ATDD

What are we Going to Talk About

- Something Something Driven Development
- Evolution of a Process
- To ATDD or not to ATDD, Is That a Question?

Acronym Confusion

BDD

- Behavior Driven Development
- Business Driven Development

Acronym Confusion

ATDD

- Acceptance Test Driven Development
- Automated Test Driven Development

AATDD?

- Automated Acceptance Test Driven Development

Over the Wall

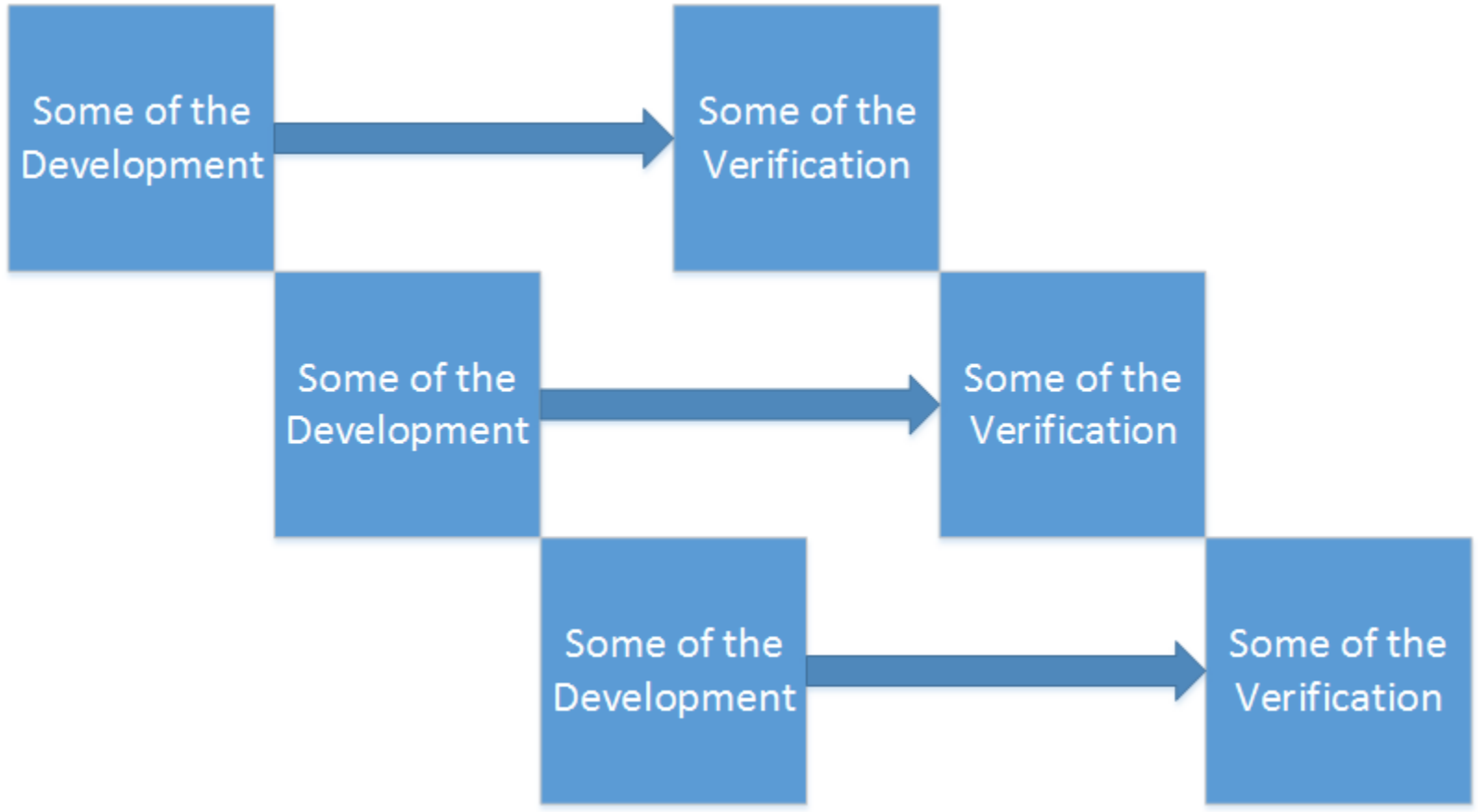
What happens if your mom asks you to clean the house while she's at the store?

You get grounded!

Over the Wall

- At least we're testing, right?
- No chance to learn.
- When will we be done testing?
  - When we hit the release date.

SCRUMFall

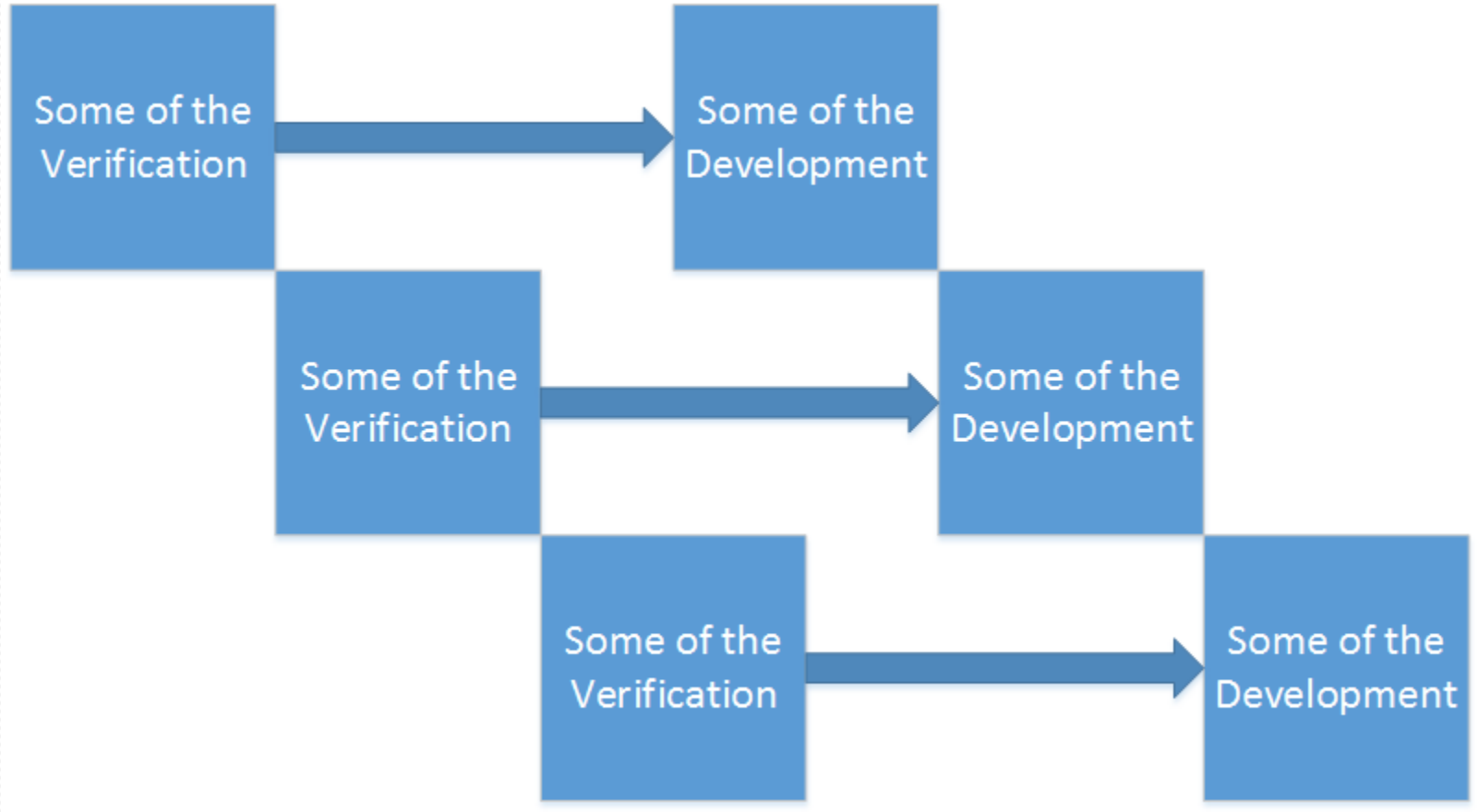

Some of the Development
Some of the Verification
Some of the Development
Some of the Verification
Some of the Development
Some of the Verification

SCRUMFall

- Opportunity to integrate feedback and improve
- Might still spend time building something we don't understand

- You're getting feedback after you clean each room.

ATDD


Some of the Verification
Some of the Development
Some of the Verification
Some of the Development
Some of the Verification
Some of the Development

ATDD

Mom: Son, can you clean the house while I'm at the store, please?

You: Sure, Mom. What cleaning will you find acceptable?

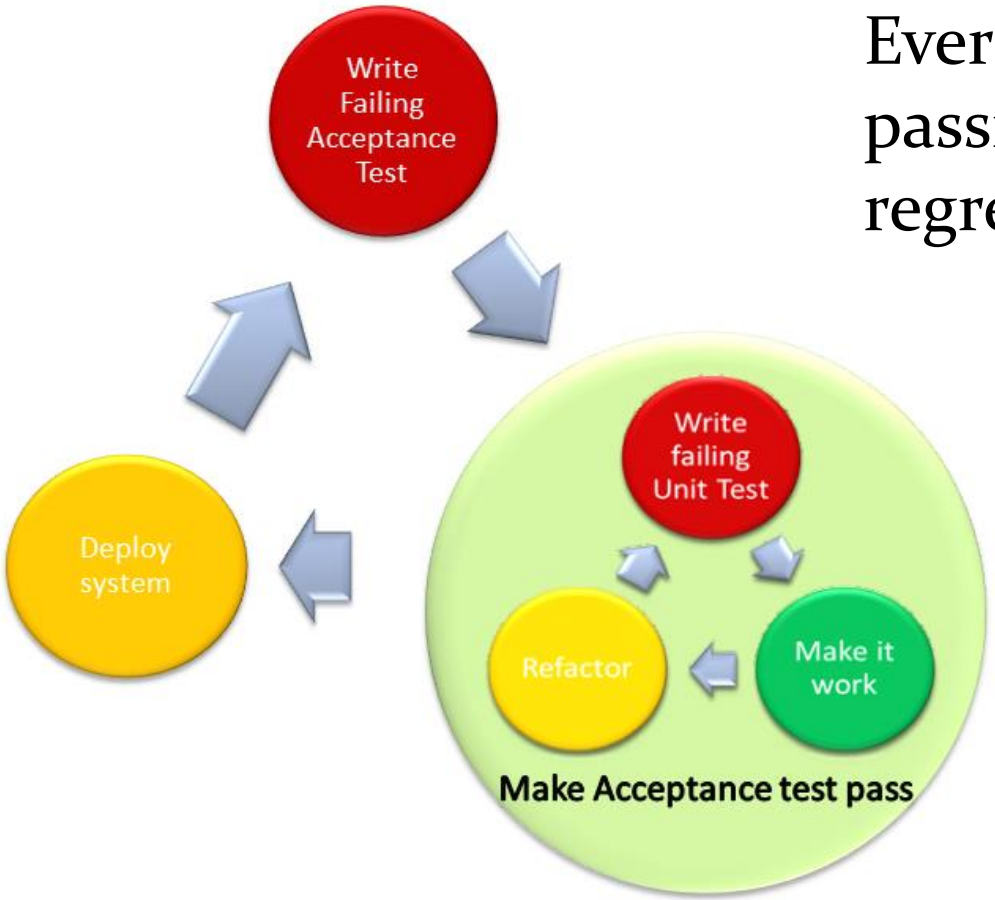
...Later...

Mom: Oh, Son, the house looks great. It's exactly what I expected. Here's the car keys and some gas money. Go have fun, dear.

History

- 1960s - Used by NASA for project Mercury
- Late 1990s and early 2000s – Various improvements, specifically around specification by example and automation
- Late 2000s - BDD become more formally defined and used as a practice
- Around 2010 - Acceptance tests evolved from BDD and tools to support automation gain maturity and begin widespread adoption

Integrating with Development



Ever-growing suite of passing tests are used as regression tests

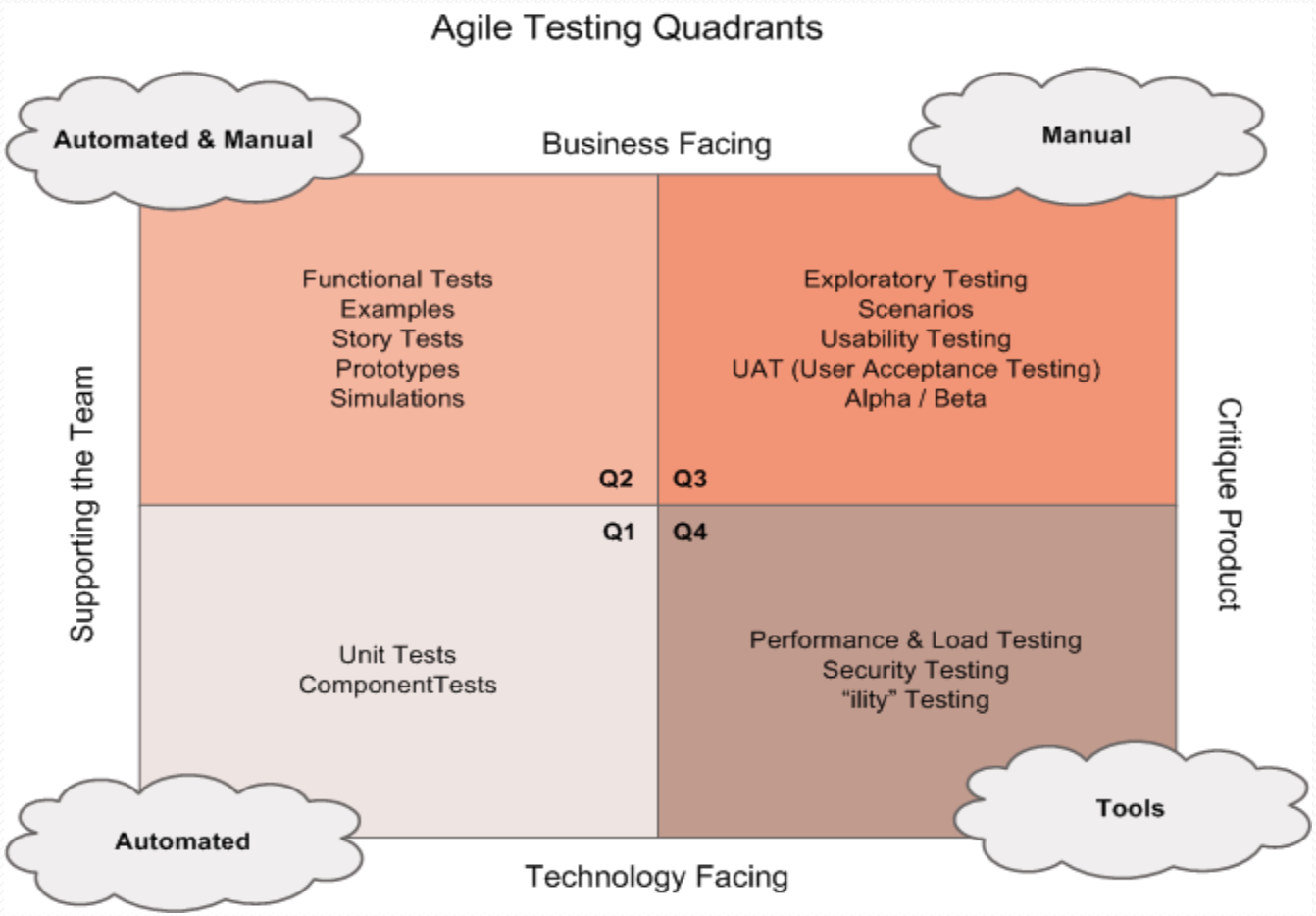
Should We Automate

MAYBE?!?!

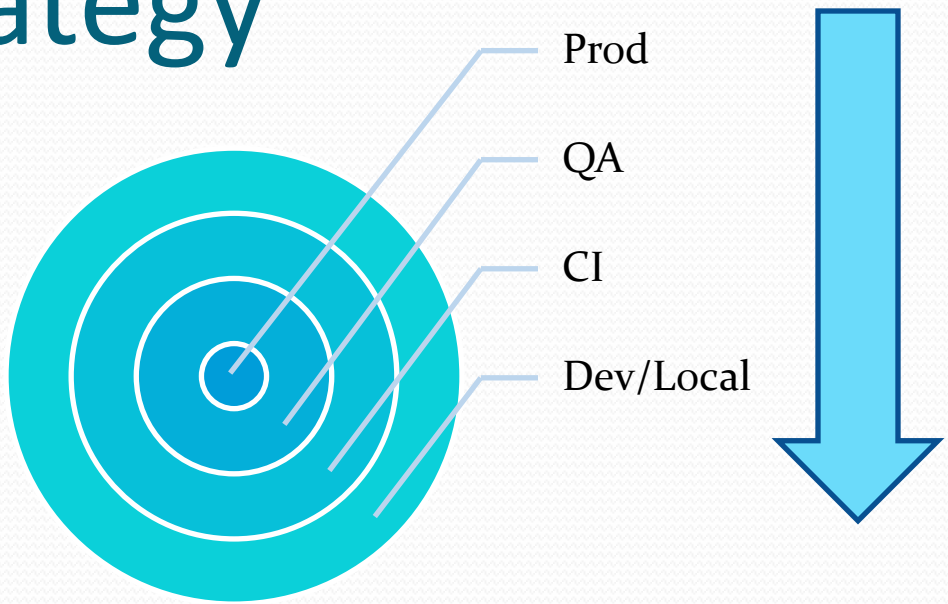
Should We Automate

- Difference between automating and utilizing automation
- Can we support the automation in the long term
- Decision for each test

Test Strategy



Test Strategy



- Build tests for how you will validate a production deployment
- Finish with tests around fine grained functions that can only run locally

What's the Process Again?

Gather Requirements	• Epics, stories, tasks, notes • Acceptance criteria
Have a Conversation	• Affirm understanding • Make sure the team is on the same page
Define Done	• What's the QA strategy for the piece of work • Use the test quadrants as a guide
Write tests	• Automate what you can, write manual tests • Create non-functional tests, get test data
Build the system	• Developers using the tests to drive their work
Perform any remaining QA	• Manual, non-functional, exploratory, etc. • Get user feedback

ATDD Benefits

- Clear definition of done
- Fewer debates about delivered vs. expected functionality
- Quickly understand scope
- Less likely to build unrequested features
- Spend time preventing defects rather than fixing them
- Reduces pre-implementation defect triage
- Quickly discover areas of misunderstanding

Not a Silver Bullet

- You might still build the wrong things (though they will be well tested!)
- Does not replace a testing strategy
- Does not eliminate project planning
- Requires high level of communication
- Does not eliminate meetings
- Still need to write very good tests
- Does not eliminate system or data dependencies

Barriers to Entry

- Need to reverse the development process, creation of failing tests must come first
- Developers need to have ability to run tests during development
- Perceived cost increase, increased visibility into cost of testing
- Typically involves automation, may need to augment skills of team members
- Reliance on record-playback tools
- Resistance to change
- Previously failed adoption attempts

Measuring Improvement

- Number of defects found in production
- Number of design defects
- Number of testing cycles
- Time spent waiting for tests to be written or test cycles to complete
- Ramp-up/onboarding time for new team members
- Time spent clarifying requirements after completion of development

Smoothing out the Bumps

- Let the customer drive the process, keep them involved throughout
- Have developers and testers working at the same level of granularity
- Automate to improve efficiency and feedback cycles
- Build smoke tests and run them often
- Have developers and testers work in pairs
- Address data and system dependencies
- Stop thinking of testing as a project phase