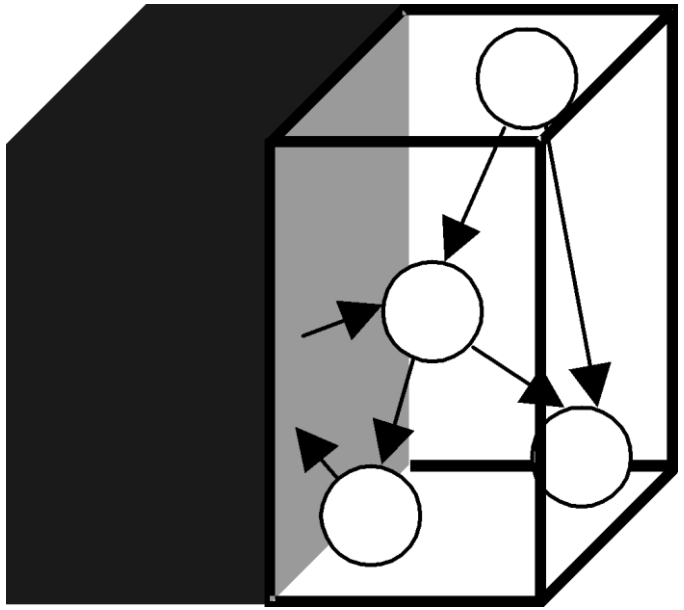
SHIFT TESTING LEFT AND DEFECTS DOWN WITH CICD

GIRTS BALTAISBRENCIS,
TAPOST2019

INTRODUCTION

WHAT ARE THE TESTING OBJECTIVES

TESTING



- Test as early as possible has always been deemed to be most efficient way of finding defects
- Different levels and layers, different test types and coverages by different project delivery players
  - Programmers
  - Manual test engineers
  - Test automation
  - Business and system analysts
  - Subject matter experts
  - Real system users or end users

EVOLUTION

HOW DID WE GET THERE

PROGRAMMER

- Setting up own workstation
- Installing famous tools famous versions
- Coding application/solution in lovely IDE
- Deploying application manually from command line or IDE with own made super duper scripts and self chosen tools
- Verifying / testing application with own made tests or test applications
- Non defects or everything works on my computer

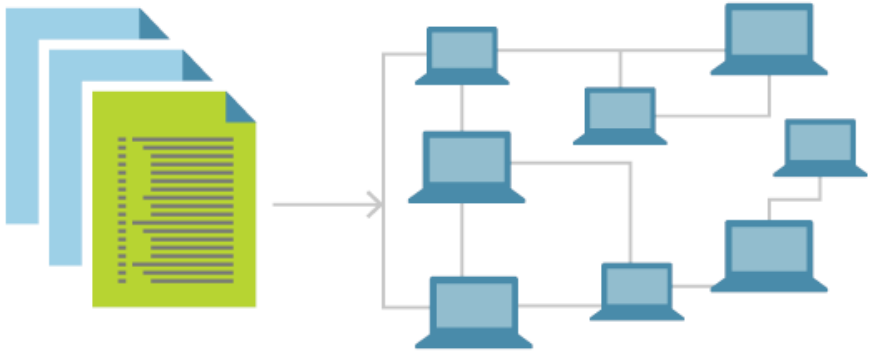
TEST ENGINEER

- Setting up own workstation
- Installing famous tools famous versions
- Coding tests in lovely IDE using different set of test tooling
- Using manually deployed test servers by either developers, testers or admins from command line or early generations of CI using different tools, scripts in different scripting languages
- Verifying / testing with own made test scripts or scripts developed for different test tools
- More realistic defects, but still many non defects

EARLY CI/CD

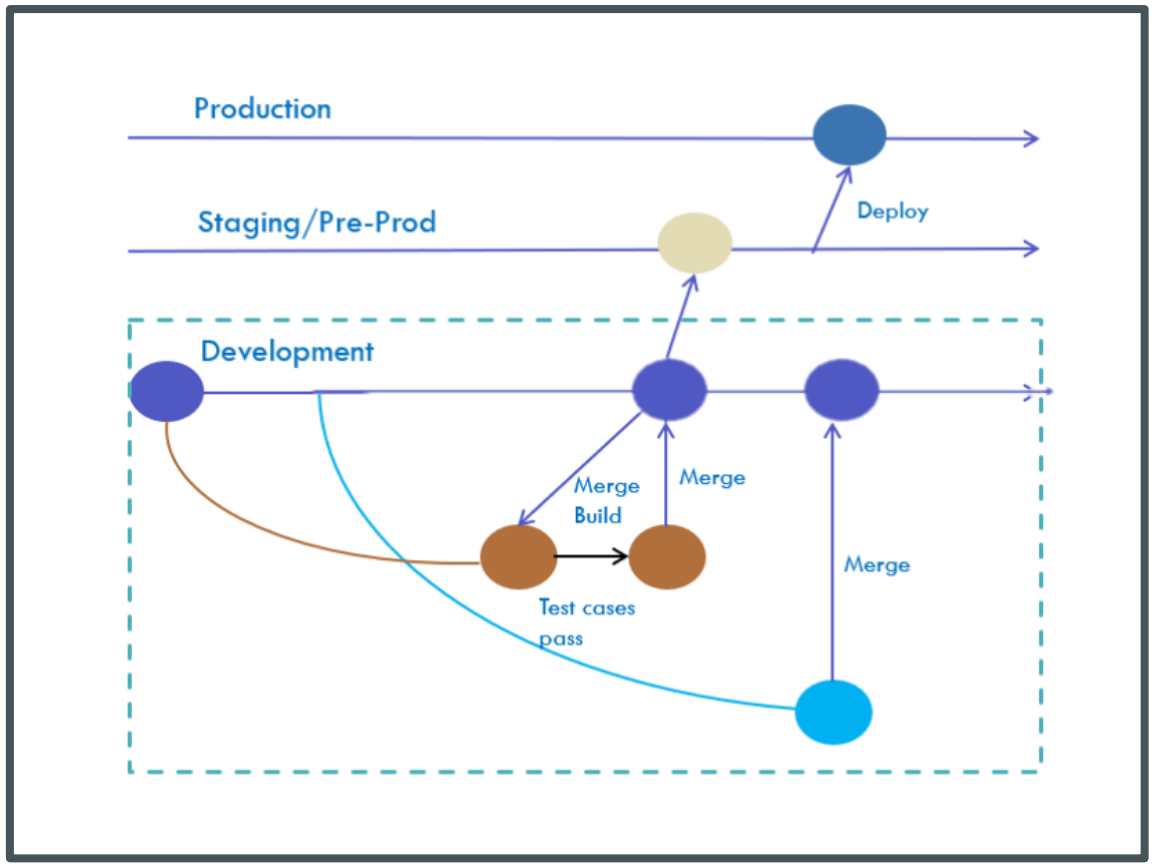
- Most of the systems in the test environment are deployed from templates or cloned from existing systems fulfilling the same role as a new systems
- Test and development workstations are still mostly randomly configured, though not playing big part in testing processes within CI/CD
- Applications and systems under test are built and deployed using CI/CD processes and pipelines
  - Build/Unit test/Integration test as part of build process
  - Smoke and limited regression test after each deployment
  - Early versions of CI/CD processes preventing code merges up stream if any of above fails
- Comprehensive regression test execution with nightly CI/CD processes

MODERN CI/CD



- Any system of any kind in any kind of environment is built by scripts taken from source management system using templates residing in containers and credentials stored in vault
- Test and development workstations still might be manually updated and configured by respective team members, but that has almost zero influence on CI/CD and testing processes. Only disadvantages to respective users when testing and debugging own code

MODERN CI/CD

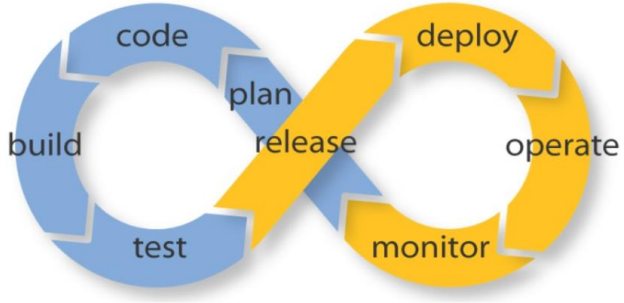


- Testing in CI/CD is triggered as early as developer initiates code commit, it runs on systems which are controlled by DevOps team.
  - Broken or failing code will not go anywhere up stream until it will be fixed, or tests updated according to changed requirements and system behavior
- Private data center based CI/CD and testing processes usually are cloud agnostic, thus can support multiple public cloud providers like AWS, Azure, GCP, Oracle and others
  - For economic efficiency CI/CD and testing processes should become cloud native and capable to utilize services there

BENEFITS

ABILITY TO RUN ALL KINDS OF TESTS EARLY IN THE DEVELOPMENT LIFE CYCLE IN HOMOGENOUS ENVIRONMENTS

WHY DOES CI/CD PROVIDE BENEFITS TO EARLY TEST EXECUTION



- Equal configurations in all environments
  - Dev/Test/UAT/Perf/PrePROD/PROD
- Utilize cloud native services for cost efficiency
- Execute as many tests as early as possible
  - Don't' let broken code up stream
- Use Mocks and stubs with as close as possible interface and 3rd party integrated system behavior
  - Same deployment tools and scripts with configuration parameters stored outside
- Same deployment tools and scripts for PrePROD and PROD with additional security measures preventing test configuration parameters harm production

RESULTS

WHERE ARE OUR DEFECTS WITH TESTS SHIFTED FAR LEFT?

DEFECTS ARE GONE ☺

- Testing shifted as far left as possible
- Executed on homogenous environments
- Equally built and managed with CI/CD for all type of environments
- “works on my computer” likewise in production
- No effort spent on non defect analysis and troubleshooting
- Less effort spent on environment management
- Many reusable pieces for new product buildouts in scope in many projects

FUTURE

ML AND AI

MACHINE LEARNING

- Building of knowledge base for Machine Learning purposes
  - Source code annotation
  - Test annotation
  - Coverage data
  - Defected areas
  - Business priority
  - System seventy
- Design data models

ARTIFICIAL INTELLIGENCE

- Interpretation of designed data model to decide on mandatory tests to be run based on gathered data for ML
  - Annotation mapping
  - Historical test coverages
  - Historical defected areas
  - Business function severity
  - System health severity
  - Additional categories you all may come up with

NEW AVENUES AND NEW ROLES

- Design and develop data models for Machine Learning
- Contribute on development processes and activities on capabilities to gather data intended for AI decision making during whole development life cycle
- Based on test execution results, review and adjust ML models and supply additional algorithms to AI for more efficient and early testing abilities
- ML and AI will not be able to live and survive without humans adjusting them

QUESTIONS

GIRTS BALTAISBRENCIS

PRESIDENT, LSTQB

VICE PRESIDENT, EIS GROUP, CI/CD PLATFORM SOLUTIONS