



# PNSQC

OCTOBER 9-11 2023

## AMP IT UP:

### TRANSFORMING QUALITY

**Brent**

**Clausner**

*Software Continuous*

*Integration with Hardware*

Copyright 2023 Carnegie Mellon University.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-05-2-389 D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at [permission@sei.cmu.edu](mailto:permission@sei.cmu.edu).

Carnegie Mellon® is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.  
DM23-0885



# Hello

- Who I am
- Introduction
- DevOps Current Practices
- Software for Hardware
- Emulation
- Code Branches
- Drawbacks
- Wrap up



# Myself

- Education
  - PTI
  - Point Park University
- Work
  - DevOps Engineer
- Hobbies
  - Video Games
  - Jogging
  - Technology

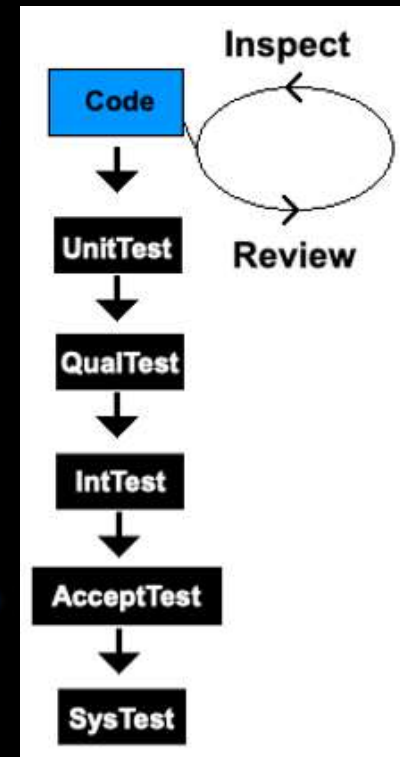


# Introduction

- Testing is important
  - Regular basis
  - Every code submission
- Hardware
  - Equipment Costs
  - Contention for use
  - Downtime for hardware refresh
- Functional Dependencies
  - Test all areas
  - Use all current changes

# Current DevOps

- Develop Code
  - Build
  - Pipeline for Static Analysis
- Inspect/Review
- Test
  - Unit Testing
    - Language based frameworks
  - Integration Testing
  - System Testing



# Current DevOps Cont'd

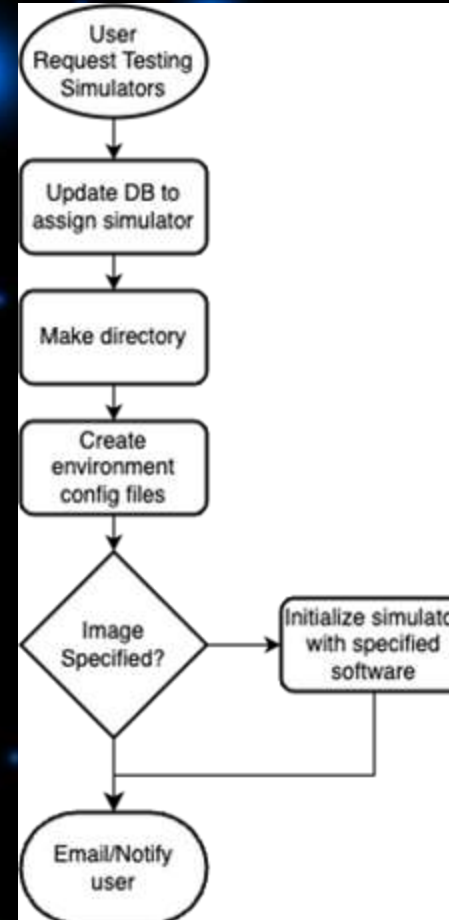
- Architecture(s)
  - x86\_64 or arm64 (Examples)
  - Cross platform builds
  - Parallel building
- Functional Areas
  - Most features touch several areas
- Example
  - REST API
    - Authentication (Every endpoint)
    - Database update
    - Bring service up
    - Responses

# Software for Hardware

- Connecting
  - rconsole
- Loading Image onto the Hardware
- Timing
  - Initialization Time
  - Asynchronous Commands
- Capturing output
- Errors
  - Console only errors
- Hardware inside pipeline

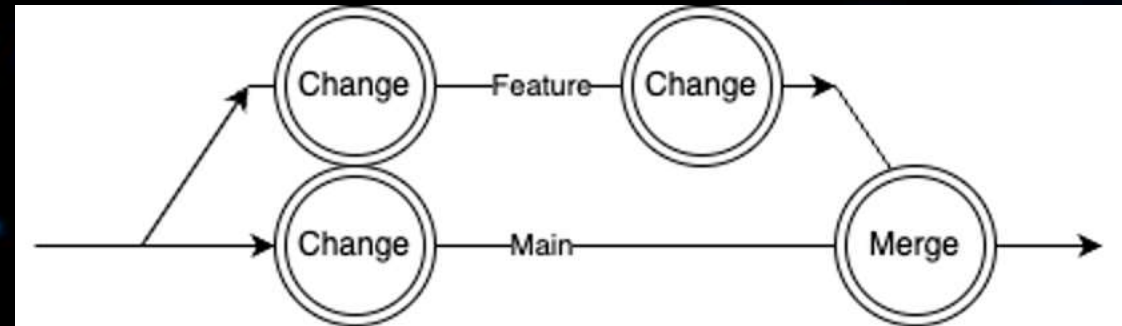
# Emulation

- Cost savings
  - Hardware
  - Time savings
  - Team sharing
- Scale
- Sample Testbed Creation
- Templates
  - Benefit multiple teams
  - Expand to test suites
  - Changes benefit all suites
  - Test in parallel threads



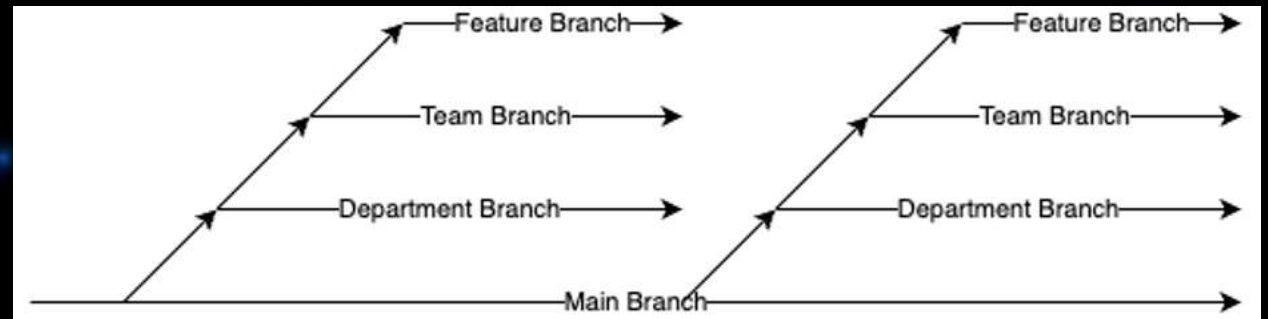
# Code Branches

- Single Main
  - Simple
  - Easy of use
  - Most small projects
  - Backing out changes
  - Merges
  - Testing on each submission
  - Sync from main
- Requires tagged version branches
  - Long term support



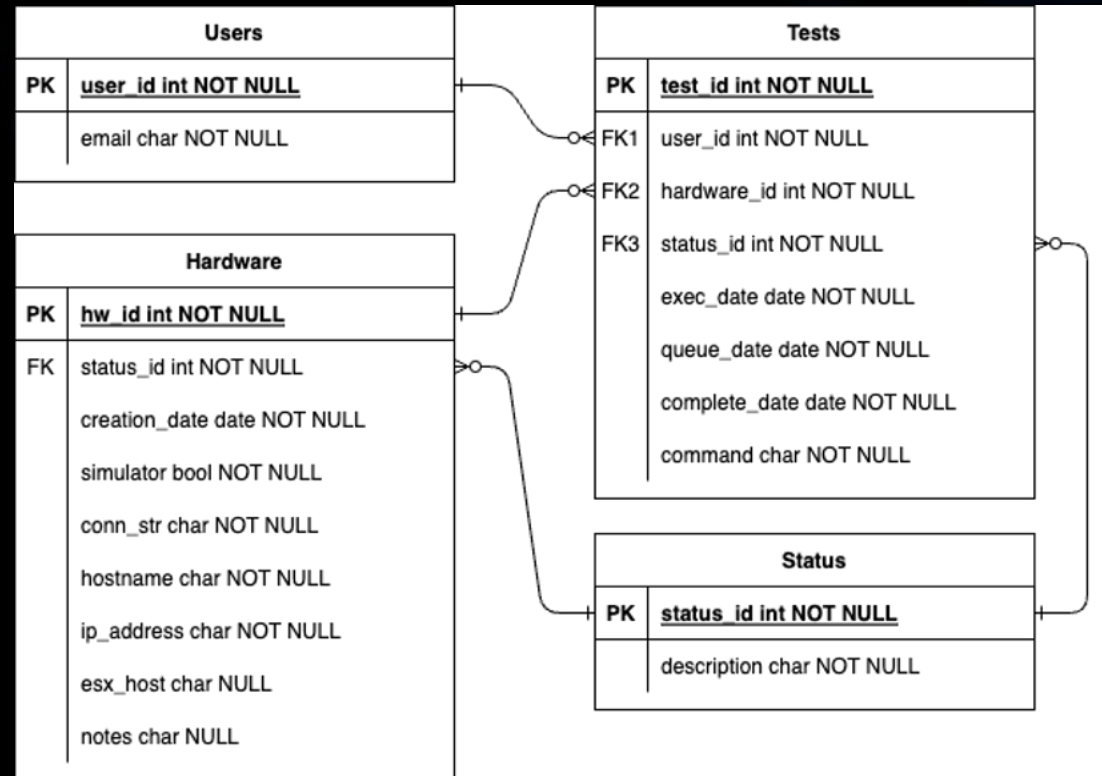
# Code Branches Cont'd

- Complex branching
  - More teams more problems
  - Slow to merge up
  - Cherry picking changes
  - Problem changes impact
  - Testing takes longer
- More teams
- Testing every Change
  - Large workload for concurrent changes
  - Sequential Queue



# Code Branches Cont'd

- Single main branch
  - Smaller feature branches
- Testing Periodically
  - Time based cron
  - Find bad changes via sorting algorithms
  - Bulk good changes validated quicker
- Handle large amounts of changes
- Better picture of quality
- Use Emulators for scaling testing
- Sample DB Schema



# Drawbacks

- Delays
  - Intermittent failures
  - Infrastructure failures
  - Multiple bad changes
    - Prevent any changes during the find phase
- Don't wait to submit before vacation
- Custom Infrastructure
  - Team to support
  - Time investment
- Time to be approved
  - Can be shortened as Developers learn the system

# Wrap up

- Write tests for areas you care about
- Reduce costs
  - Emulators
  - Test more frequently
- Prevent bad changes quicker

# Questions?

???

---

**Brent Clausner**

Software Continuous Integration with Hardware

[Distribution Statement A] Approved for public release and unlimited distribution.

# References

- Tim Menzies, William Nichols, Forrest Shull, & Lucas Layman (Retrieved on 2023, May 29) Are delayed issues harder to resolve? Revisiting cost-to-fix of defects throughout the lifecycle. Retrieved from: <https://link.springer.com/article/10.1007/s10664-016-9469-x>
- David Davis (Retrieved on 2023, May 30) What is VMware ESXi Server and Why do I Need It? Retrieved from: <https://www.pluralsight.com/blog/it-ops/what-is-vmware-esx-server-and-why-you-need-it>
- Install GitLab Runner (Retrieved on 2023, May 30) Retrieved from: <https://docs.gitlab.com/runner/install/index.html>
- Stable Diffusion web UI: <https://github.com/AUTOMATIC1111/stable-diffusion-webui>

# PNSQC

OCTOBER 9-11 2023

AMP IT UP:

TRANSFORMING QUALITY

THANK YOU

Brent Clausner

Software Continuous Integration  
with Hardware

