CEC Continuing Education Center

DesignNews

Introduction to Build Systems and CMake

DAY 5 : Adopting Modern Practices

Sponsored by

DigiKey

BENINGO EMBEDDED GROUP

informa markets

Webinar Logistics

- Turn on your system sound to hear the streaming presentation.
- If you have technical problems, click “Help” or submit a question asking for assistance.
- Participate in ‘Group Chat’ by maximizing the chat widget in your dock.

01

Review:
The Problem

The Problem

There are several problems that teams are facing:

- Managing multiple build configurations
- Slow builds
- Software quality issues
- Inability to use modern techniques like DevOps, Simulation, TDD, etc, effectively
- Productivity issues (time to market, product quality)

The Solution

A carefully designed CMake build system will:

- Simplify build configurations with better dependency management
- Allow for faster, cross-platform builds
- Enable consistency across different development environments
- Unlock modern development processes and tools like DevOps, Simulation, and TDD
- Increase productivity

THE SPEAKER

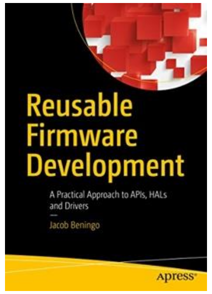
Jacob Beningo

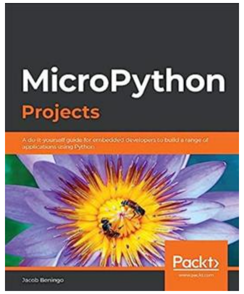
Jacob@beningo.com

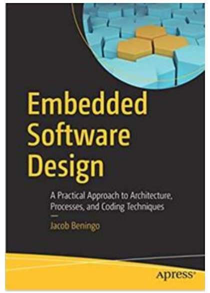
Beningo Embedded Group – CEO / Founder

Focus: Embedded Software Consulting and Training

Help teams deliver higher-quality embedded software faster. We specialize in creating and promoting embedded software excellence in businesses around the world.


Reusable Firmware Development
A Practical Approach to APIs, HALs and Drivers
Jacob Beningo
Apress


MicroPython Projects
A do-it-yourself guide for embedded developers to build a range of applications using Python
Jacob Beningo
Packt


Embedded Software Design
A Practical Approach to Architecture, Processes, and Coding Techniques
Jacob Beningo
Apress

Blogs for:

- DesignNews.com
- Embedded.com
- EmbeddedRelated.com
- MLRelated.com

Visit **www.beningo.com** to learn more

The Plan

Transform Your Build Process: Streamline, Modernize, and Boost Productivity with CMake

Step 1

Learn the Technology

Step 2

Design the Solution

Step 3

Adopt Modern Practices

02

Modern Practices

Why Modern Practices?

Quality

- Buggy software
- Constant bug fixes
- Customer complaints

Development Costs

- Smaller budgets
- More features
- Increased complexity

Time to Market

- More debugging
- Missed deadlines
- Integration woes

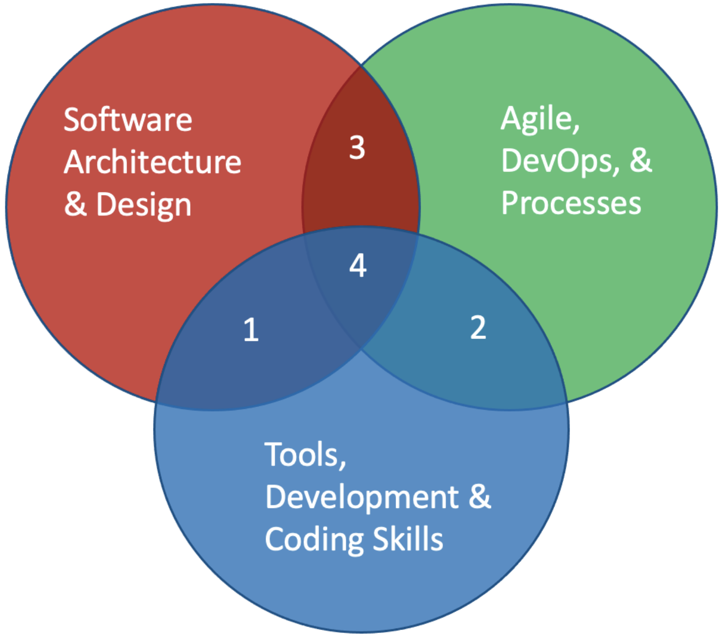
Scalable Solutions

- Tightly coupled code
- Vendor dependency
- Inflexible architecture

Challenges and Solutions

- Late Deliveries -> DevOps
- Software Quality -> Test-Driven Development
- Unavailable Hardware -> Simulation
- Poor Customer Feedback -> Observability
- Multiple Target Support -> Modern Build System

The Embedded Software Triad


Software Architecture & Design
Agile, DevOps, & Processes
Tools, Development & Coding Skills
3
4
1
2

1 - Late, Inconsistent, Quality Issues
2 - Late, Rework, Lost / Meandering
3 - Never completed
4 - Successful Delivery

Audience POLL Question

Which Triad area do you think you fall into?

- a) 1 – Late, Inconsistent, Quality Issues
- B) 2 – Late, Rework, Lost / Meandering
- C) 3 – Never Completed
- D) Successful, on-time delivery

03

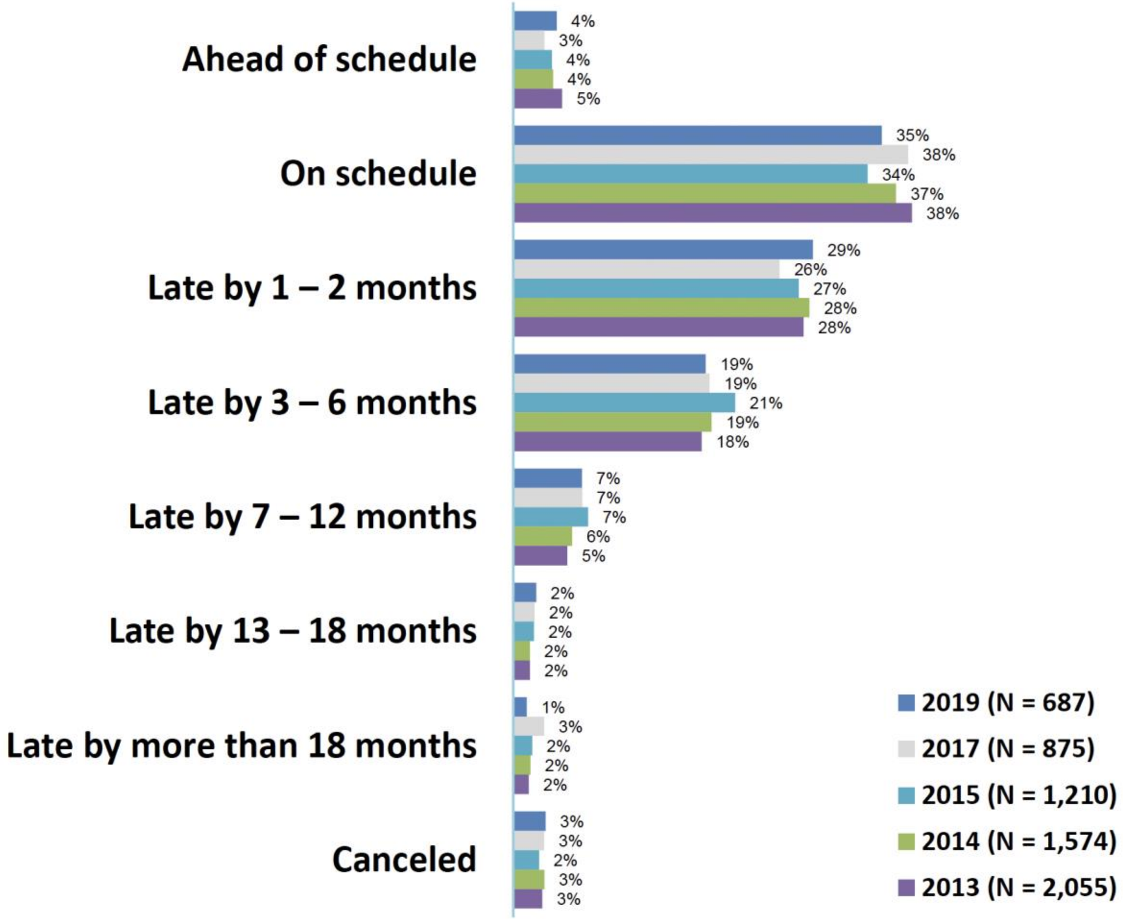
Embedded DevOps

Introduction to Dev(Sec)OPS

Embedded Projects are delivered on-time ~35% of the time.

DevOps is all about improving:
- Efficiency
- Speed
- Quality

It's trying to help you deliver higher-quality products faster!

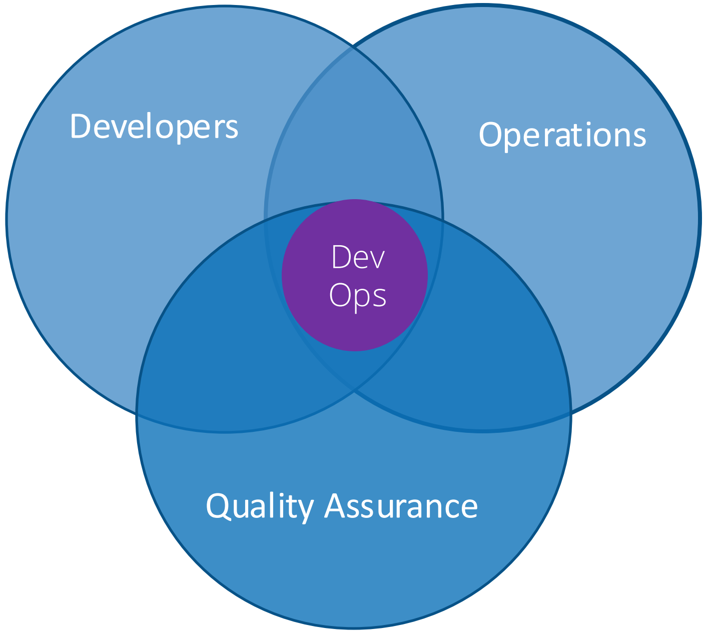


Principles

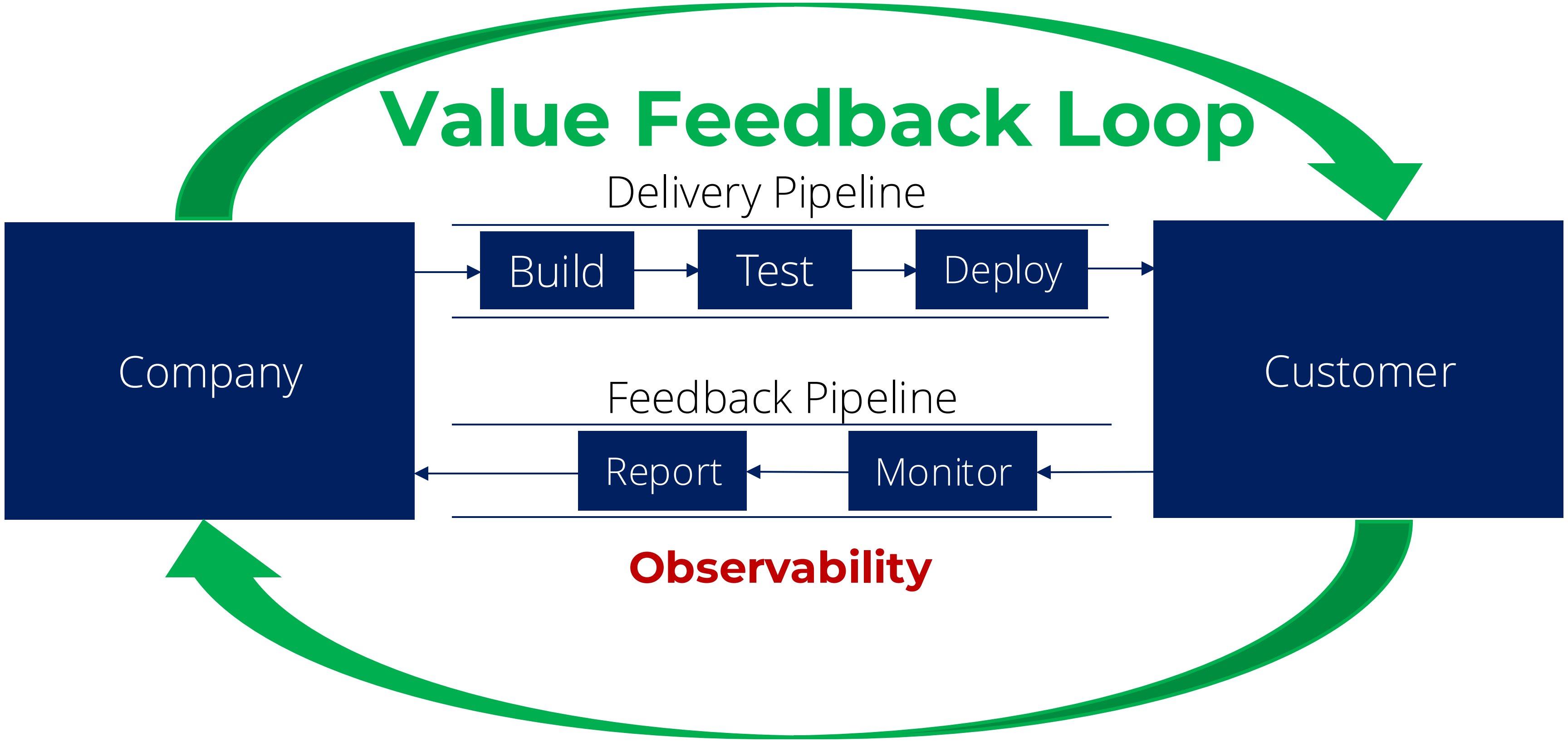
4 Principles guide all DevOps Processes

- Focus on providing incremental value to the users or customers in small and frequent iterations
- Improve collaboration and communication between development and operations teams.
- Automate as much of the software development life cycle as possible
- Continuously improve the software product

DevSecOps is DevOps with an emphasis on integrating security practices



DevOps Principles in Action



Audience POLL Question

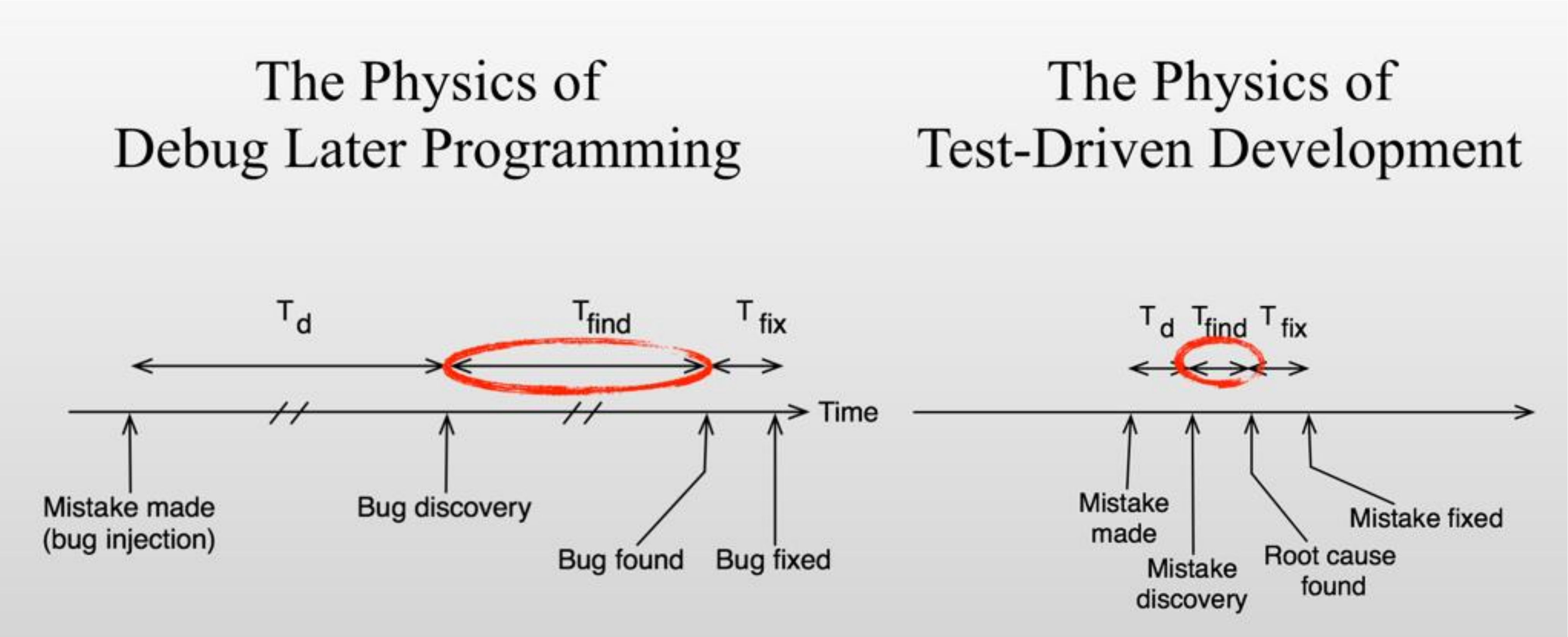
What DevOps features allow you to receive feedback from the customer?
A) Delivery Pipeline
B) Feedback Pipeline
C) CMake
D) None of the Above

04

Test-Driven Development

TDD Physics

Source: James Grenning; TDD for Embedded C; Pragmatic Programmers



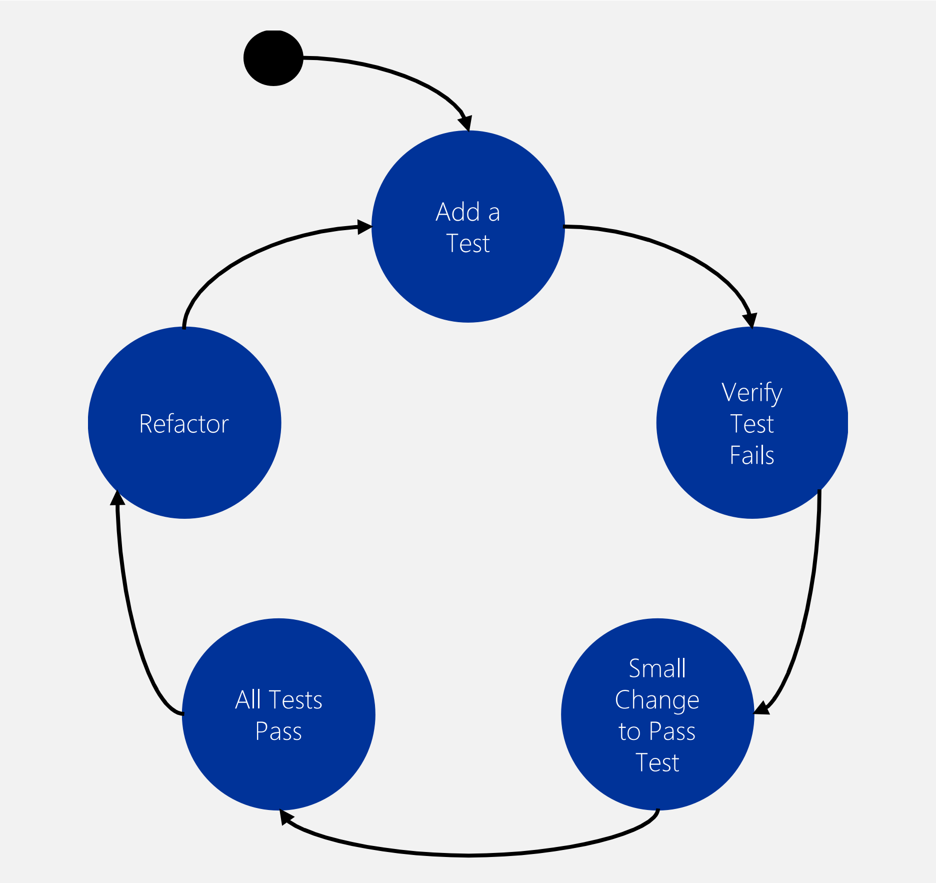
Test-Driven Development (TDD)

TDD is a technique for building software incrementally that allows the test cases to drive the production code development.

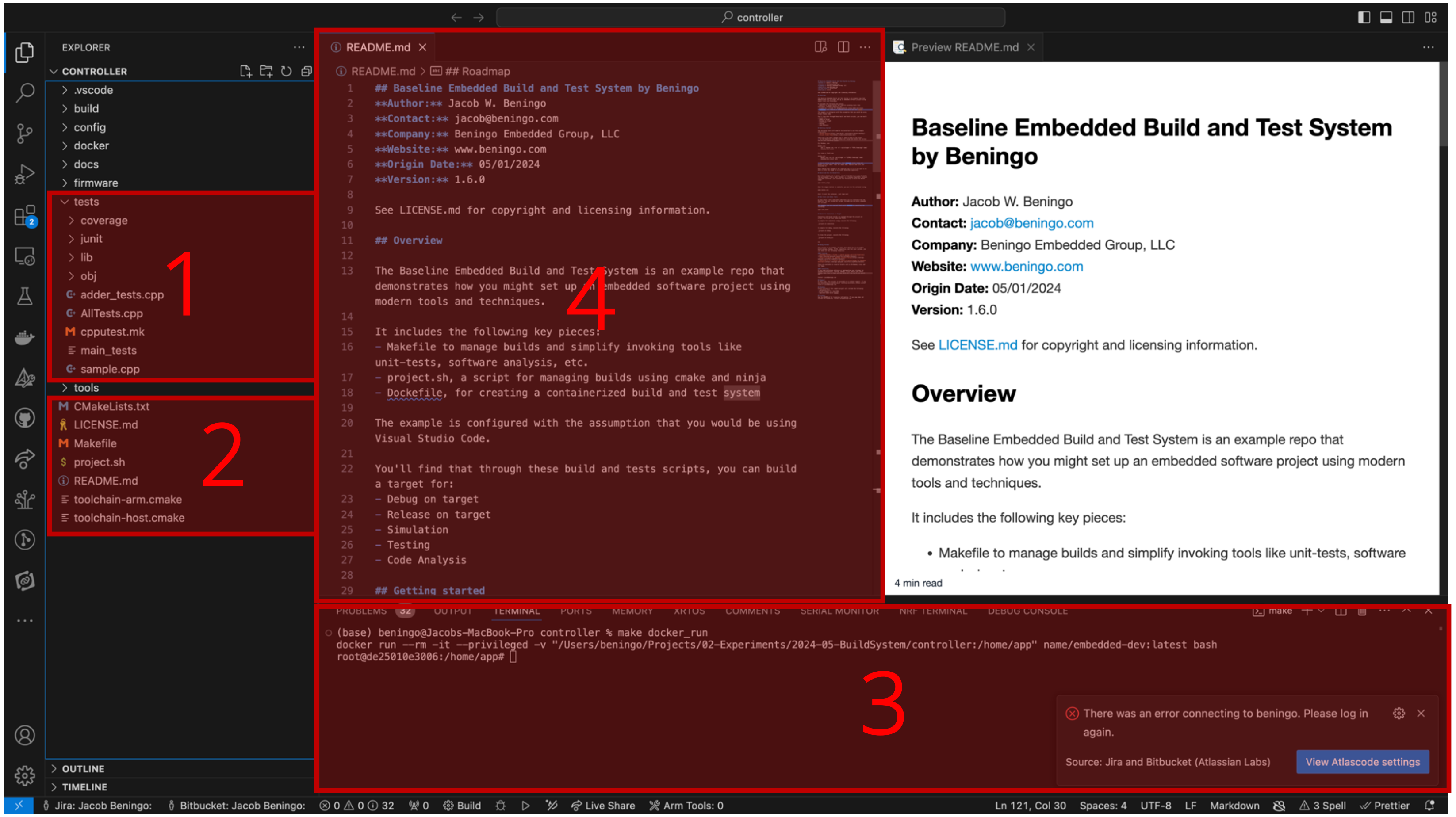
- TDD improves code quality through cleaner, less buggy code
- Improves design due to developers thinking more carefully about what they are doing
- Code is debugged more efficiently due to failing tests that pinpoint exactly what the problem is
- Reduced development time and cost by catching issues earlier in the development cycle
- Developers can refactor with confidence due to existing tests

The TDD Microcycle

1. Add a small test
2. Run all the tests and see the new one fail. (Maybe not even compile!)
3. Make the small change(s) needed to pass the test
4. Run all the tests and see the new one pass
5. Refactor to remove duplication and improve the expressiveness of the tests



Developing your Tests



```markdown
## Baseline Embedded Build and Test System by Beningo
**Author:** Jacob W. Beningo
**Contact:** jacob@beningo.com
**Company:** Beningo Embedded Group, LLC
**Website:** www.beningo.com
**Origin Date:** 05/01/2024
**Version:** 1.6.0

See LICENSE.md for copyright and licensing information.

## Overview

The Baseline Embedded Build and Test System is an example repo that
demonstrates how you might set up an embedded software project using
modern tools and techniques.

It includes the following key pieces:
- Makefile to manage builds and simplify invoking tools like
unit-tests, software analysis, etc.
- project.sh, a script for managing builds using cmake and ninja
- Dockefile, for creating a containerized build and test system

The example is configured with the assumption that you would be using
Visual Studio Code.

You'll find that through these build and tests scripts, you can build
a target for:
- Debug on target
- Release on target
- Simulation
- Testing
- Code Analysis

## Getting started
```

Baseline Embedded Build and Test System by Beningo

Author: Jacob W. Beningo
Contact: jacob@beningo.com
Company: Beningo Embedded Group, LLC
Website: www.beningo.com
Origin Date: 05/01/2024
Version: 1.6.0

See LICENSE.md for copyright and licensing information.

Overview

The Baseline Embedded Build and Test System is an example repo that demonstrates how you might set up an embedded software project using modern tools and techniques.

It includes the following key pieces:

- Makefile to manage builds and simplify invoking tools like unit-tests, software

4 min read

```
(base) beningo@Jacobs-MacBook-Pro controller % make docker_run
docker run --rm -it --privileged -v "/Users/beningo/Projects/02-Experiments/2024-05-BuildSystem/controller:/home/app" name/embedded-dev:latest bash
root@de25010e3006:/home/app# 
```

There was an error connecting to beningo. Please log in again.

Source: Jira and Bitbucket (Atlassian Labs)

Audience POLL Question

How do you feel about TDD?

a) For TDD

b) Cautiously optimistic

c) Skeptical, but see the value in it

d) Rubbish! I can't support this concept

●● Next Steps

05

Embedded Build System

Transform your build system with the free Beningo Embedded Build System example:

- Docker container build system
- Makefile-based
- CMake with Ninja Example
- Compilation scripts
- Integrated tools like cpputest

https://mailchi.mp/beningo/beningo-devops

Additional Resources

Please consider the resources below:

- Jacob's Blogs
- Jacob's CEC courses
- Embedded Software Academy

- Embedded Bytes Newsletter
  - http://bit.ly/1BAHYXm

www.beningo.com

Consulting | Coaching | Training

EMBEDDED SOFTWARE ACADEMY BY BENINGO

Next Steps

- ✓ Introduction to Embedded Build Systems
- ✓ CMake Fundamentals
- ✓ CMake for Embedded Systems
- ✓ Designing your Build System
- ✓ Adopting Modern Practices

CEC Continuing Education Center

DesignNews

Thank You

Sponsored by

DigiKey

BENINGO EMBEDDED GROUP

informa markets