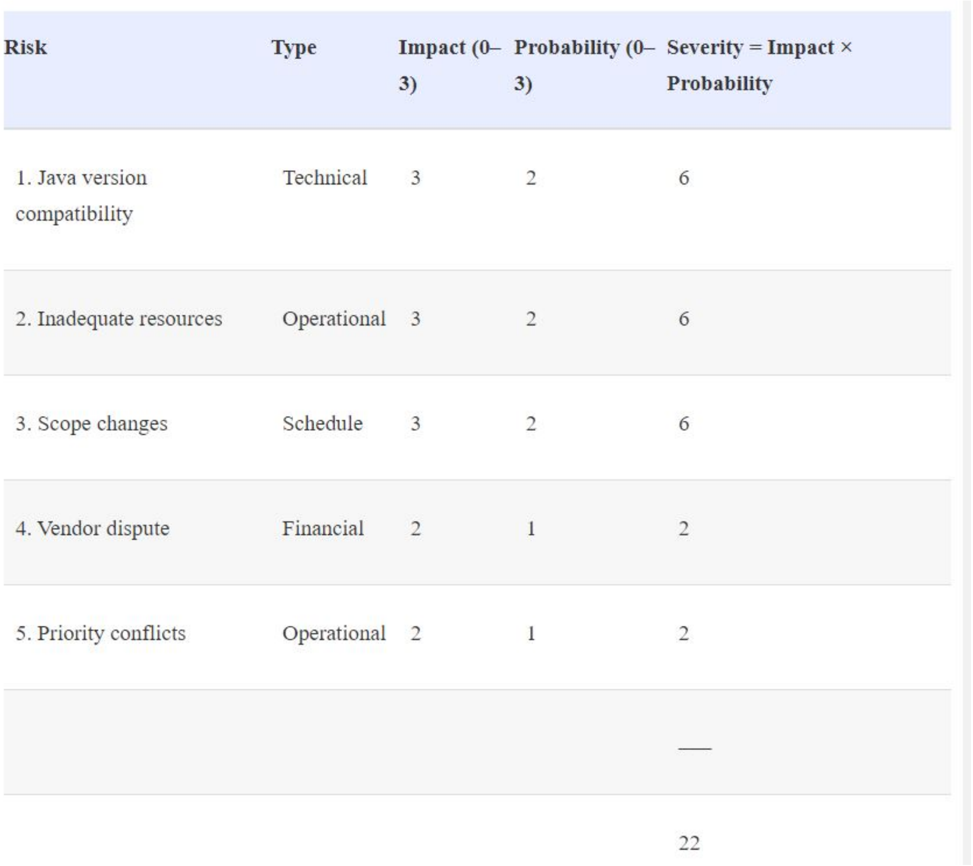
# Mitigating Risks with Agile Methods

- The flexibility of agile methods automatically reduces risk in the business environment.
  - Risk is mitigated because agile methods are flexible with adding or changing user requirements at any time in the project.
  - Missing or forgotten requirements can be included as soon as they are identified.
  - This results in low costs associated with managing this category of risks.
- Regular feedback reduces risk-related expectations.
  - As a result of the iterative nature of agile methods, there is adequate time to get feedback and establish expectations during the life cycle of the project.
  - Stakeholders and the agile team can avoid surprises because of requirements that have been communicated inadequately.

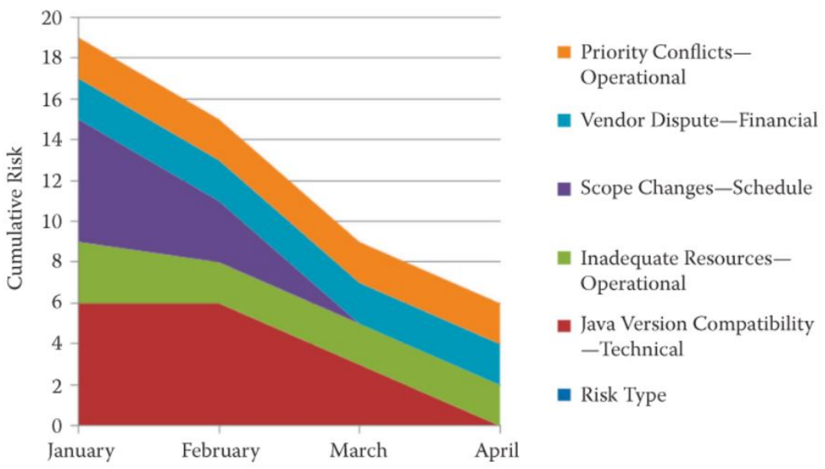
# Mitigating Risks with Agile Methods ….

- Agile team ownership supports reduced estimation risk.
  - When the agile team takes responsibility for estimates of backlog items, this leads to increased accuracy of the estimates that they provide which in turn results in the timely delivery of the product.
- Transparency is a risk reducer of undetected risk.
  - As a result of transparency, risks are always detected and addressed as early as possible.
  - This leads to better risk management and mitigation. During daily meetings, obstacles are communicated on a regular basis.
- Iterative delivery causes a reduction in investment-related risk.
  - As value is being continuously delivered through the iterations, investment risk is automatically reduced for the end customer.

# Risk Register- Another example



| Risk | Type | Impact (0–3) | Probability (0–3) | Severity = Impact × Probability |
|---|---|---|---|---|
| 1. Java version compatibility | Technical | 3 | 2 | 6 |
| 2. Inadequate resources | Operational | 3 | 2 | 6 |
| 3. Scope changes | Schedule | 3 | 2 | 6 |
| 4. Vendor dispute | Financial | 2 | 1 | 2 |
| 5. Priority conflicts | Operational | 2 | 1 | 2 |
| | | | | — |
| | | | | 22 |


Cumulative Risk
January
February
March
April
Priority Conflicts—Operational
Vendor Dispute—Financial
Scope Changes—Schedule
Inadequate Resources—Operational
Java Version Compatibility—Technical
Risk Type