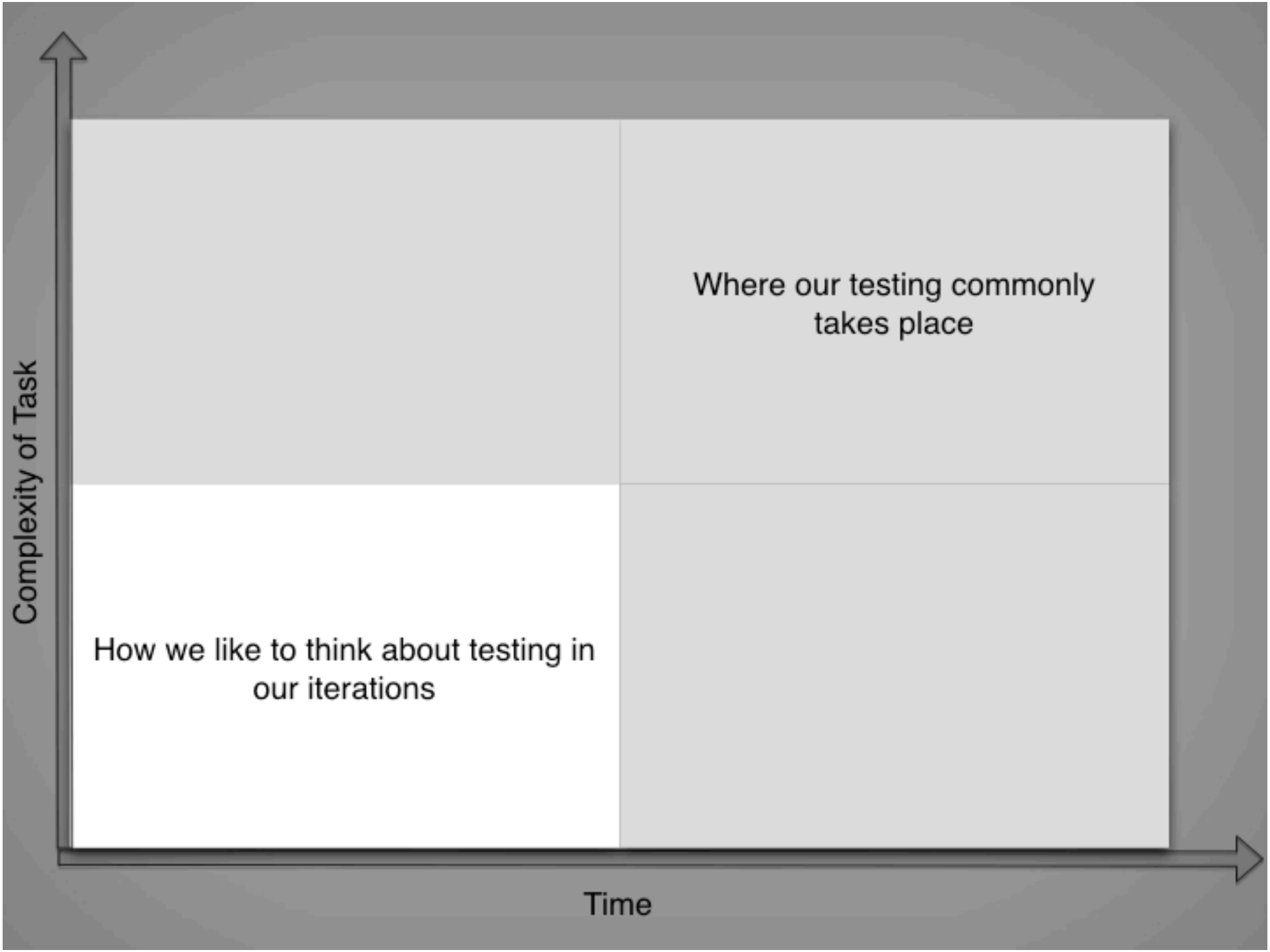
Working Testing Tasks into the Product Backlog

Michael Kelly
@michael_d_kelly

Testing Complexity on Agile Projects


Where our testing commonly takes place
How we like to think about testing in our iterations
Complexity of Task
Time

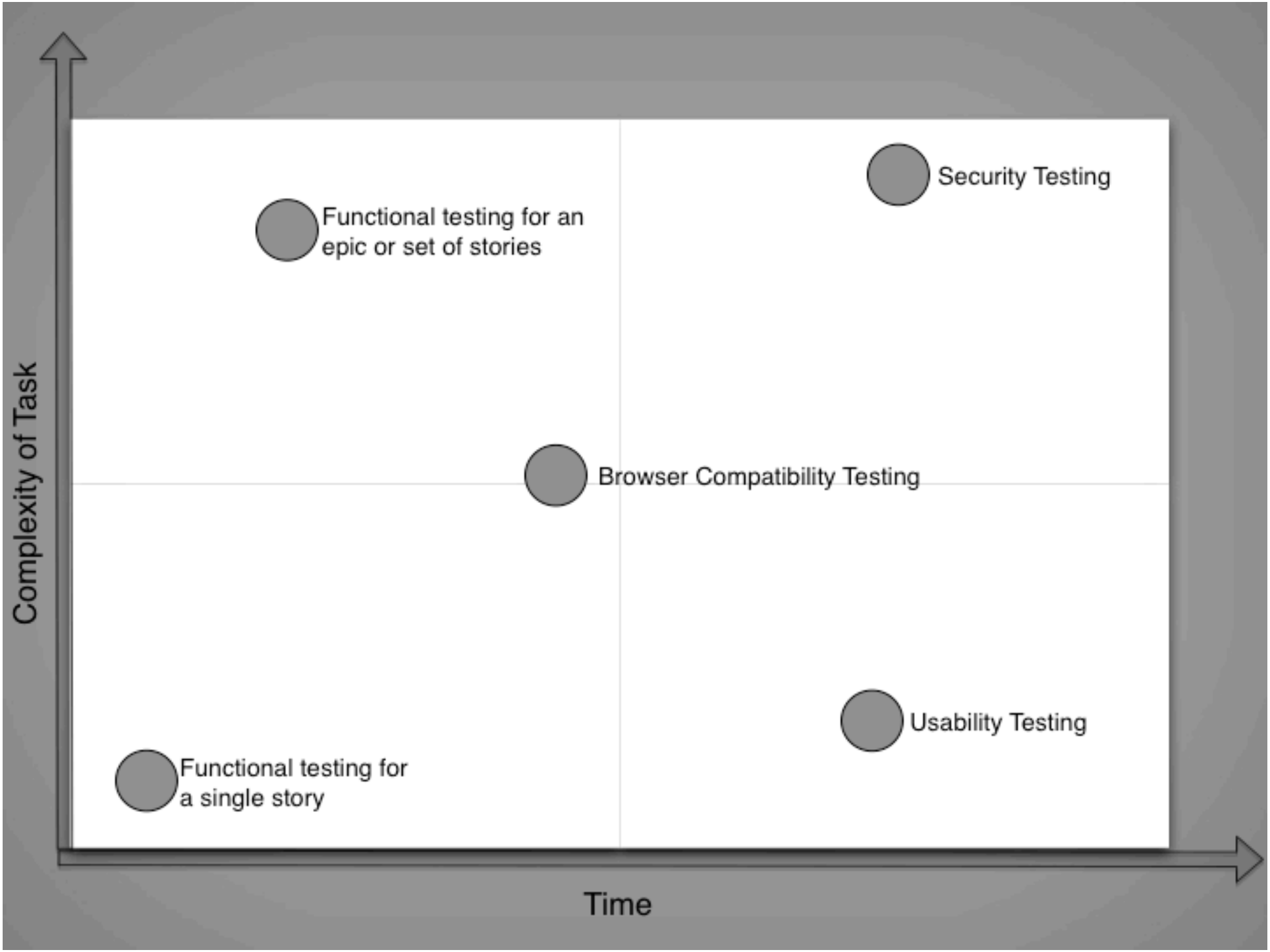
More complex

- Feature testing for non-trivial functionality
- Testing complex user-interface interactions
- Products with many integrations to other systems
- Real-time systems
- Performance testing
- Security testing

Take more time

- Compatibility testing
- Internationalization testing
- Usability testing
- Alpha/Beta testing
- Certification testing
- Performance testing
- Security testing
- Configuration testing
- Regression testing *

* Can often take longer than a single iteration on large projects.


Security Testing
Functional testing for an epic or set of stories
Browser Compatibility Testing
Usability Testing
Functional testing for a single story
Complexity of Task
Time

In addition to complex testing, there are complexities and issues that come with an emergent architecture and design

Sprint 1: We lay some foundational code, getting scaffolding in place, basic templates, and we get tools like CI and basic code coverage in place.

Sprint 2: We start to lay in some core features. We're focused on getting enough functionality that we can release to production and start the "release to prod at the end of every sprint" cycle.

Sprint N: We deploy our first usable code to production, and we get users on it so we can get feedback.

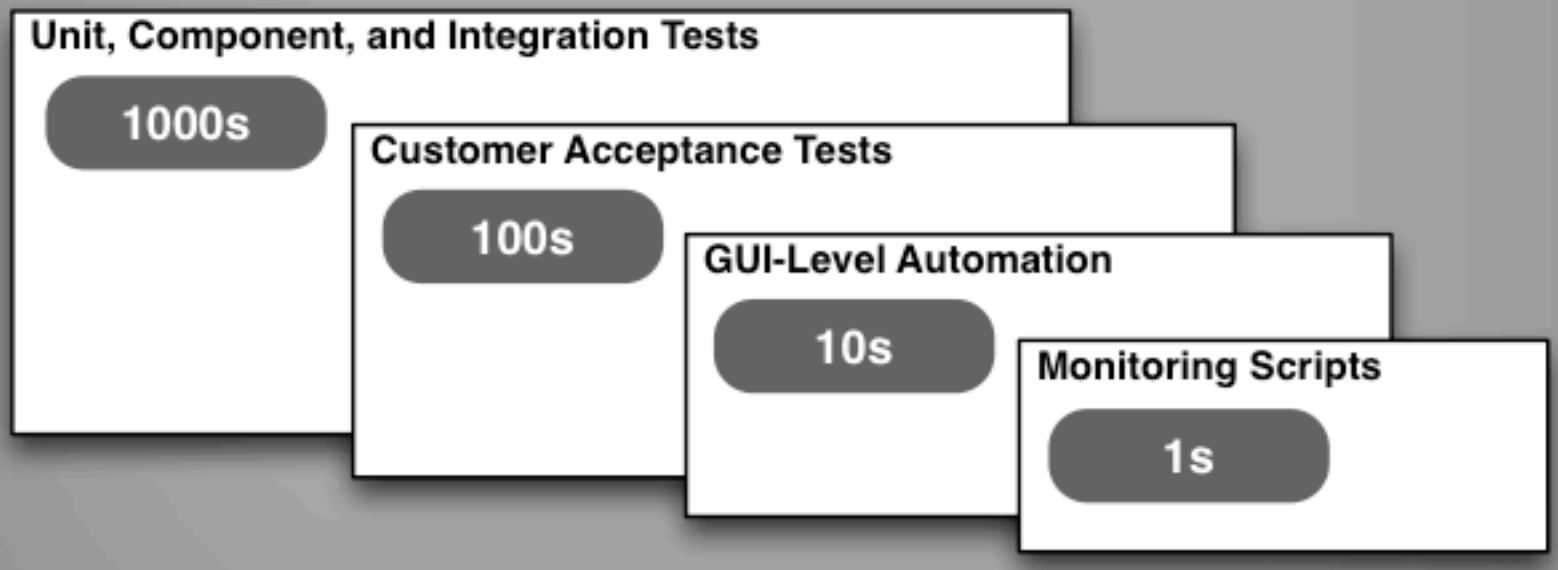
Sprint N+1: We're iterating on customer feedback and stories for new features.

Sprint N+5: We find that some core piece of functionality (our homepage, search, or some other feature) is dog slow for some reason. It happened over time, but it's finally gotten to the point where the team writes a story to crush the problem.

Sprint N+6: We're rolling smooth again in production, on to more stories.

Managing the Complexity

Test Automation



Better Tooling

- Increased specialization by individual products (instead of “enterprise” solutions)
- Better test case/coverage management
- Better visibility into the product (analytics, monitoring, alerting, etc)
- Better environment management capability

Outsourcing

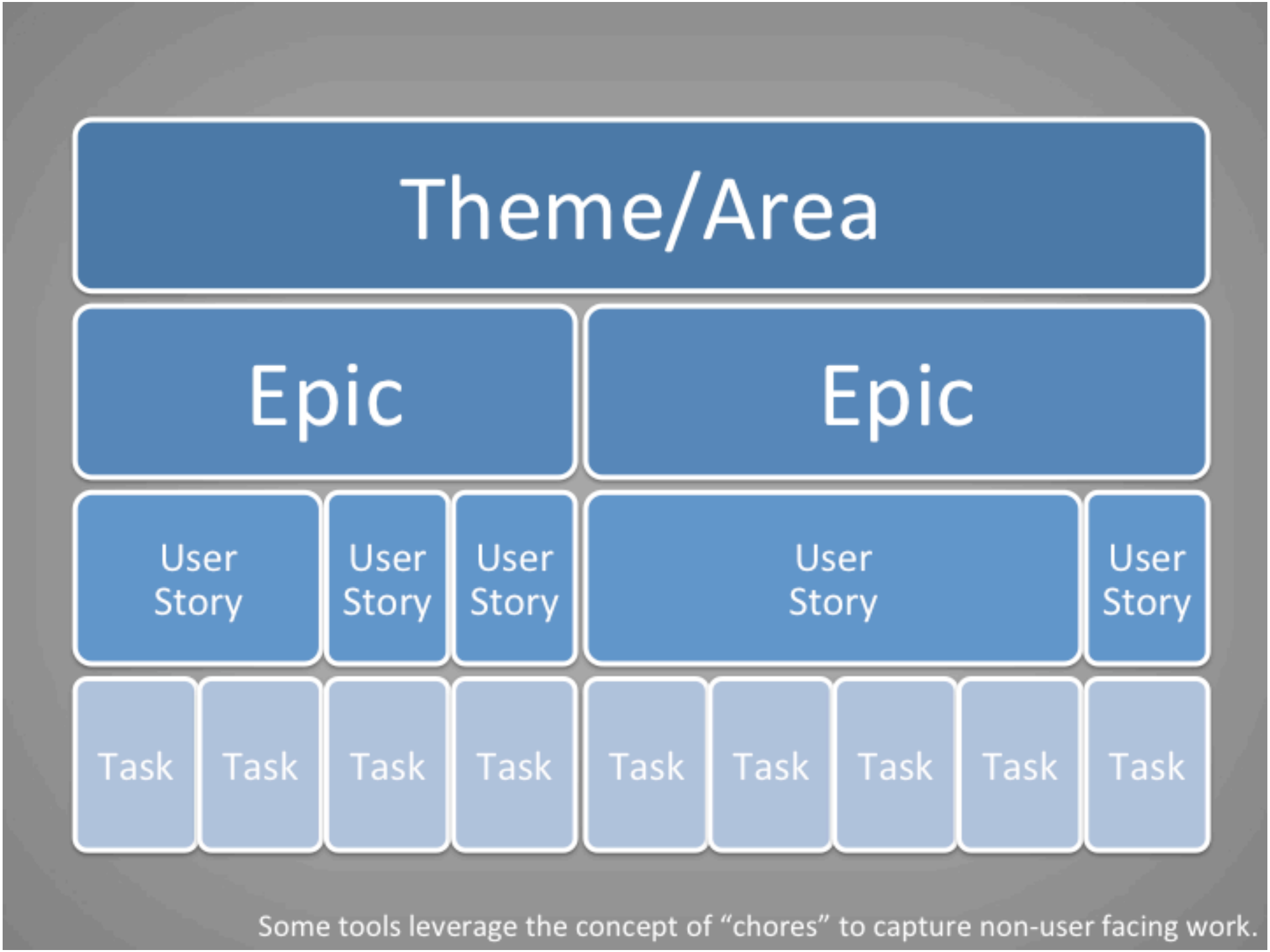
- Partner with test labs or consultants that specialize in:
  - Performance
  - Usability
  - Security
  - Compatibility (mobile or browser)
  - Etc.

Transition Iterations

Iterations focused on

- “hardening” activities such as bug-fix and testing
- deployment activities like environment setup and documentation
- communication activities like messaging and training

Layering Your Testing


Theme/Area
Epic
Epic
User Story
User Story
User Story
User Story
User Story
Task
Task
Task
Task
Task
Task
Task
Task
Task
Some tools leverage the concept of “chores” to capture non-user facing work.

Testing by the Story

- Testing is a task (or set of tasks) on each story
  - developer tests
  - customer acceptance tests
  - possibly additional functional tests/charters
  - possibly service-level or user interface automation
- More complex testing tasks can be separate stories
  - testing in a specific environment with seeded data
  - testing for specific configurations or data sets

Testing by the Epic

- Write stories to test the interaction of features delivered under the epic:
  - traversing use-cases or sequence diagrams (automated or manual)
  - large or complex data sets
  - system events and/or calendar events (daily, weekly, monthly, quarterly, etc)
  - take into account operations (analytics, monitoring, alerting)
  - complex 3rd party integrations and/or dependencies
  - take into account maintenance/setup activities (archiving/deleting, deployment/rollback, etc)
- Write stories to test specialized types of testing that only make sense to perform at the Epic level
  - usability
  - performance
  - security
  - failover/recovery
  - etc.

Testing by the Theme

- For themes targeting "improvement over time" - areas like performance and usability - you might have a number of stories/tests that build on one another that you execute over time:
  - establish a baseline of data via a small focused test bed
  - perform specific tuning activities
  - rerun baseline test and compare
  - slightly increase the core test-bed adding new tests and baseline data
  - repeat, over time growing the test suite and available data under that theme

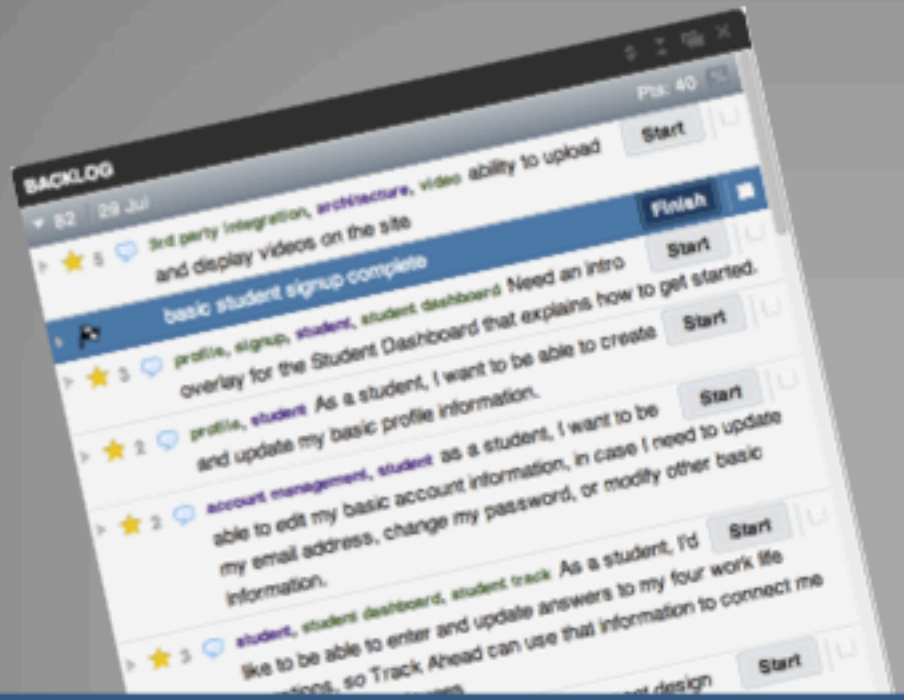
Roadmaps and Backlogs

Sprint Planning using Pivotal Tracker	
Done	What I did
Current	What I am doing
Backlog	What I will do
Icebox	What I might do

A place to start:

- Develop concept-level designs
- Leverage designs to build epic-level Icebox
- Pull epics into Backlog, breaking each epic into stories
- Stories can either be:
  - Specific to a discipline*
  - Span multiple disciplines* using tasks
- Leverage placeholder stories liberally

* Common disciplines include design, development, testing, and infrastructure



- Leverage the Backlog to develop a visual roadmap of the project with key dates/milestones shown
- To the best of your ability, plan for periodic releases to production with incremental value

Basic User Roles, Signup, and Account Admin Features
Mar 12, 2012

Ability to:
- signup as a student
- add comapny admins
- add employees as a company admin
- assign TrackAhead admin role
- create a student profile
- update student account information
- update employee account information
- deactivate an account
- deactivate a company

Basic Student Track Functionality
Apr 9, 2012

Ability to:
- view my student dashboard
- watch company videos and answer questionnaires
- earn points as a student

Basic Employee Dashboard and Interactions with Students
Apr 23, 2012

Ability to:
- view an employee dashboard
- view the company leaderboard
- identify students for possible interactions
- earn points as a student or employee
- start an interaction
- evaluate my interactions

Reporting and Marketing Pages/Sites
May 7, 2012

Ability to:
- review recruitment reports
- track detailed site statistics

Websites for:
- marketing to students
- marketing to companies

MVP
May 14, 2012

Mar 14, 2012 | Mar 21, 2012 | Mar 28, 2012 | Apr 4, 2012 | Apr 11, 2012 | Apr 18, 2012 | Apr 25, 2012 | May 2, 2012 | May 9, 2012 | May 16, 2012 | Feature Sched

Before the start of each sprint

- Re-estimate “on tap” stories
  - breakout into smaller stories as needed
  - talk about and manage dependencies for design, development, infrastructure, and testing
  - identify new stories as needed based on our updated understanding of scope and risk
  - remove any placeholder stories if they still exist
- Review and update sprint priorities with product owner(s)

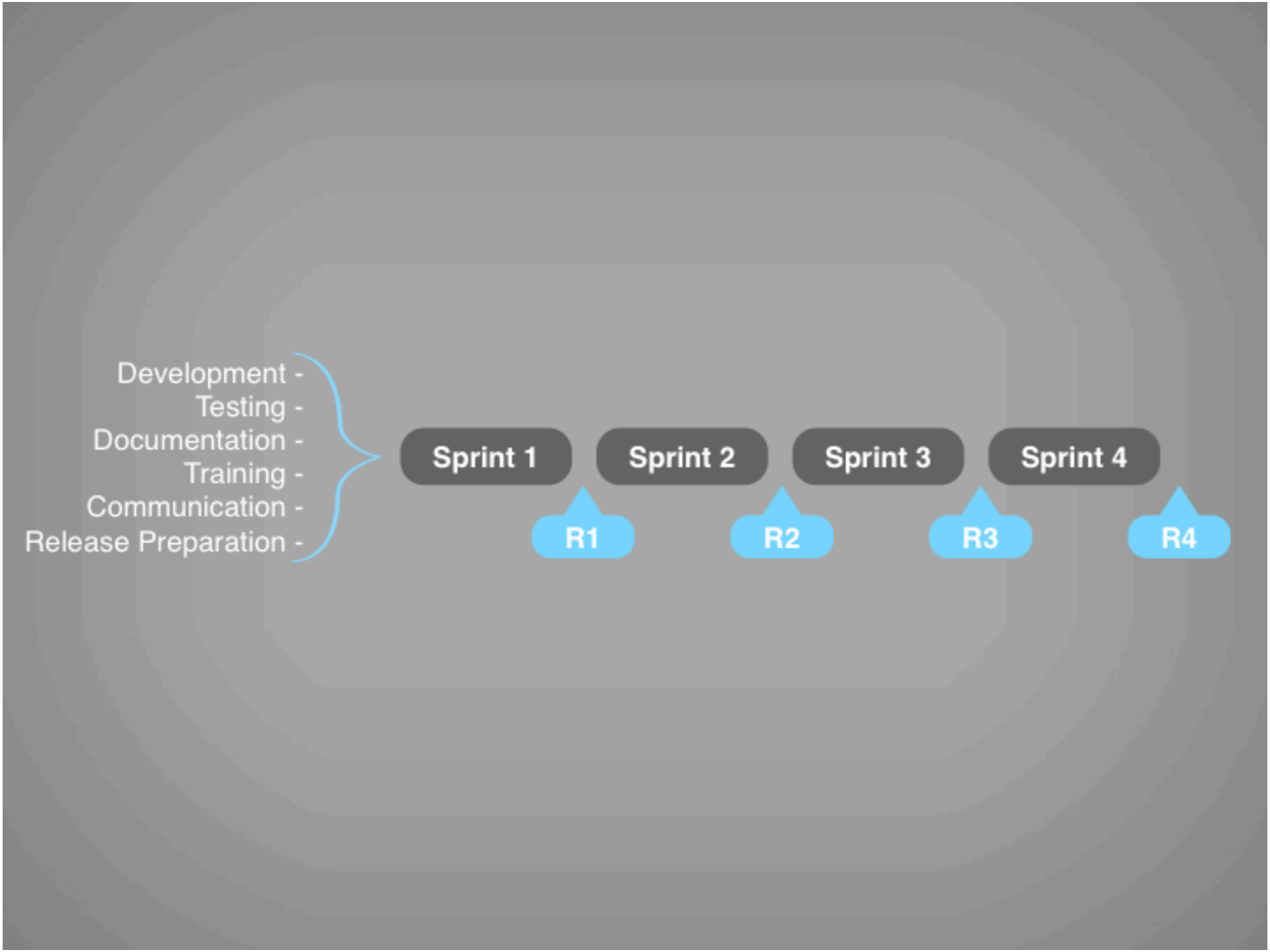
On a regular basis (monthly or quarterly-ish)

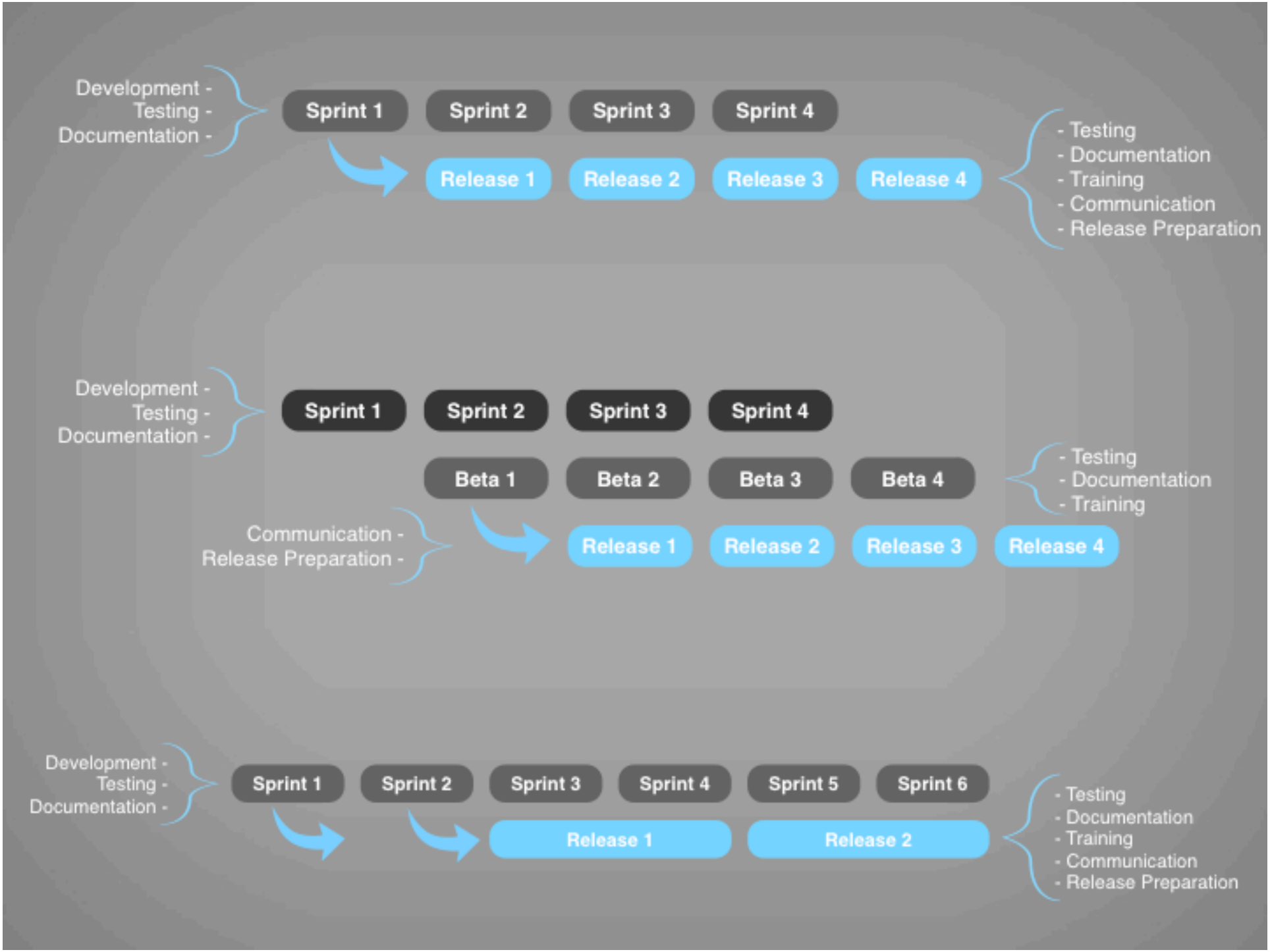
- Review the overall product Backlog for gaps based on current understanding
  - Identify new stories as needed
  - Remove stories no longer needed
  - Target placeholders to see if you can replace them with real stories
- Review Icebox to see if there are stories/epics that need to be moved into scope
  - Review newly created stories/chores/bugs to see if any of them need to be prioritized in the Backlog
  - Revisit themes to see if all the correct epics are still represented somewhere in the Icebox and/or Backlog
  - Perform initial story breakout for epics if moving them into the Backlog
- Reforecast the overall project roadmap and provide updated snapshot to the team and product owners (visual and/or Gantt)

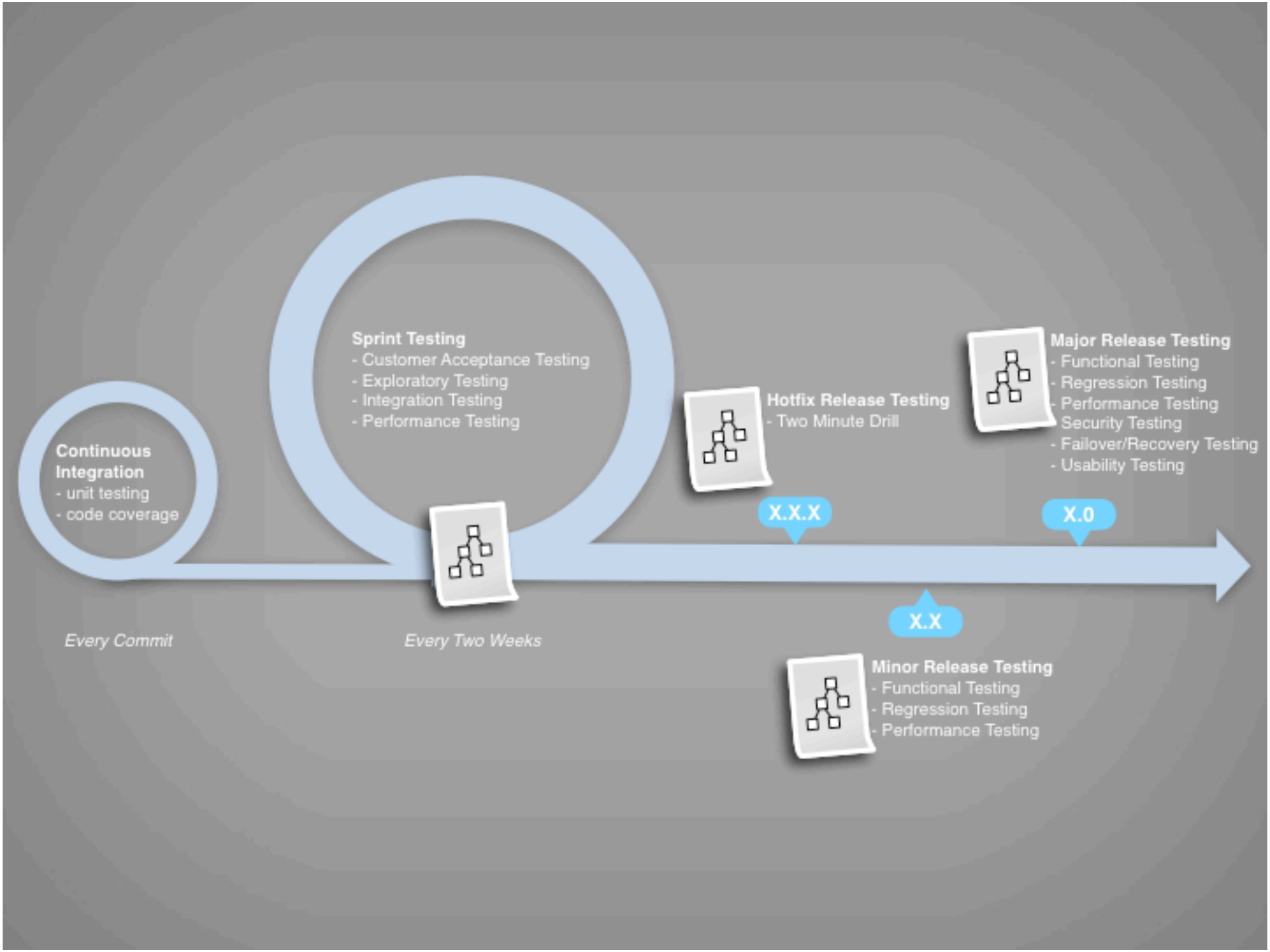
Someone on the team is…

- Managing testing risk and coverage
  - Understands what's been automated as part of regular regression
  - Curates defects, exceptions, user feedback, and other qualitative data to build a working model of product quality
  - Writes new testing stories as needed
- Coordinating who does what with regards to testing
  - Quarterbacks larger testing efforts for areas that span a single story (performance, security, usability, etc)
  - Manages any outsourced relationships when it comes to testing
- Advocating for testability
  - Identifies new tools, APIs, and features to facilitate easier testing
  - Facilitating and training others on how to better test
  - Researching and removing testing roadblocks as needed

Change the Way You Think About Releases


Development -
Testing -
Documentation -
Training -
Communication -
Release Preparation -
Sprint 1
Sprint 2
Sprint 3
Sprint 4
R1
R2
R3
R4


Development -
Testing -
Documentation -
Sprint 1
Sprint 2
Sprint 3
Sprint 4
Release 1
Release 2
Release 3
Release 4
- Testing
- Documentation
- Training
- Communication
- Release Preparation
Development -
Testing -
Documentation -
Sprint 1
Sprint 2
Sprint 3
Sprint 4
Beta 1
Beta 2
Beta 3
Beta 4
- Testing
- Documentation
- Training
Communication -
Release Preparation -
Release 1
Release 2
Release 3
Release 4
Development -
Testing -
Documentation -
Sprint 1
Sprint 2
Sprint 3
Sprint 4
Sprint 5
Sprint 6
Release 1
Release 2
- Testing
- Documentation
- Training
- Communication
- Release Preparation


Continuous Integration
- unit testing
- code coverage
Every Commit
Sprint Testing
- Customer Acceptance Testing
- Exploratory Testing
- Integration Testing
- Performance Testing
Every Two Weeks
Hotfix Release Testing
- Two Minute Drill
X.X.X
Minor Release Testing
- Functional Testing
- Regression Testing
- Performance Testing
X.X
Major Release Testing
- Functional Testing
- Regression Testing
- Performance Testing
Security Testing
- Failover/Recovery Testing
- Usability Testing
X.0

Release-Driven Testing

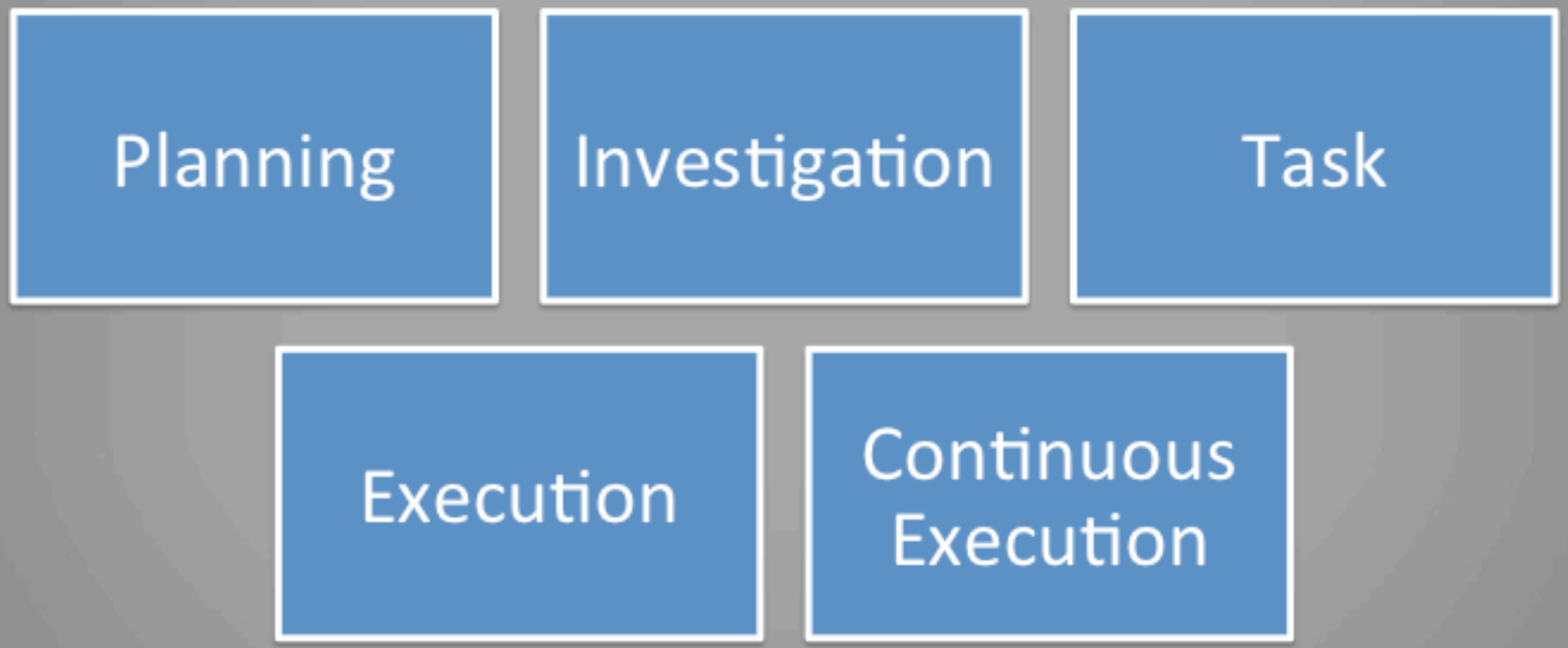
- Can create “space” for testing in tight iterations, while preserving the power of “small batches”
- Can remove some testing from the backlog entirely by creating regular “release-based” testing activities
- Can facilitate release criteria related to test coverage targets/goals

The Stories

Testing Stories

- Should have clear objectives for what value they are attempting to deliver
- Don't have to be framed as value to some specific user
- Can sometimes be time-boxed
- Should have clear deliverables that can be accepted or rejected
- Should be estimated by the entire team, just like any other story
- May have (and often will have) dependencies on other stories

Types of Testing Stories/Chores



Planning

- **Objective:** Develop a plan or strategy for some aspect of testing relating to the project
- **Deliverables:** Common deliverables might include:
  - a document or wiki page for review
  - a collection of estimated stories/chores that support the plan
  - a possible timeline to support the plan
- **Examples:** The following might be some example planning stories/chores:
  - Develop a performance testing plan for MVP launch
  - Develop a usability test plan for student dashboards
  - Develop a plan for how we will coordinate test environments and data with clients X, Y, and Z

Investigation

- **Objective:** Time-boxed investigation/research of some aspect of testing relating to the project
- **Deliverables:** Common deliverables might include:
  - a brief summary of investigation/research performed
  - a summary of findings
  - a recommendation
- **Examples:** The following might be some example investigation stories/chores:
  - Research compatibility testing vendors capabilities and pricing and make a recommendation
  - Research Android user market and provide a recommendation on initial test target coverage requirements
  - Figure out how to mock test data for service XYZ

Task

- **Objective:** A discreet task (or collection of tasks) to accomplish related to testing
- **Deliverables:** Common deliverables might include:
  - the setup or configuration of a tool, environment, or data
  - the automation of a test or set of tests
  - the development of some key artifact to support a broader test effort (usage model, exit interview, coverage matrix, etc)
- **Examples:** The following might be some example task stories/ chores:
  - Automate regression tests for XYZ
  - Develop initial usage model for customer signup
  - Develop demographic specific questionnaires for usability testing
  - Develop coverage matrix for customer billing entries by product type, geographic region, and payment terms

Execution

- **Objective:** Execution of a discreet set of test activities. I think of this as executing a charter in Session Based Test Management.
- **Deliverables:** Common deliverables might include:
  - defects
  - a summary of test results
  - a summary of testing performed
  - follow up test stories or product enhancement requests
- **Examples:** The following might be some example execution stories/ chores:
  - Test for deliverability related issues related to product emails
  - Test for data accuracy issues related to calculations on service and fundraising summary stats
  - Test for data accuracy issues related to Piggy Bank commitment calculations
  - Test for performance related issues related to mapping capabilities
  - Perform a copy review of the various descriptive dialogs in the app looking for typos and grammar issues

Continuous Execution

- **Objective:** Execution of a set of test activities that support a larger testing effort that spans multiple iterations. This is commonly done for things like performance or usability testing. You can think of this as testing work that's "never done."
- **Deliverables:** Common deliverables might include:
  - defects
  - a summary of test results
  - a summary of testing performed
  - updated planning or test documentation
  - updated baseline information
  - follow up test stories or product enhancement requests
- **Examples:** The following might be some example execution stories/chores:
  - Regression test the performance of XYZ, updated for new ABC functionality
  - Review beta test feedback and product analytics for ABC and conduct follow up interviews as necessary
  - Bulk test policies X,000 through Y,000 (load policies into the test environment for processing, review resulting exceptions and errors, aggregate issues into the smallest set possible and log them for remediation)

Questions?

www.MichaelDKelly.com
@michael_d_kelly