Releasing your potential

logica

Testing in Scrum Projects

Kalevi Evans

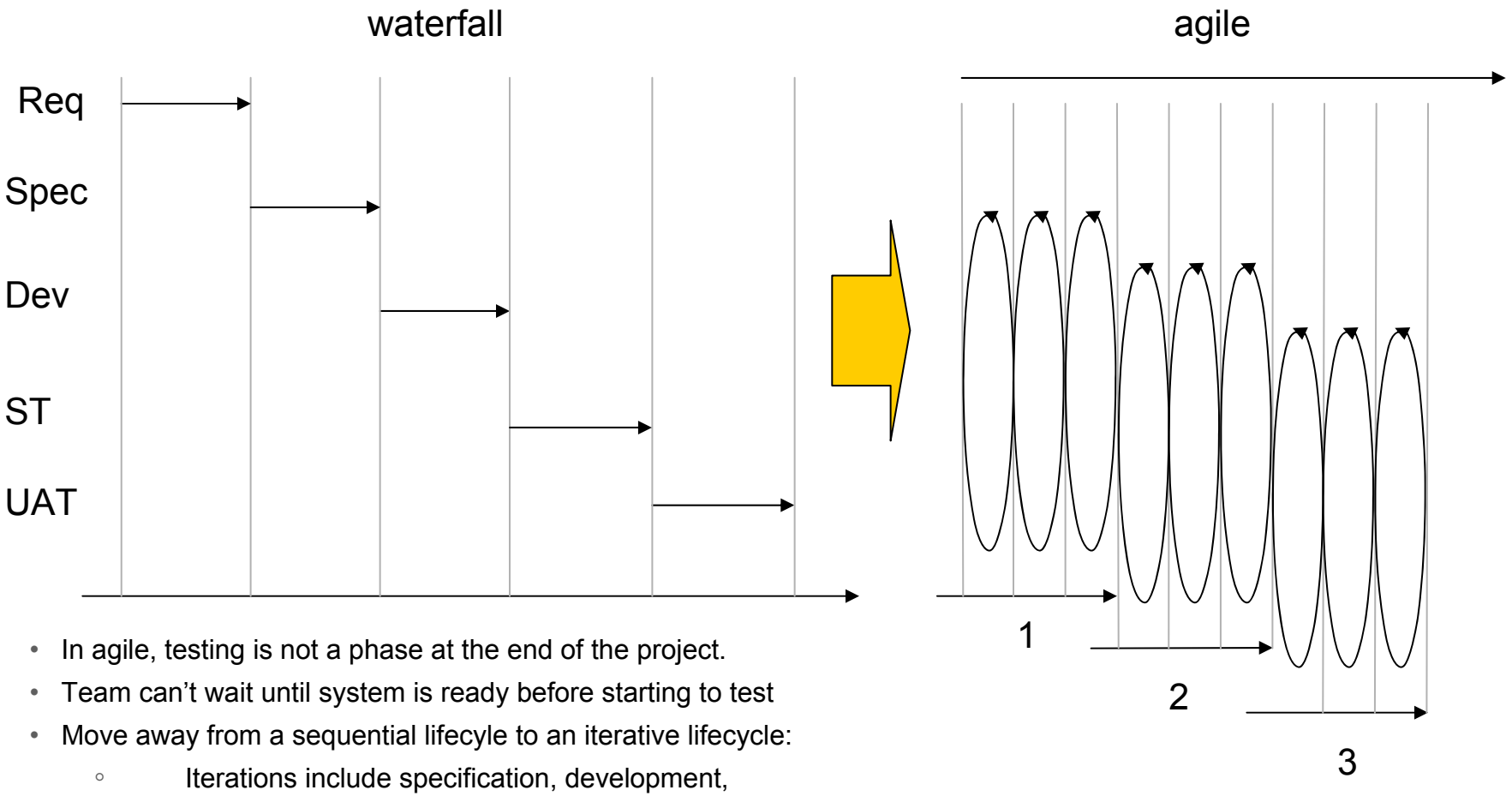
About Me

- Logica Suomi Oy (formerly WM-Data)
- Over 6 years experience
- Experience working in projects that apply the following software development processes: Waterfall, Iterative, V-Model, and Agile
- Currently working as a Test Manager in the Identity and Access Management concept area (IAM)

About this presentation

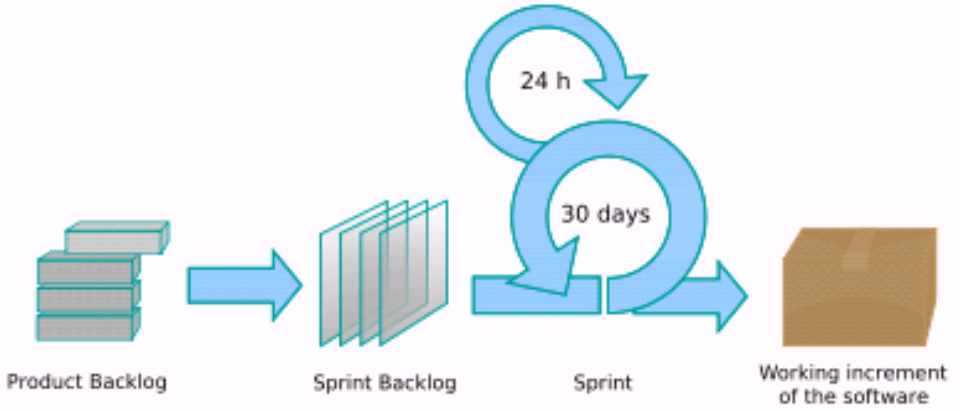
- Based on lessons learned from projects where Agile has been applied
- Focus is on Scrum
- Presentation is aimed at highlighting how important testing is in agile projects
- And the important role that the testing team has in process development

Waterfall v Agile



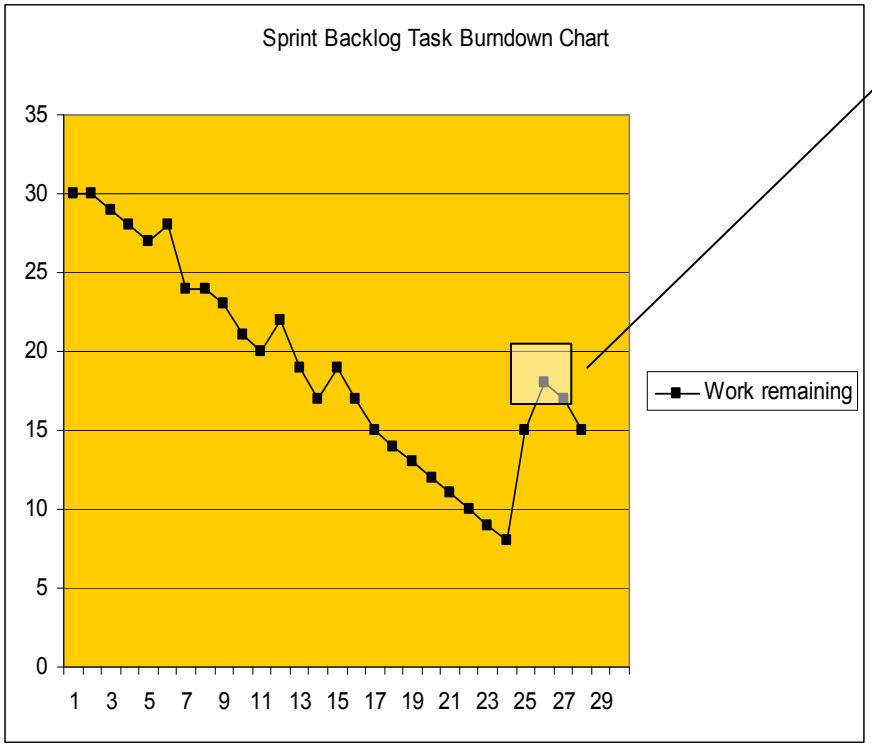
- In agile, testing is not a phase at the end of the project.
- Team can't wait until system is ready before starting to test
- Move away from a sequential lifecyle to an iterative lifecycle:
  - Iterations include specification, development, testing, and acceptance testing phases
  - There is collaboration between developers, testers and the customer
  - The team follow a common process

Agile Targets For Testing Professionals?



- Delivering working software, and just as important it is about delivering what the customer wants
- Transfer testing from the most inflexible phase in the project (where testing often occurs) to more flexible phases of the project
- Getting feedback as early as possible, and use this to reach our targets
- Testing is not only about finding faults but also about preventing faults being introduced into product
- And of course, to ensure quality

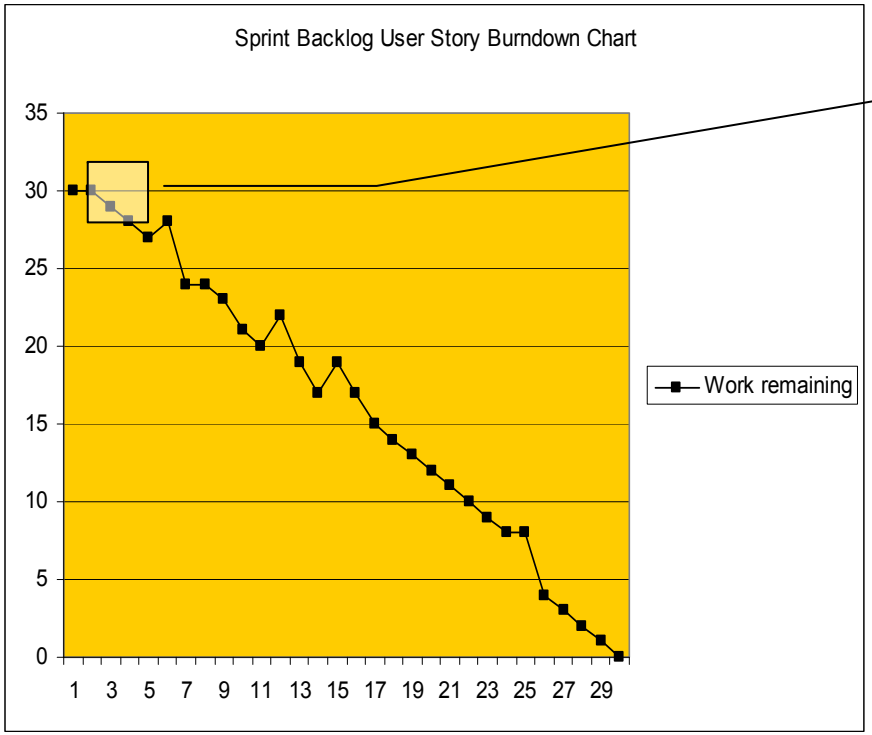
What We Want to Avoid in Scrum Projects



system or acceptance tests fail at later stages of sprint, indicative that we have failed to:

- Transfer testing to earlier phases of sprint
- Get feedback as early as possible
- Fail to meet customers requirements

Moving In The Right Direction!

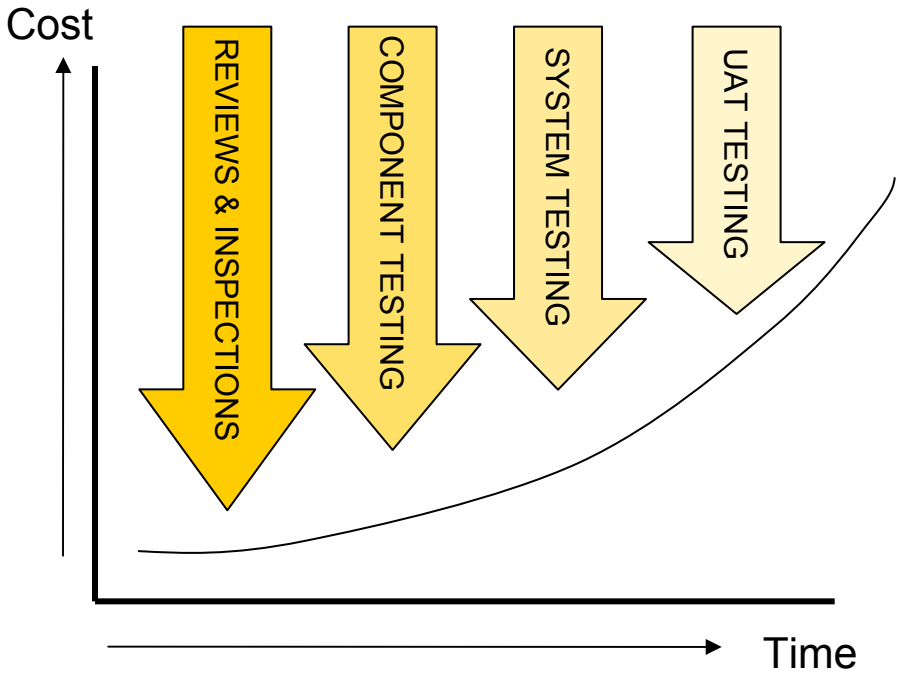


- Testing is the real measure of progress. And in agile this means that a user story is not ready until:
  - Development is completed
  - All associated tests have passed
  - Acceptance tests have passed

By following these steps we manage to get feedback as soon as possible, thus avoiding any nasty suprises at the end of the sprint!

This Approach Is Nothing New

- Scrum projects are no different from other intelligently managed projects (Waterfall, V-model) that recognise the added-value of testing and the importance of performing and involving testing at different stages of the SDLC
- The essence of agile is that testing activities are planned and executed from the very beginning AND we can meet our targets by working more closely together

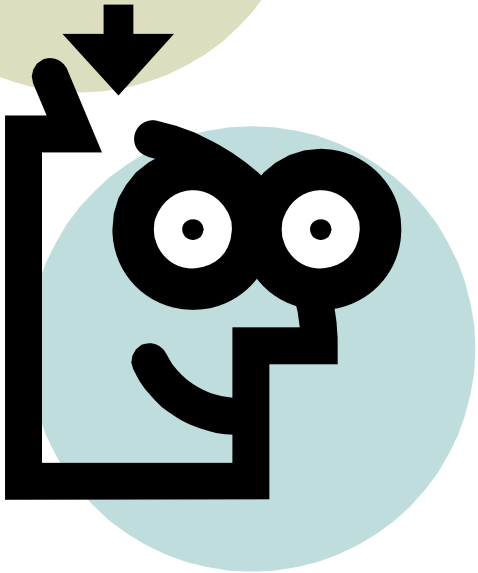


So How Do We Successfully Deliver Working Software In Practice?

Rule Number 1: Don’t Misinterpret the Agile Manifesto!

The Manifesto describes
the *values* of the agile community

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan



http://agilemanifesto.org/

And Some Guidelines ;-)

- Pilot project should be arranged in which scrum is applied in pure form. Each sprint should have clear and acheivable goals
- Possible challenges in scrum need to be discussed before the project kicks off and strategy for how we can handle them
- Firstly, the entire project team must understand the scrum methodology, and their respective roles. Common understanding on scope of sprint backlog
- Instead of having processes for the developers, and processes for testers we should have a common process that is easy to follow and which everyone understands
- We must understand how the schedule will affect our ways of working, and how we will apply agile processes to deliver working software
- All project members are equally involved in each stage of a scrum project and input is expected and equally valued

Risk & Requirement Based Test Management (RRBT)

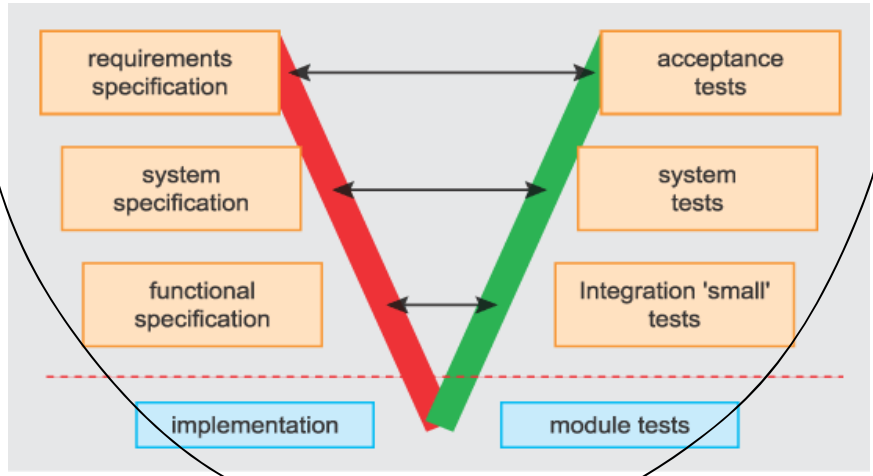
- Convey agile testing strategy and agile processses
- RRBT
  - Complements agile
  - Getting timely feedback from customers also means that we understand product risks. RRBT is about communicating with stakeholders and identifying these risks
  - Taking that knowledge of requirements and risks and useing it to prioritise our work
- Identifying what documentation is relevant
  - For compliancy
  - For current sprint
  - For re-use in future sprints
  - For handover to maintenance testing teams
- Facilitation & Leadership, & Coaching
  - Facilitate organization of team
  - Encourage team to work closely
  - Coaching role

Use Mini V-Model for each Requirement

- The V-Model is an extension of Waterfall.
- Demonstrates the relationship between each phase in the lifecycle and its associated phase of testing
- The agile approach can be considered an extension of the V-Model.
- The real difference here is that we no longer look at software development as a sequential one-pass process, but numerous iterations
- We could start talking about a Squashed V-model in which the order in which we perform tasks is less important

Each requirement

Test Level	Required
Component	✓
System	✓
Acceptance	✓



Risk Mitigation & Quick Feedback - Collaborate With The Customer

- Requires customer commitment not just involvement
- Testers mustn't shy away from the customer - instead engage customer
  - In designing acceptance test cases
  - Helps us understand the product risks related to each requirement and helps to streamline prioritization of tasks
  - Allows us to communicate product & project risks quickly and efficiently

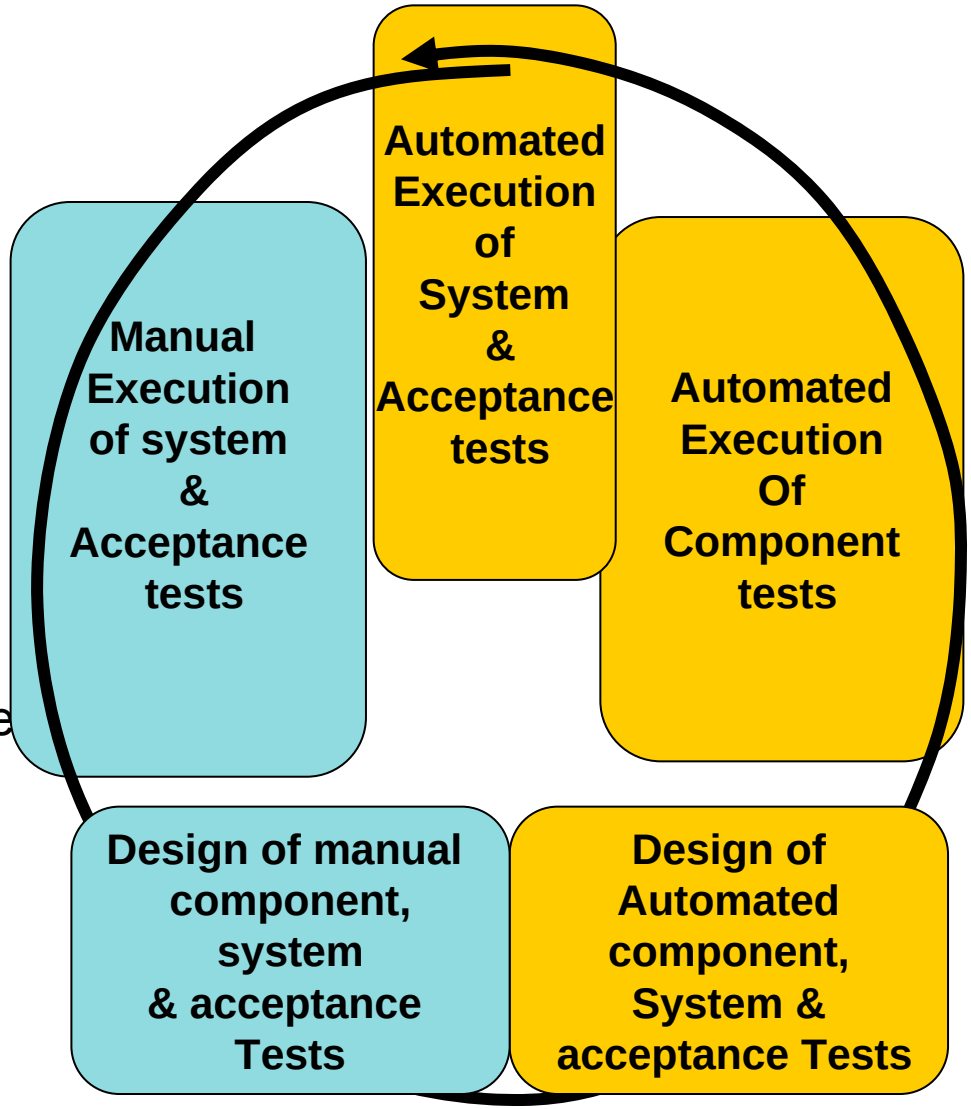
Defect Prevention *over* Defect Detection

- Systematic Test Design Techniques (BS 7925-2)
  - *As stated in ISTQB foundation certificate in software testing - systematic testing should be performed and this should be the starting point for all testing.*
  - Analyse each requirement and ask the question: *How do we test this*?
- Test-Driven Development (TDD)
- Systematic testing can also be complemented with exploratory testing Definition: Exploratory testing is simultaneous learning, test design, and test execution

Key Benefits
By applying systematic test design techniques and test-driven development we *can* make the project more agile by preventing defects entering the system. We thus help reduce overheads such as the number of defects logged to the system, excessive use of test management tools, and increased communication via tools
Increase test coverage
By documenting our tests we can reproduce them

Automating Our Tests – Gives Us More Room

- By automating our tests we empower ourselves with a regression test suite that maintains the same test coverage!
- By taking a TDD approach we can automate component tests
- Its just as important to automate the system & acceptance tests. With all that refactoring taking place we need to provide quick feedback that the internal changes have not affected the external behaviour
- We can't and shouldn't automate everything (cost and value, maintainability)



Bottom Up Development – Facilitates Efficient Testing

Bottom Up development

- System and acceptance tests can be spread evenly throughout the sprint
- Component tests can already start on day 1 of the sprint
- These act as building blocks for finalising the acceptance tests towards the end of the sprint
- Work on each requirement until its completed…then move on to next

Bottom Up Development – Facilitates Efficient Testing

Bottom Up development

- System and acceptance tests can be spread evenly throughout the end of the sprint
- Component tests can already start on day 1 of the sprint
- These act as building blocks for finalising the acceptance tests towards the end of the sprint

Top Down Development

- System and acceptance tests cannot be run until the end of the sprint
- Also component test execution will be pushed to end of sprint
- Trying to squeeze all these tests at the end of sprint increases the risk that we will not meet our target

Make More Informed Decisions About Code Quality & Readiness

- **Testing professionals** need to step up into the ”cockpit”
- Consultant role
- Understand the quality of code and be able to influence its quality
- Must understand the development process
- The tester still tests!

- **Coders** must take more responsibility over testing their own code
- TDD - ensures that all code is tested ”Don’t write code until you have written an associated test case”
- Must understand the testing process
- The developer still develops!

Key Benefit

– Through close collaboration, TDD, application of systematic test design techniques, risk and requirements based testing, and automation, the team can communicate more confidently about the readiness of the system

– A key purpose of testing is to provide timely information about the quality of the system being built

Example

Requirement:

Implement a Exchange/Outlook Distribution List which is based on group membership

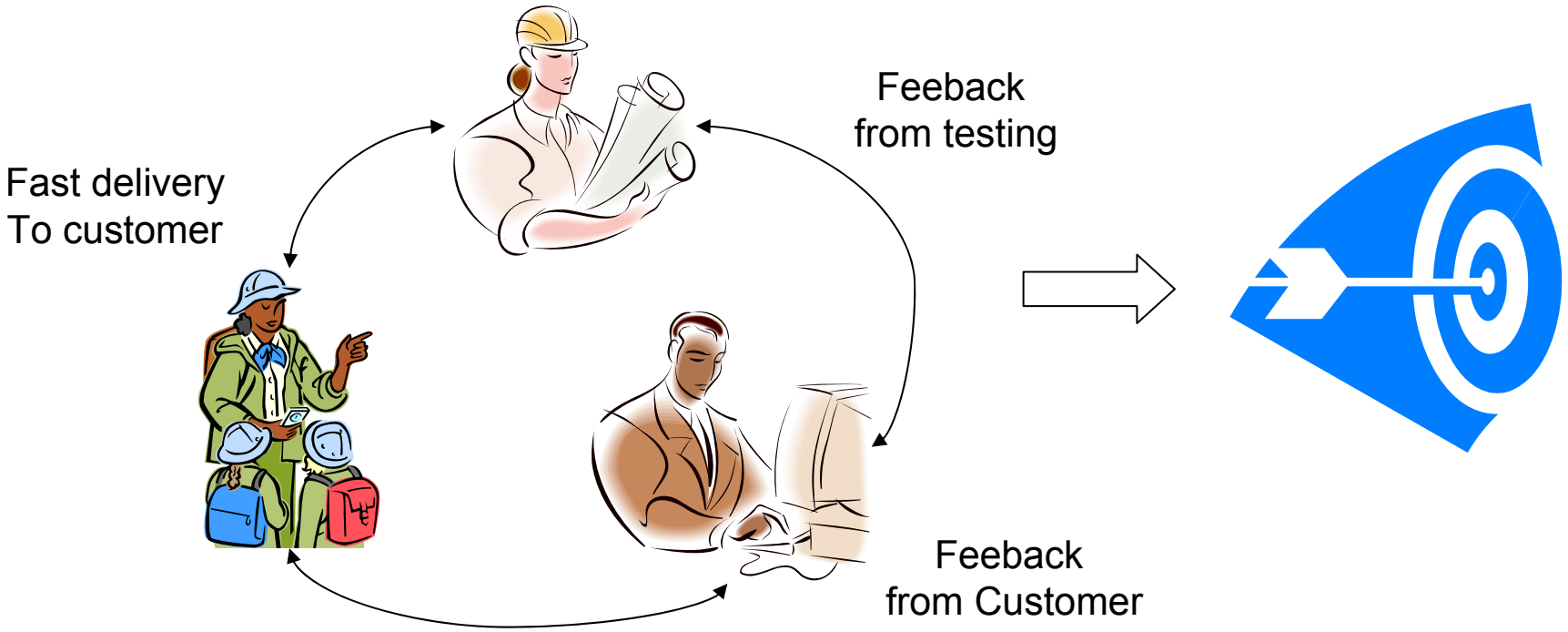
IF group has 500 or more members

THEN flag an attribute

Phase	Task	Value
1	Analyse risks with the customer using RRBT	Team understood product risks and used this to streamline activities
2	Identity test design techniques: - Boundary value analysis -Equivalence partioning -State transition analysis	Risks and requirements linked to test cases
3	Apply test-driven development which was also partially automated	The team knew that relevant test cases had been applied and that we had high test coverage. Able to free time to work on system tests and acceptance tests
4	Communicate status to customer	Able to communicate confidently. Led to more enthusiastic collaboration with customer

Shared Responsibility

- We are all responsible for the success of the sprint
- Individual responsibility is replaced by shared responsibility



Summary

- Being part of decision-making process to deliver working software
- Streamlining processes so that the sprint backlog is completed on time and applying a common agile process that ensure that the system is thoroughly tested in each sprint
- Influencing and understanding the quality of code better in order to make more confident decisions about readiness
- The testing community should therefore welcome agile projects

- *Sprint Retrospective meeting is important*

”At regular intervals the team reflects on how to become more effective, then tunes and adjusts it behaviour accordingly”

– http://agilemanifesto.org/principles.html

Releasing your potential

logica

Thanks!

Contact Details

Kalevi Evans

Kalevi.evans@logica.com

+358-40-5502616

References

- www.agilemanifesto.org

- Successful Test Management, An Integral Approach, *Iris Pinkster et al, 2006*

- Combining Agile, RRBT & TestFrame – *Logica Whitepaper*

- Agile Project Management with Scrum, *Dafydd Rees*
http://www.itwales.com/998612.htm

- ISTQB Foundation Course in Software Testing

-Agile & Scrum: What are these methodologies and how will they impact QA/testing roles? *Marina Gil Santamaria*
http://www.stickyminds.com/sitewide.asp?ObjectId=12964&Function=edetail&ObjectType=ART

-Exploratory Testing Explained, *James Bach*
http://www.satisfice.com/articles/et-article.pdf