Agile Project Failure: Root Causes and Corrective Actions

Jeffery Payne | Chief Executive Officer | jeff.payne@coveros.com

Jeffery Payne

Jeffery Payne is CEO and founder of Coveros, Inc., a company that helps organizations accelerate software delivery using agile methods. Prior to founding Coveros, he was the co-founder of application security company Cigital, where he served as CEO for 16 years.

Jeffery is a recognized software expert and popular keynote speaker at both business and technology conferences on a variety of software quality, security, DevOps, and agile topics. He has testified in front of congress on issues such as digital rights mgmt., software quality, and software research.

Jeffery is the technical editor of the AgileConnection community (www.agileconnection.com)

WHO WE ARE

Coveros helps organizations accelerate the delivery of secure, reliable software.

We provide consulting, coaching, and learning opportunities to enterprises, teams, and individuals.

Our Products + Services

Consulting Services

- Agile software development
- Agile testing and automation
- DevSecOps engineering
- Application security

Immersive Learning

- 50+ hands-on training classes taught in both public and private settings
- Enterprise, team, individual coaching to reinforce concepts taught in class

Community Support

Software Conferences

STAR EAST

STAR WEST

Agile + DevOps WEST

Agile + DevOps EAST

Software Communities

AGILECONNECTION™

STICKYMINDS™

TECHWELLHUB
A SLACK COMMUNITY

Coveros customers receive discounted or free conference passes based on volume of business

Agenda

Case Study #1

- Healthcare system
- Scenario
- Root causes of failure
- Corrective actions

Case Study #2

- Financial services organization
- Scenario
- Root causes of failure
- Corrective actions

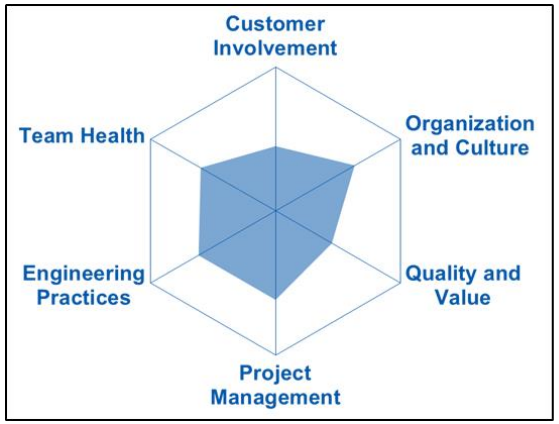
Discussions and Q&A

- Discuss healthcare system results
- Discuss financial services organization results
- Overall Q&A

Assessment Methodology

15+ Years of Data **200+ Assessments**

Teams to Enterprise



Score Summary	
Customer Involvement	2.25
Organization and Culture	3.17
Quality and Value	2.25
Project Management	3.08
Engineering Practices	3.08
Team Health	3.00
Overall Weighted Score	**13.83**
Overall Maturity	**2.81**

Agile, DevOps, Testing, DevSecOps Models

Case Study #1

Scenario:

- Three-year project to implement mission-critical healthcare system
- Organization has received agile training and purchased tools from a tool vendor
- Vocal stakeholders often disagree on priorities
- Development teams structured by app tier
- Testing team separate from dev teams
- Trouble hiring, training, and retaining staff
- Development and testing envs do not match
- Releases are done manually

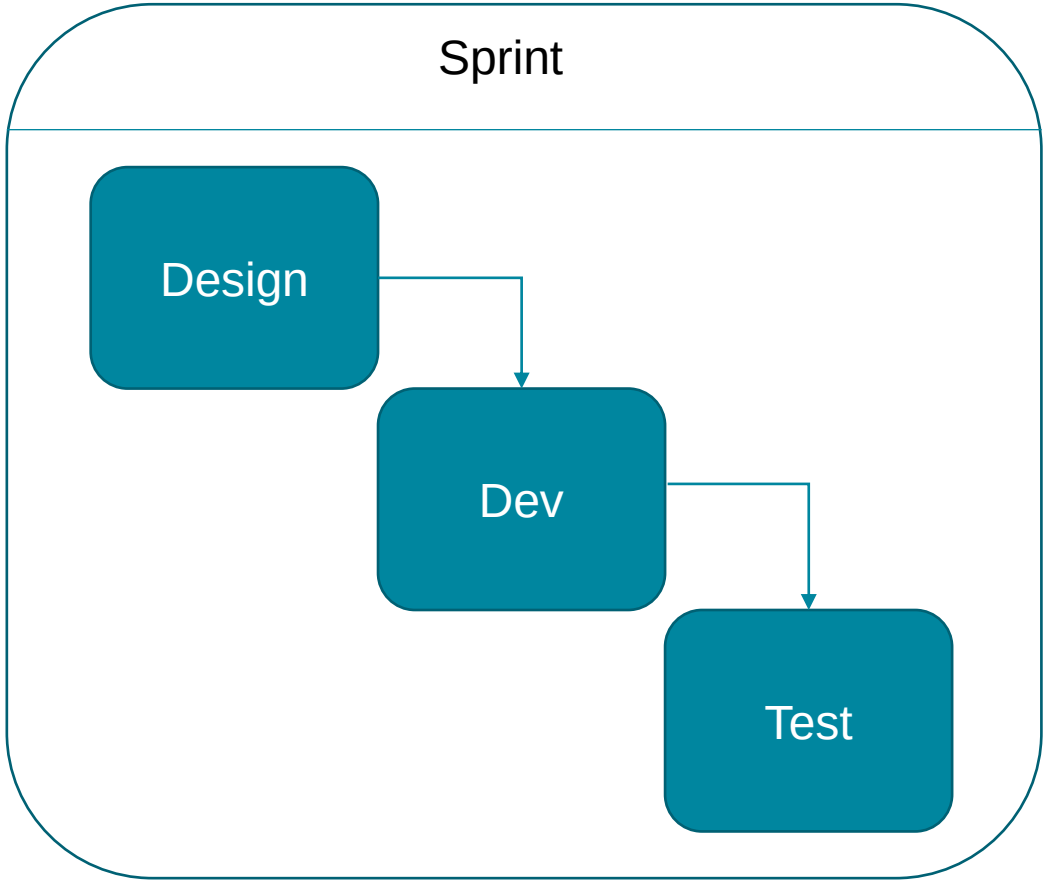
Large-scale Healthcare System

Reason #1: Scrummerfall

Many orgs adopt Scrum but forget to change the way software is designed, developed, tested & delivered.

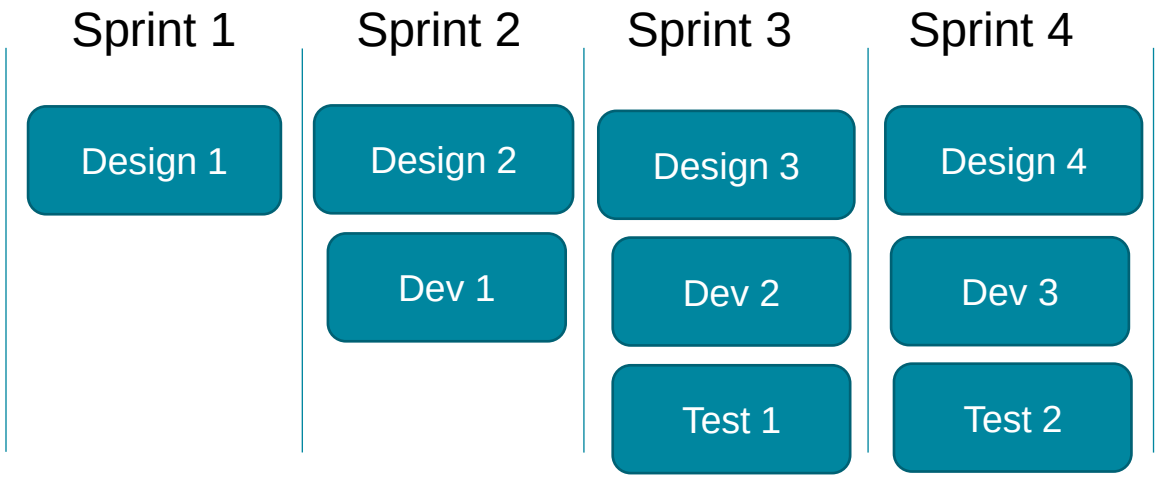
Scrummerfall Classic

- Waterfall handoffs are still done, just really fast!
- Often occurs when business is pushing for agile but development teams are not engaged.
- Results in elongated Sprints or working software not being delivered each Sprint



Scrummerfall V2.0

- Shifting testing (and sometimes design) to other sprints
- Further elongates delivery of working software
- Dangerous disconnect for the bug fixing process

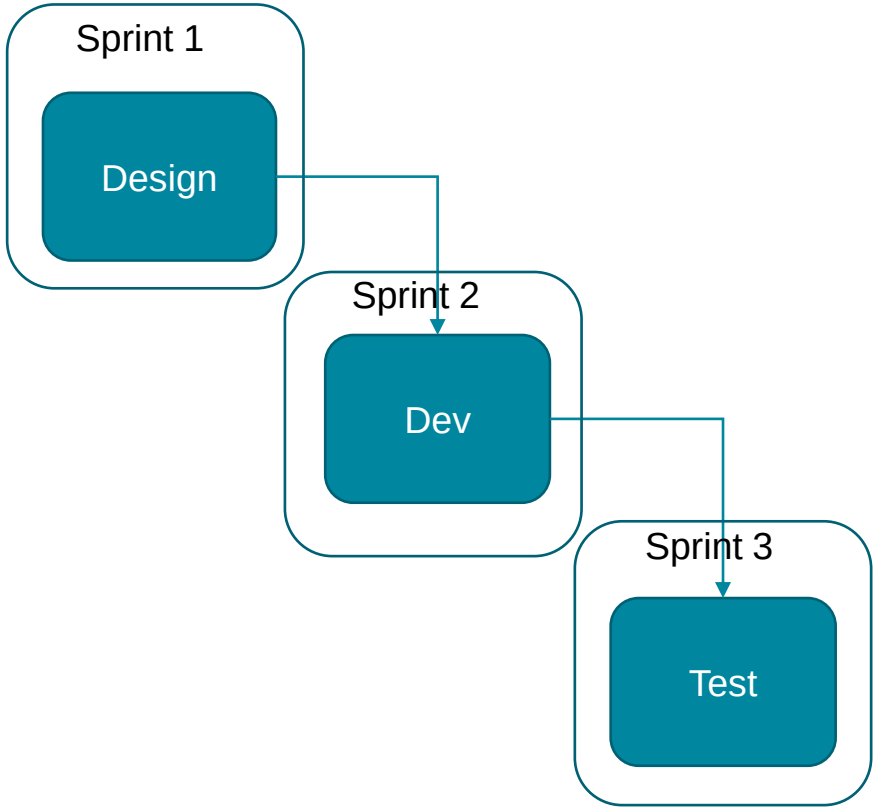


Scrummerfall Federal

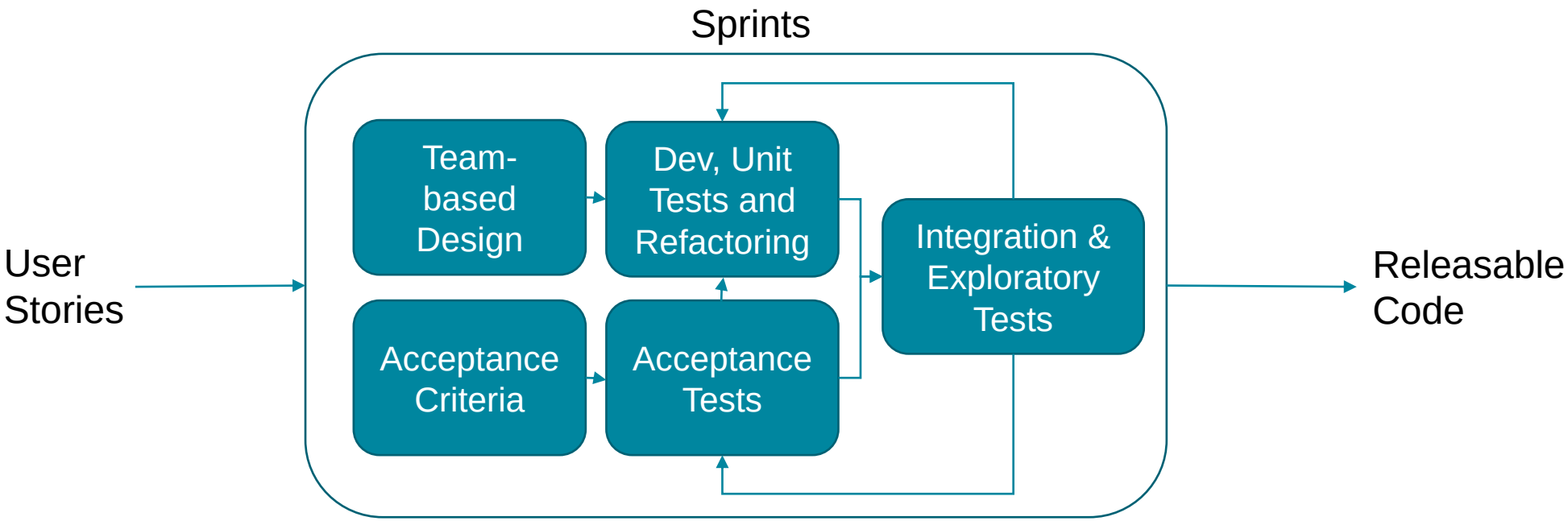
- Occurs when an organization formally separates planning/design, development, and testing in their process
- Results in significant rework/waste as requirements and design have to be updated
- Working software is never produced during sprints as testing is a ‘hardening’ sprint

Scrummerfall Nextgen

This is so not agile I can’t even comment!



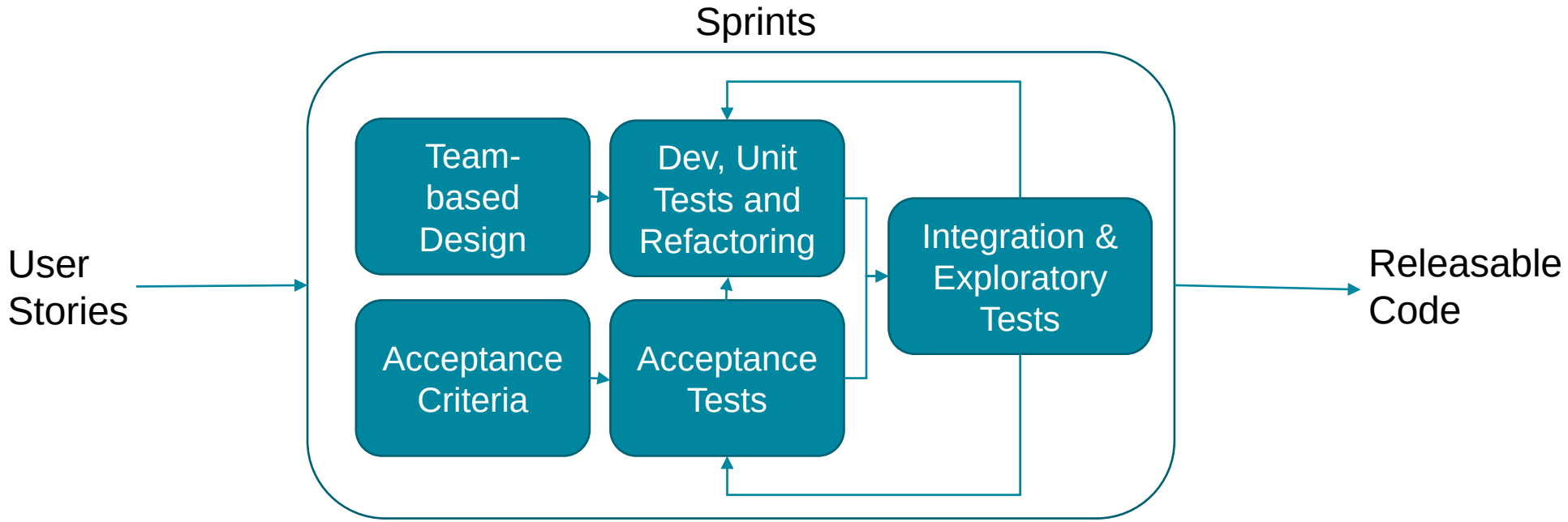
Scrummerfall – What do to?



Encourage team-based activities

- Restructure your activities to design, build, test as a TEAM
- Use behavioral-driven development (BDD) to align the business, development, and testing upfront in each sprint
- Whole team quality = if everyone's not done, no one is done

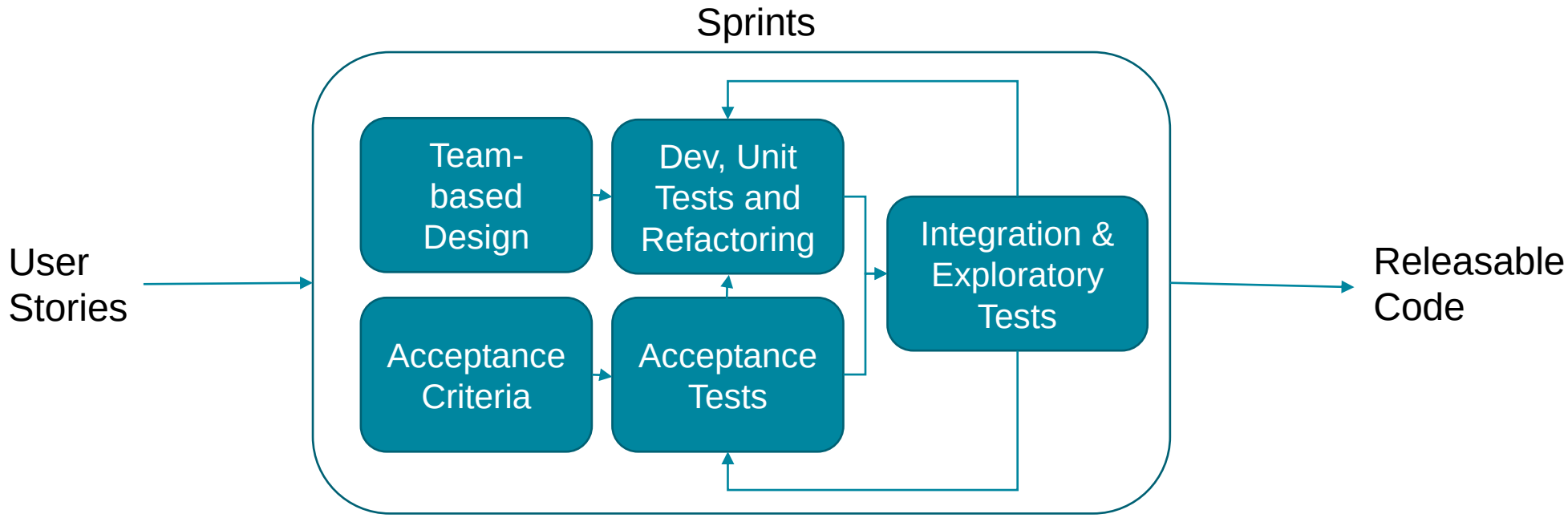
Scrummerfall – What do to?



Quantify success and hold to it

- Define a quantifiable definition of done (DoD) for stories and sprints
- Move toward a DoD for producing releasable code instead of just working code
- Establish quality gates for the entire delivery process

Scrummerfall – What do to?



Pair developers and testers to reinforce collaboration

- Create story acceptance tests together upfrontt
- Review eacother'srs tests for completeness
- Pair program occasionally
- Pair test occasionally

Reason #2: Lack of Automation

Without investing in some amount of automation, working software will not be produced during sprints

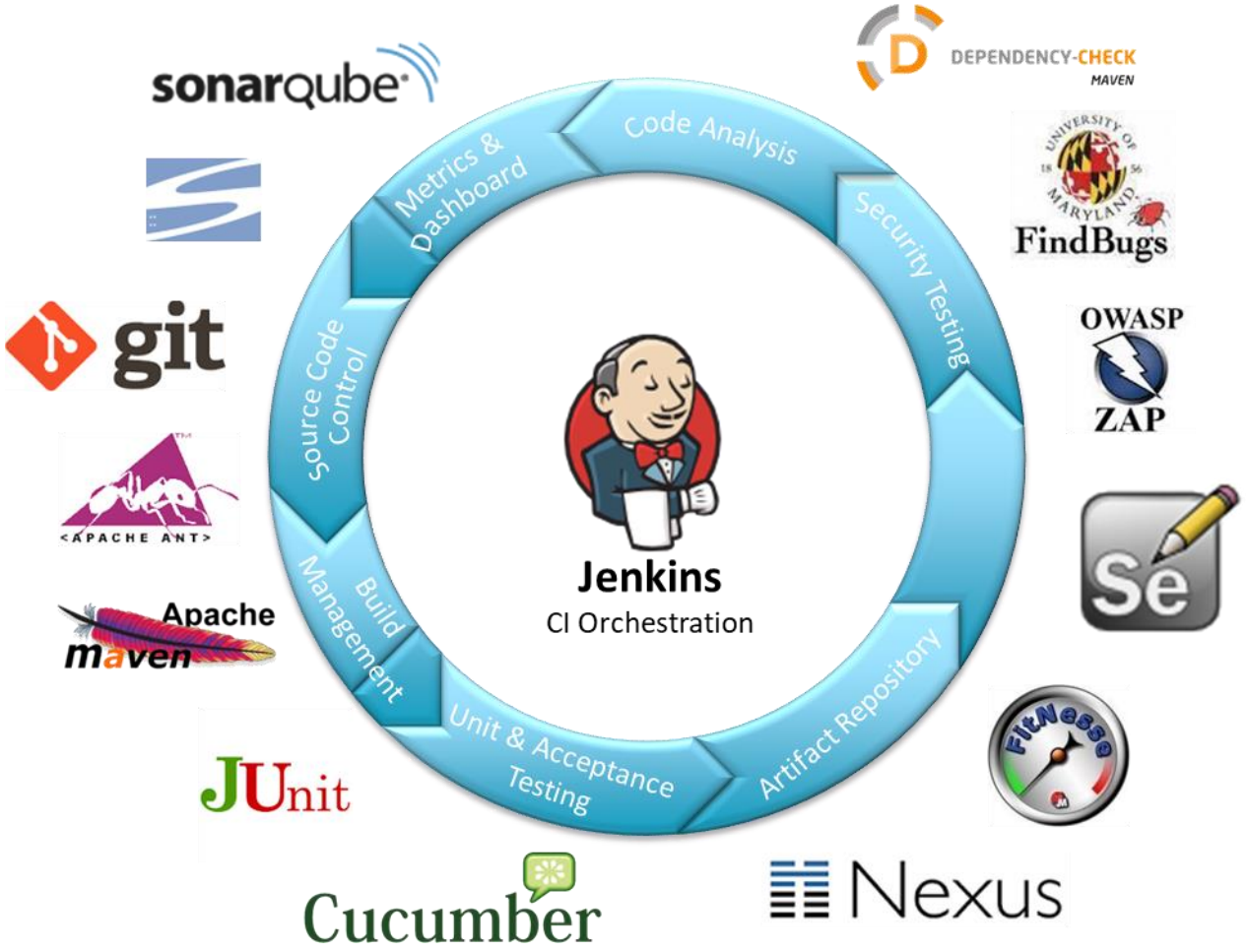
Lack of Automation – Agile is like a Russian doll

- Early on sprints are completed successfully
- As software grows, testing and integration challenges grow
- Teams are forced to elongate sprints or shift to Scrummerfall
- Teams deliver software that isn't fully tested

Lack of Automation – What to do?

Adopt continuous integration

- A robust automated continuous integration capability is essential
- Your branch/merge strategy must align with CI
- Integrate lightweight security and performance analysis as part of builds



Lack of Automation – What to do?

Automate Toil

- Unit testing
- Regression testing (below UI!)
- Smoke testing
- Environment setup/teardown
- Security testing and code scanning

Reason #3: Ever Changing Requirements

While agile encourages change, no software development approach can handle instability.

Ever Changing Requirements – What to do?

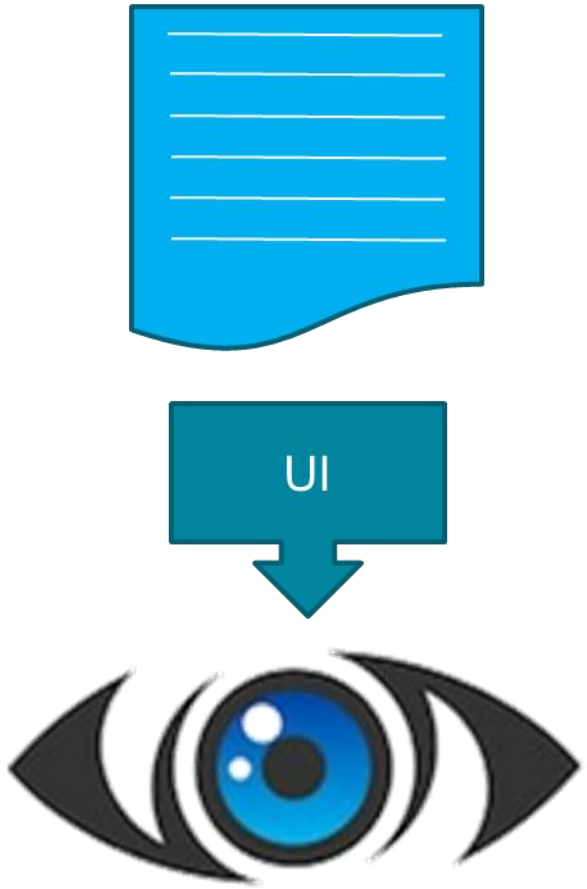
Strengthen your Product Owners

- There can be many stakeholders but only one product owner
- Stakeholders give opinions, product owners make decisions
- If your product owner is too busy to support your team(s), identify business analysts to help

Ever Changing Requirements – What to do?

Make Requirements Visible

- Customers don’t know what they want until they see it
- A Sprint is just a requirements process
- Get to something visible as fast as possible
- Mock/prototype screens prior to sprints



Ever Changing Requirements – What to do?

Get To The End Customer

- Use focus groups to validate product features
- Work through customer services to find customers with strong opinions
- Establish a ‘continuous beta’ program
- Release frequently!

Case Study #1 – frAgile results

Results:

- Application business layer has 5-7 levels of services
- Only 12 concurrent users can access the system
- Only 6 users get through the registration process on day one
- Releases take days or weeks to successfully complete
- Contractor is fired
- Congressional hearings are held

Large-scale Healthcare System

What would you do?

If you stepped into this project, what else might you do to fix it?

Case Study #1 – Epilogue

A year later:

- Product is re-architected by tiger teams
- System supports 250,000 concurrent users
- 500K – 1M registrations processed per day
- DevOps automation put in place to support ‘push button’ deployments
- Testing team integrated into agile development

Large-scale Healthcare System

Case Study #2

Scenario:

- Old-school financial services organization decides to embrace agile
- Senior management does not socialize change or define roles
- Organization believes agile comes “for free”
- Scaling starts immediately through agile coaches and training for all teams
- Company declares itself the ‘Spotify’ of financial services
- Senior mgmt. drives maturity through bonuses

Reason #4: Lack of management support

Without adequate ‘air cover’ for agile initiatives, your success rate will plummet

Lack of management support – What to do?

Education Senior Leadership

- Focus on values and principles
- Focus on agile leadership training & coaching for middle management
- Set appropriate expectations for change
- Continually report on progress and repeat the vision over and over

Lack of management support – What to do?

Use Demonstrations To Build Awareness

- Demonstrate tangible progress at the end of every sprint
- Schedule demos with stakeholders and others at least once per month to increase awareness and buy-in
- Share successes with other teams to grow team support

Lack of management support – What to do?

Advocate for Agile

- Share successes with mgmt. as frequently as possible.
- Track and make visible metrics
- Seek to constantly educate on new concepts and ideas
- Use case studies and experts to dispel myths

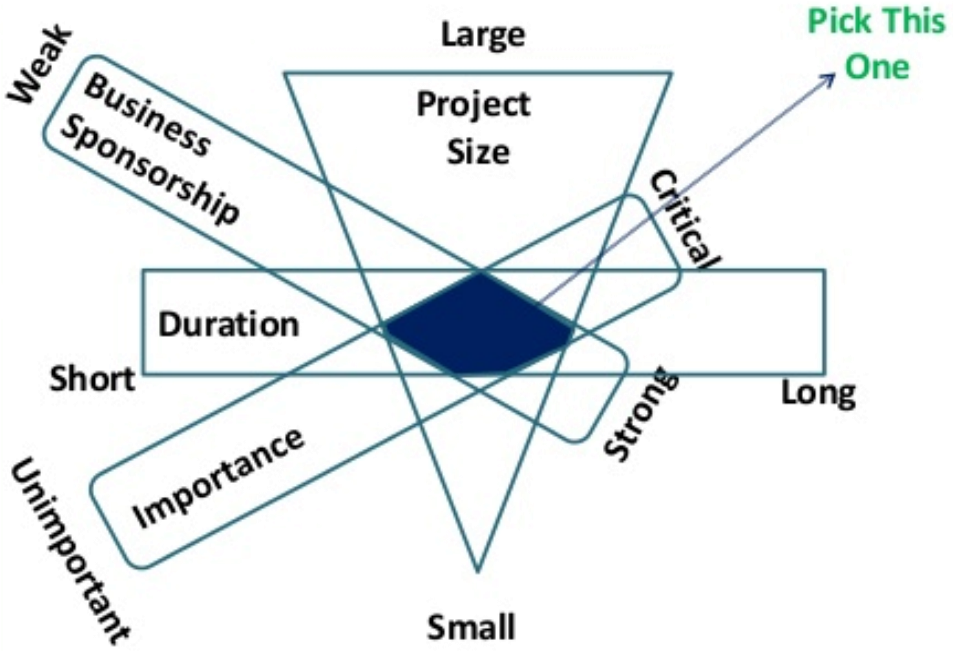
Reason #5: Adopting Agile Too Fast

Slow is smooth, smooth is fast

Adopting too fast – What to do?

Pilot Change

- Start agile adoption by choosing the right agile pilot(s)
- Select pilot team carefully as you want them to become your internal agile champions
- Share pilot progress and successes constantly



Adopting too fast – What to do?

Build Working Code Before Scaling

- Make sure your pilot teams are delivering working code nearly every sprint before trying to scale
- Scale from the bottom up as needed instead of using a top to bottom framework
- Scale iteratively just like everything else you do in agile

Reason #6: Too Self Directed

Self-directed does not mean unmanaged

Too Self Directed – What to do?

Create a Center of Enablement

- Transform an existing PMO to an agile PMO
- Focus of each is on enablement, coaching, traand ining, support with lightweight agile governance
- Define guardrails

Too Self Directed – What to do?

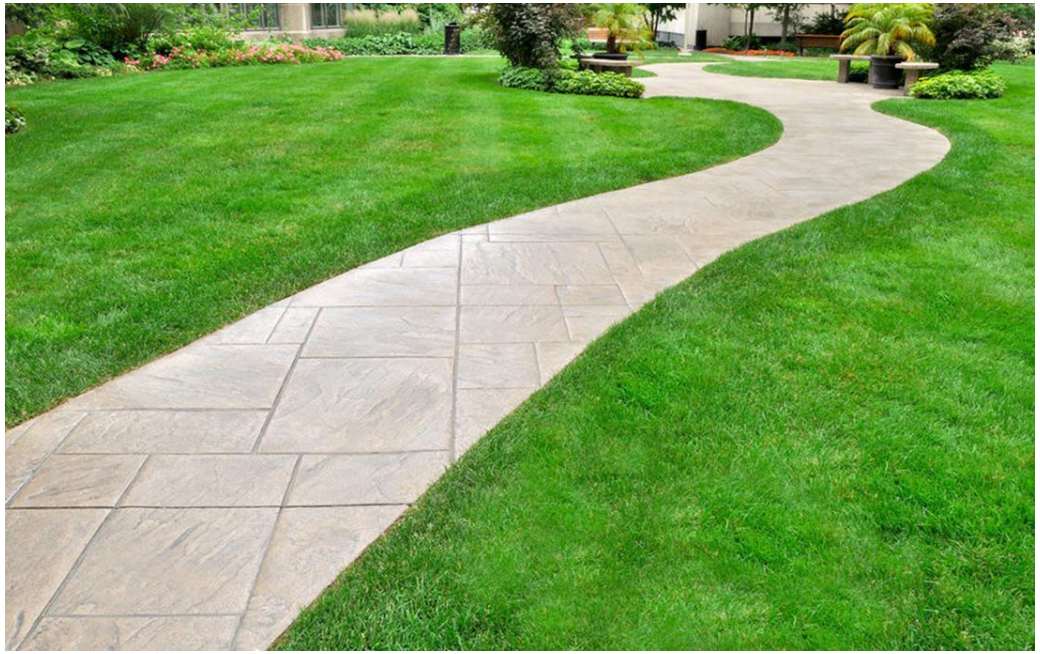
Create Agile Playbooks

- Define best practices for those agile techniques many in your organization find of value
- Provide training and coaching around their use on projects
- Build a Dojo to support learning

Too Self Directed – What to do?

Create Paved Paths

- Provide teams with tooling that the enterprise supports through licenses, training, coaching
- Incorporate paved paths into COE
- Iterate on Paved Paths as needs change



Case Study #2 – frAgile results

Results:

- Middle management attempts to shift the organization back to the original dev process
- Agile coaches are fired if they push back on non-agile practices
- Teams now focus on ways to argue for better maturity scores than working to improve
- Agile center of excellence loses the ability to drive agile improvements

What would you do?

If you stepped into this project, what else might you do to fix it?

Case Study #2 – Epilogue

Still To Be Written

QUESTIONS?

Thank you!

coveros