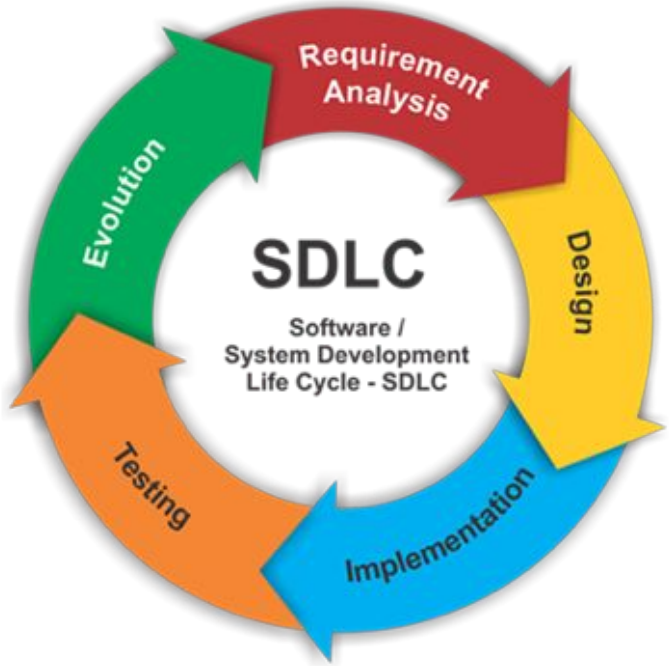
Introducción a DevOps y Metodologías Afines

Ciclos de CI/CD y herramientas

Federico Barceló – Profesor Adjunto
Escuela de Tecnología – Facultad de Ingeniería
UNIVERSIDAD ORT Uruguay
federico@barcelo.com.uy

AGENDA

1. Ciclo de desarrollo old school
2. Continous Integration
3. Continous Delivery
4. Continous Deployment
5. Herramientas


Requirement Analysis
Design
Implementation
Testing
Evolution
SDLC
Software / System Development Life Cycle - SDLC

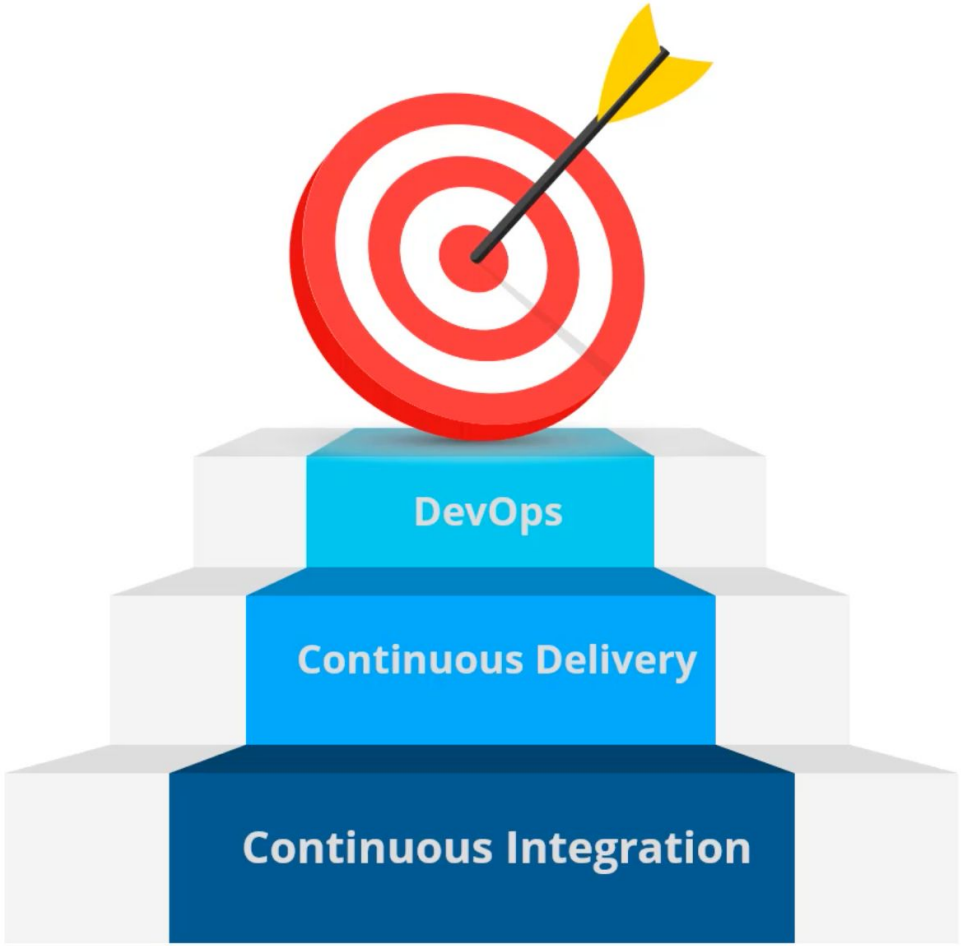
Continuous Integration

Continuous Delivery

Shorter SDLC

Faster Time-To-Market

Business Revenue


DevOps
Continuous Delivery
Continuous Integration

MIKE'S
ELECTRONICS

Public Website

Software Development Team

Product Management


git
Software Development Team
Build & Integration Team
Compile
Package

Operations Team

Test Environment

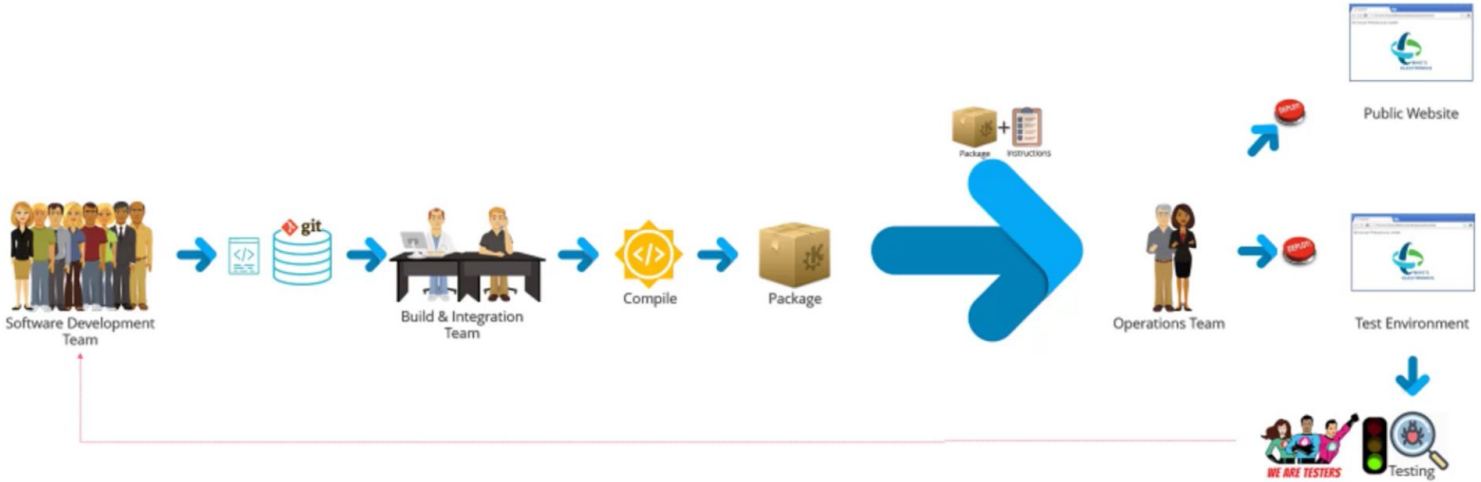
Testing

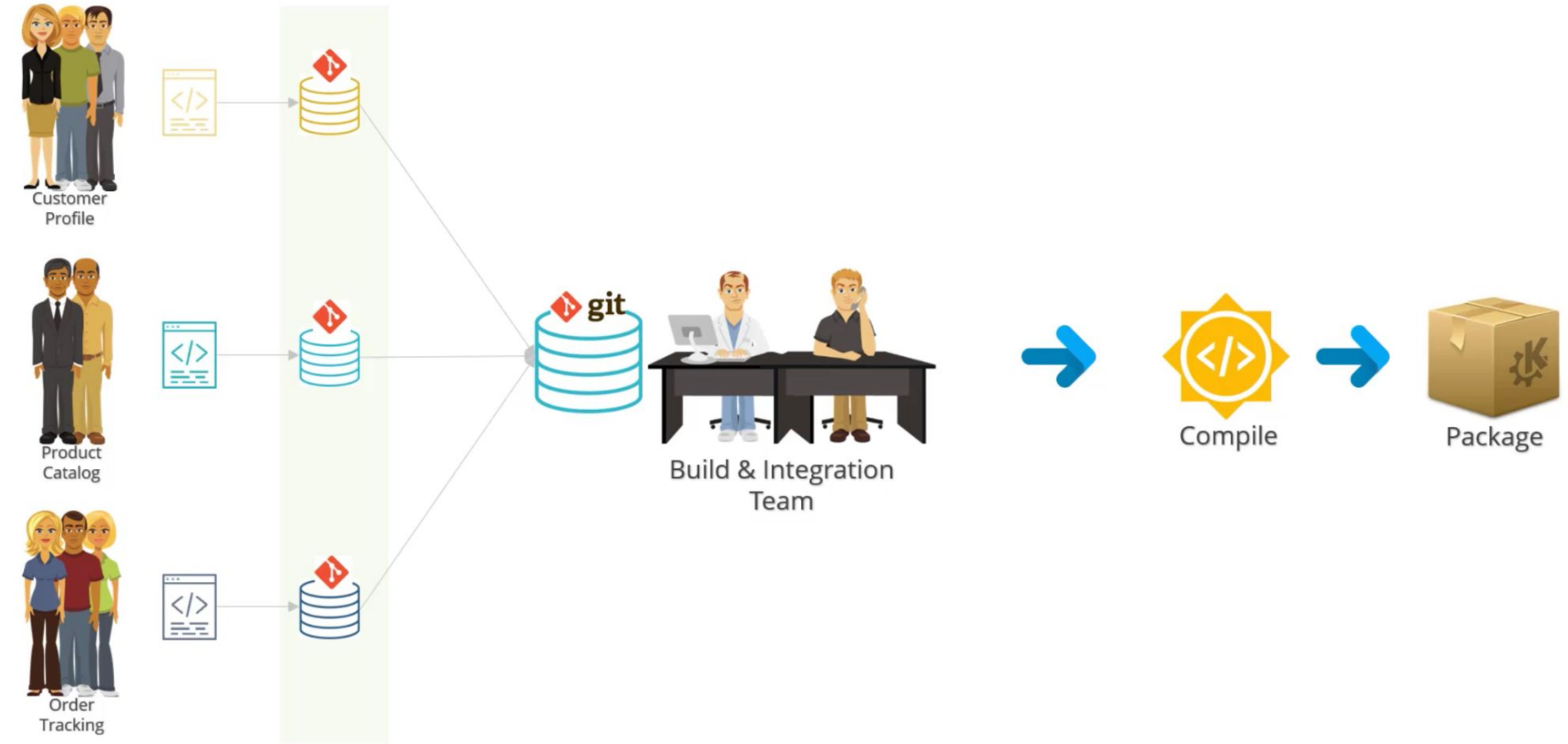
Operations Team

Public Website

Test Environment

2 weeks to 2 months (Iteration)




Customer Profile
Product Catalog
Order Tracking
git
Build & Integration Team
Compile
Package

Old School Code Integration

Pain Points

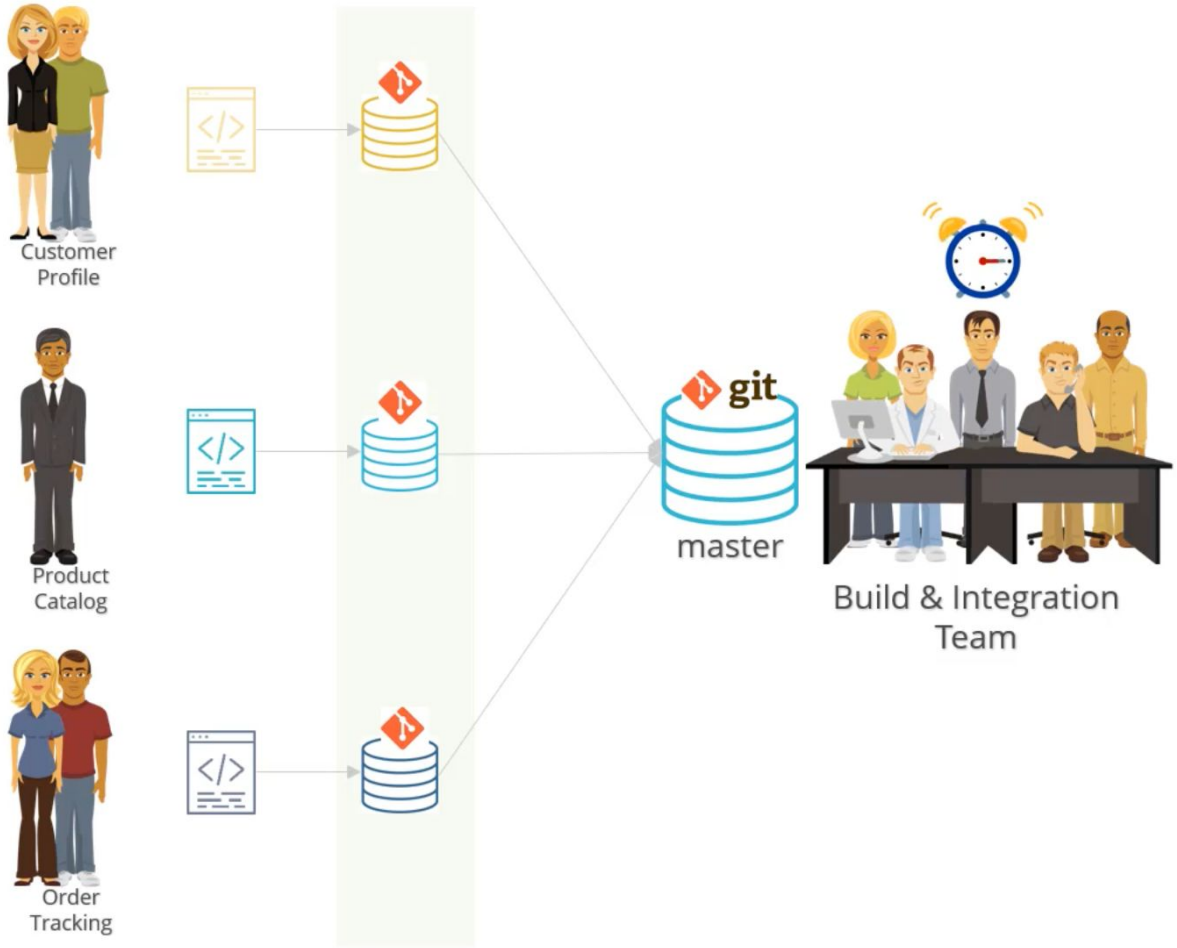
Pain Point #1: Integration is Painful and Effort Consuming

Pain Point #2: Fixing Issues At The End Of Iterations

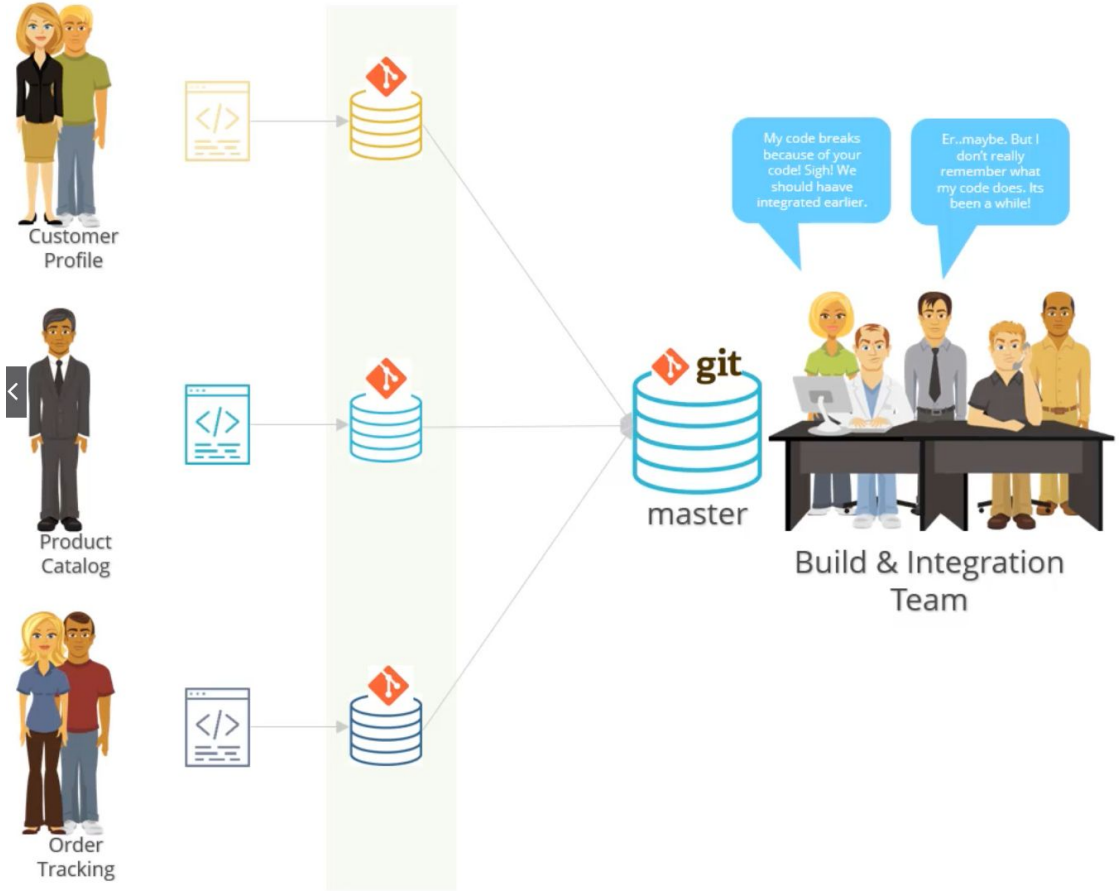
Pain Point #3: Intermediate Merge Issues Can Hold Up Teams

Pain Point #4: Long Feedback Cycle For Functional Defects

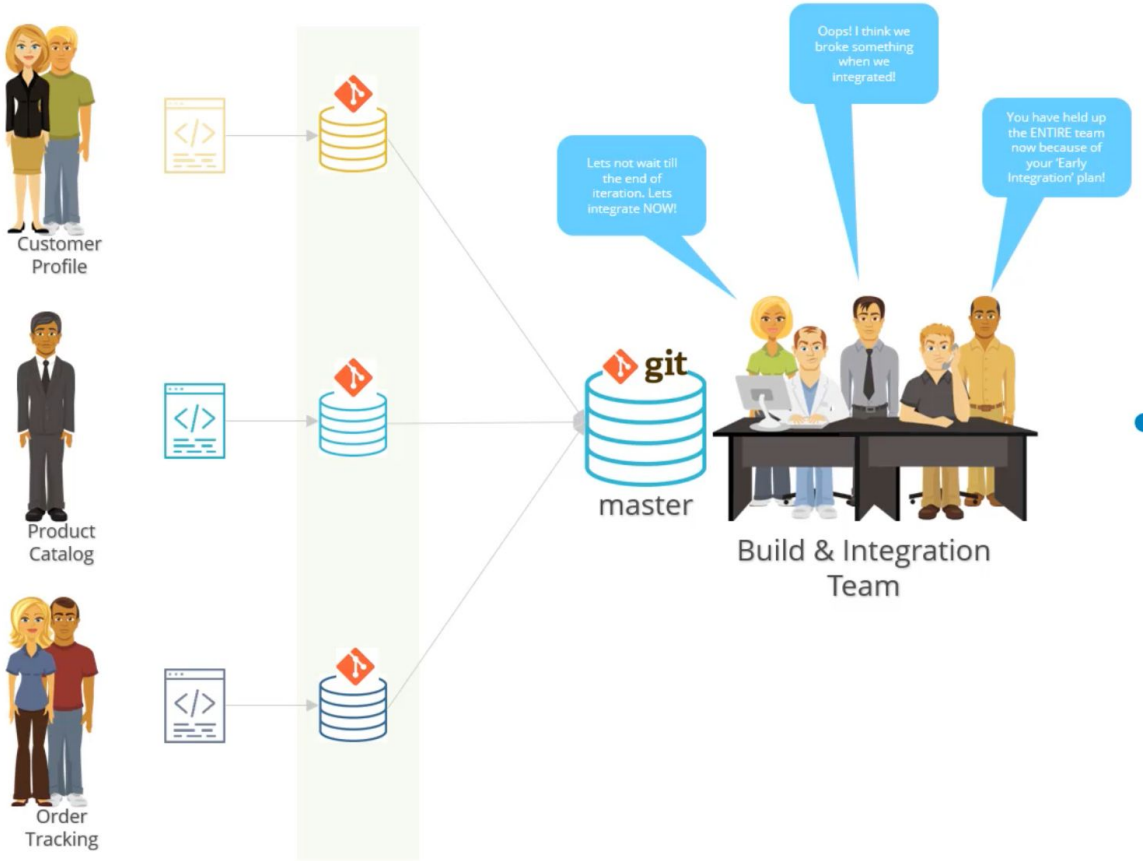
Pain Point #5: Long Iterations



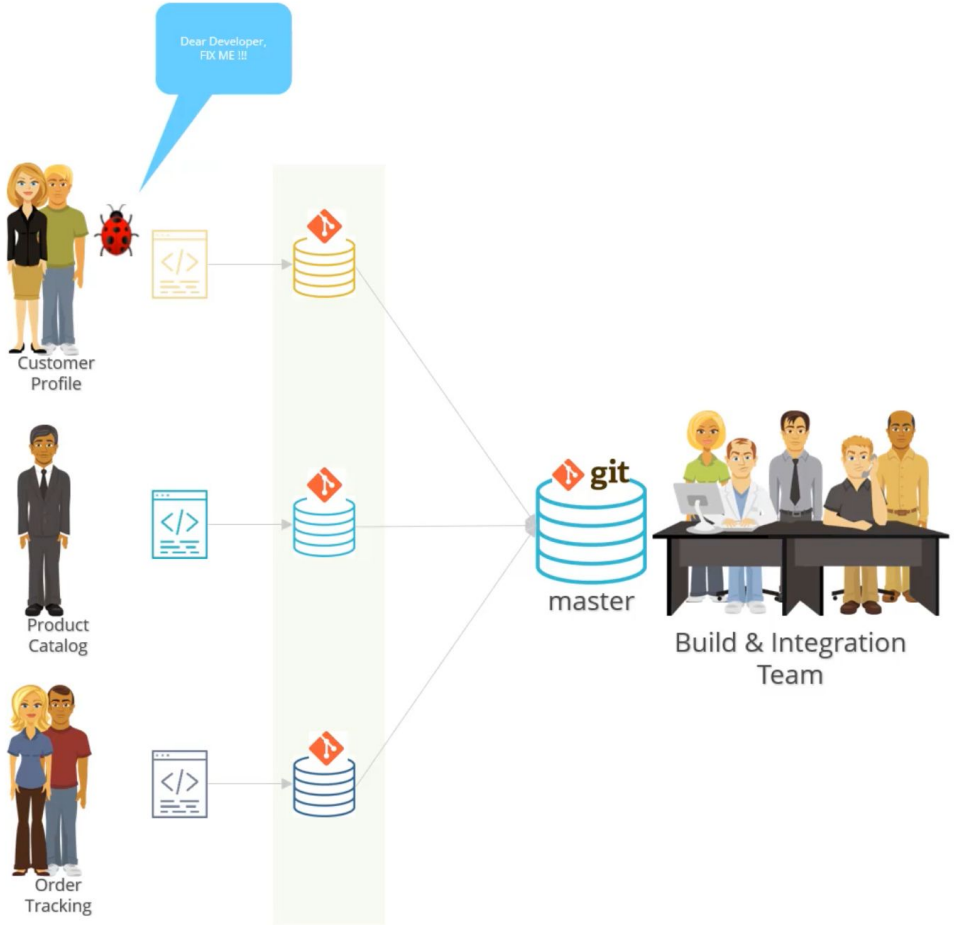
Pain Point #1: Integration is Painful and Effort Consuming



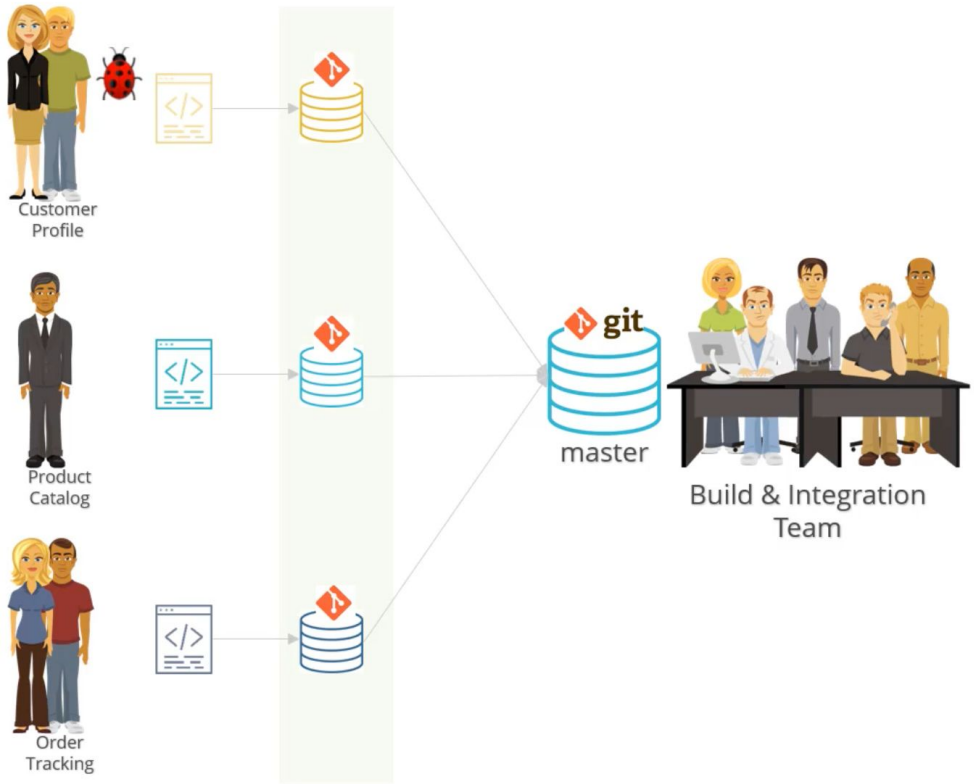
Pain Point #2: Fixing Issues At The End Of Iterations



Pain Point #3: Intermediate Merge Issues Can Hold Up Teams

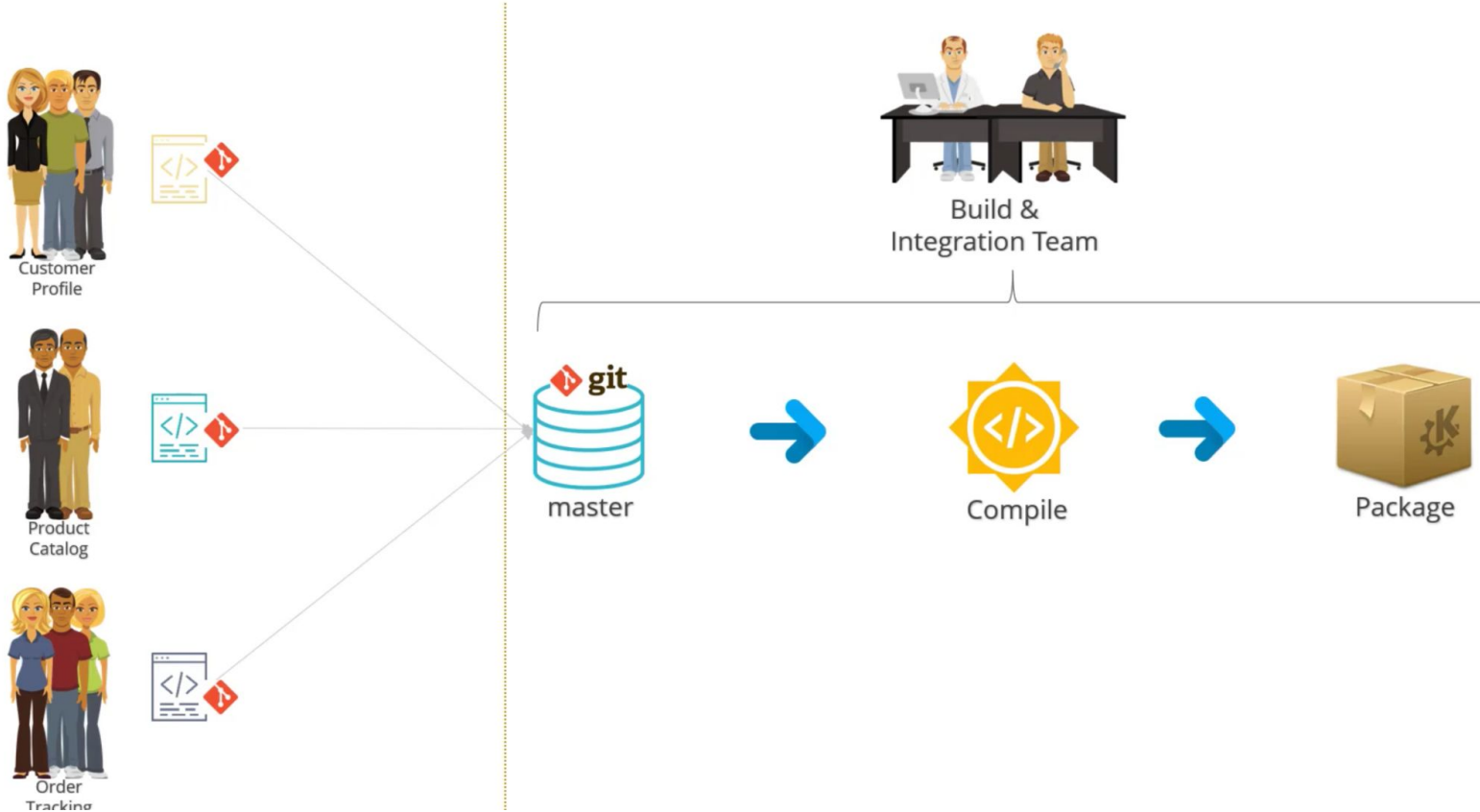


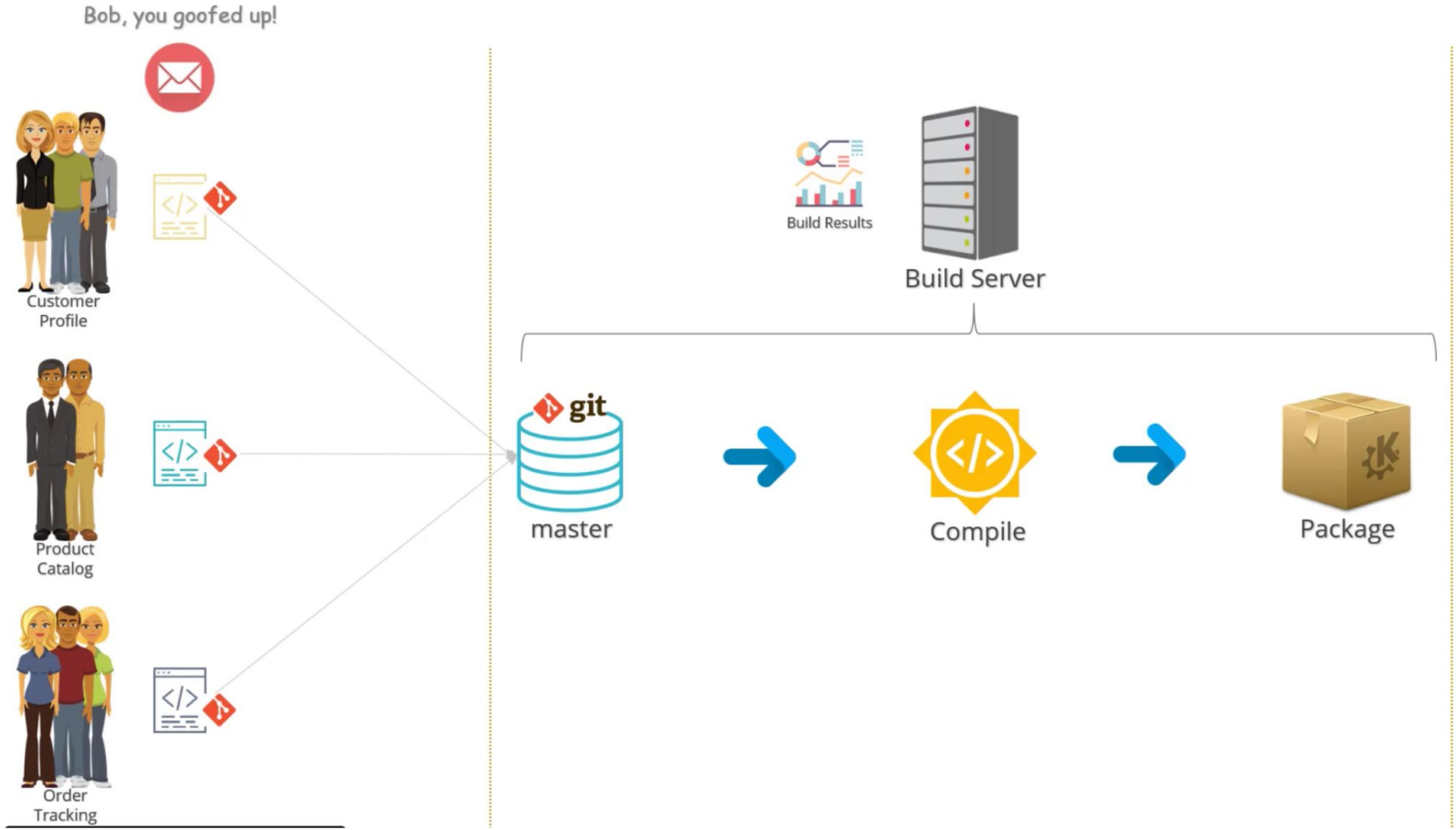
Pain Point #4: Long Feedback Cycle For Functional Defects

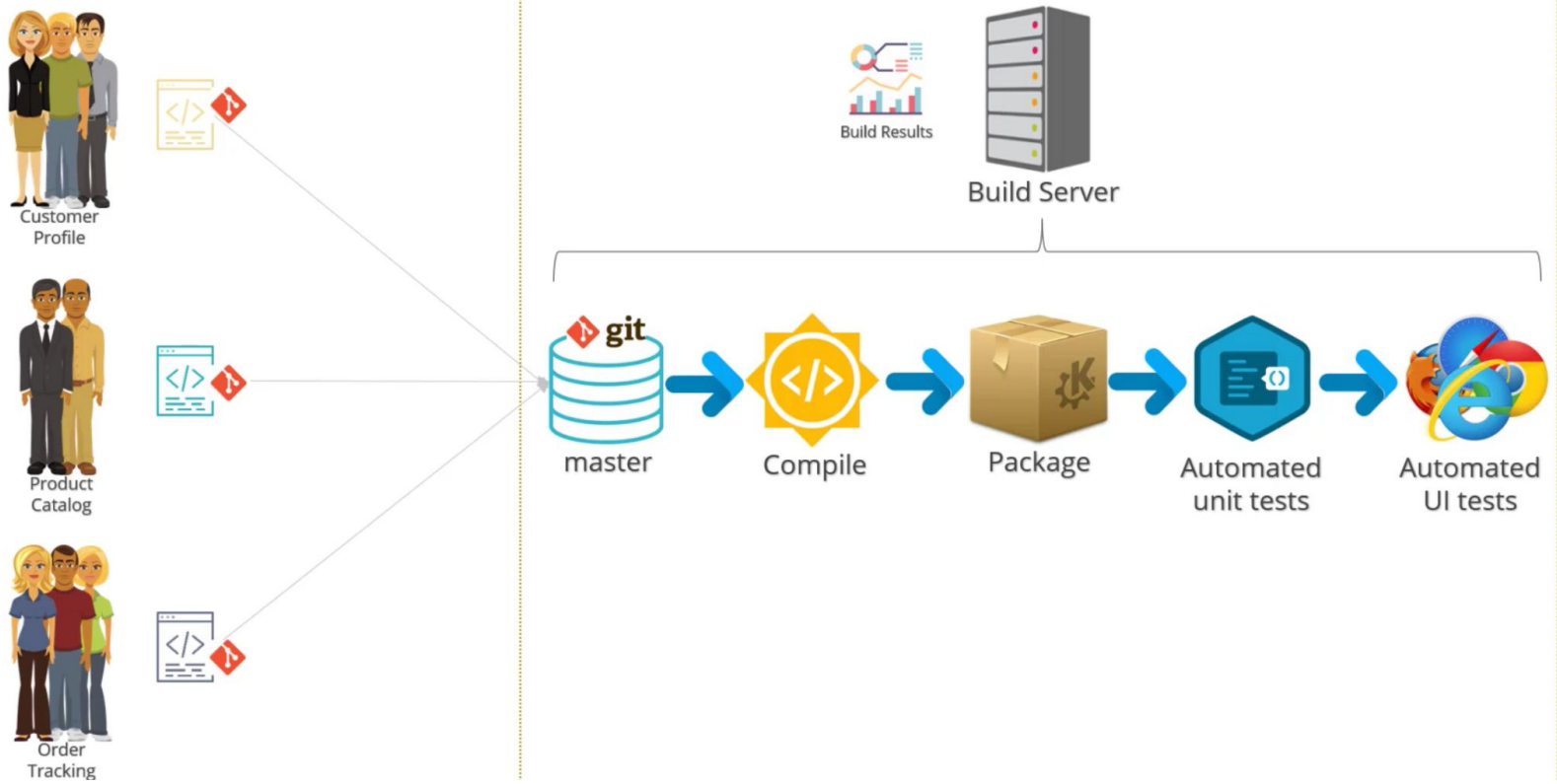


Pain Point #5: Long Iterations

Continous integration


Customer Profile
Product Catalog
Order Tracking
Build & Integration Team
git
master
Compile
Package


Bob, you goofed up!
Customer Profile
Product Catalog
Order Tracking
Build Results
Build Server
git
master
Compile
Package


Customer Profile
Product Catalog
Order Tracking
Build Results
Build Server
git
master
Compile
Package
Automated unit tests
Automated UI tests

Cardinal Principles of Continuous Integration

- A single central repository where the code lives.
- Developers check-in/commit their code frequently.
- Build should be triggered every time a developer checks in code.
- Build should be automated and fast.
- Build should compile the code as well as run automated.
- Fixing a failed build should be top priority for the developers.
- Build results should always be communicated to all developers.

Integration

Old School vs CI

Pain Point #1: Integration is Painful and Effort Consuming.

CI: Integration is automated and quick.

Pain Point #2: Fixing Issues At The End Of Iterations.

CI: Issues show up early , because of frequent integration.

Pain Point #3: Merge Issues Can Hold Up Teams.

CI: Broken builds are fixed with immediate priority by developers.

Pain Point #4: Long Feedback Cycle For Functional Defects.

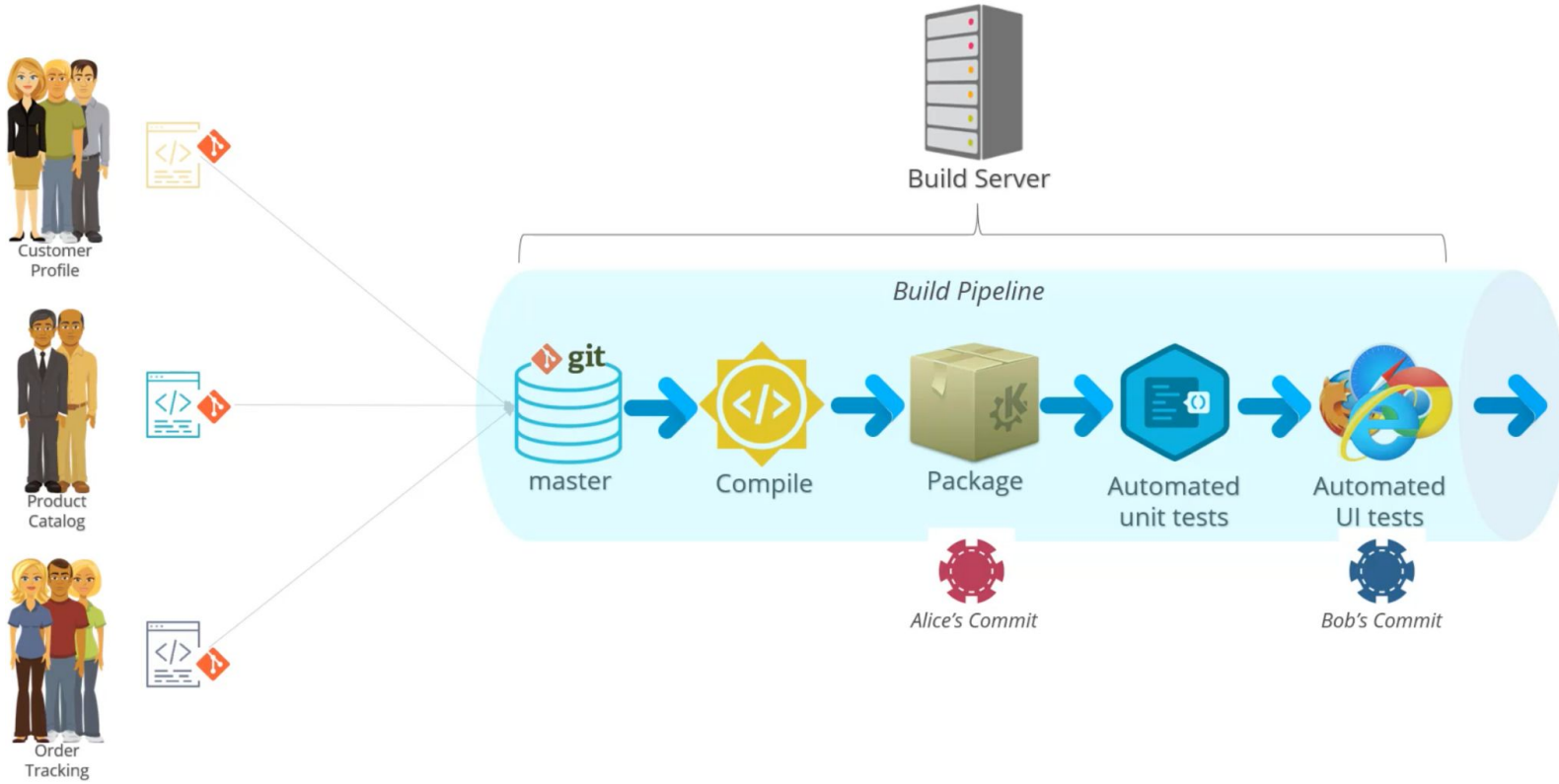
CI: Shorter feedback cycle – Developer is notified immediately.

Pain Point #5: Long Iterations

CI: Shorter Iterations. Faster time-to-market.

Car Assembly Line

Car Assembly Line


Customer Profile
Product Catalog
Order Tracking
Build Server
Build Pipeline
git
master
Compile
Package
Automated unit tests
Automated UI tests
Alice's Commit
Bob's Commit

Operations Team

Instructions for Operations Team

---

1. Copy a set of config files(files ending in .properties) from the root of the package, to the folder /etc/config on the server.

2. Download and install the latest software patches from the website http://xxx/yyy/patches on the Operating System.

3. Setup environment variables on the OS. If you are deploying to the test environment, use the first set of values attached. If you are deploying to production, use the second set of values attached.

Test

---

env=test
db=mockdb
url=test.mikeselectronics.com

Production

---

env=prod
db=productdb
url=mikeselectronics.com

Regards,
The Development Team

Package Instructions

Operations Team

Read Instructions

Prepare Environment

Test Environment

Public Website

Pain Point #1: Correctness of Instructions

Pain Point #2: Difference in instructions across environments

Pain Point #3: Error prone nature of manual tasks

Pain Point #4: Deployments are sophisticated, high-impact with downtime

Continous delivery

Package Instructions

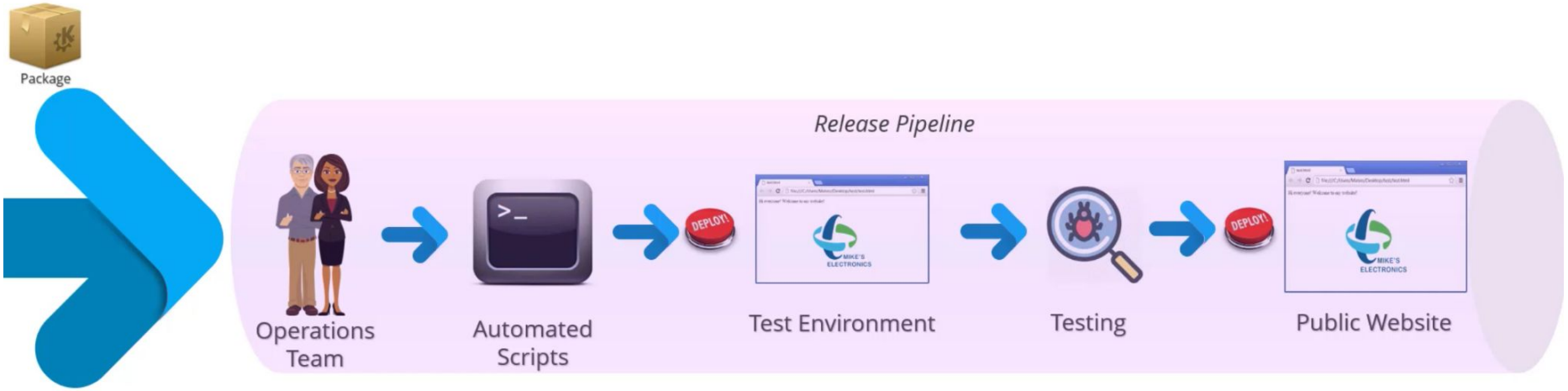
Operations Team

Read Instructions

Prepare Environment

Test Environment

Public Website


Package
Release Pipeline
Operations Team
Automated Scripts
DEPLOY!
Test Environment
Testing
DEPLOY!
Public Website

Old School Operations
vs
Continuous Delivery (CD)

Pain Point #1: Correctness of Instructions.

CD: Correctness of automated scripts can be verified at creation time.

Pain Point #2: Difference in instructions across environments.

CD: Automated scripts can easily pick the tasks for each environment.

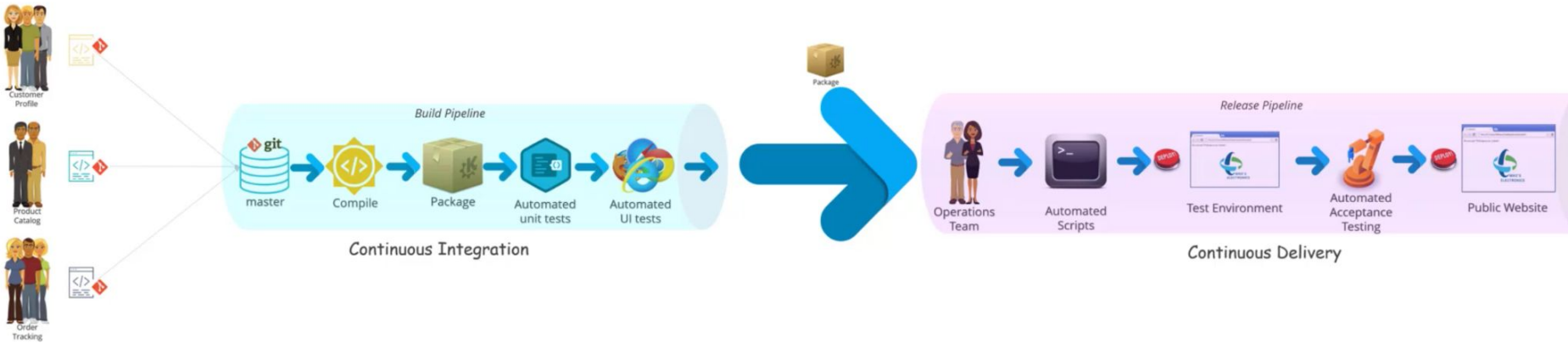
Pain Point #3: Error prone nature of manual tasks.

CD: Automation prevents the occurence of human errors.

Pain Point #4: Deployments are sophisticated, high-impact with downtime.

CD: Automated deployments, easily repeatable, lesser time-to-market.

Continous Deployment


Customer Profile
Product Catalog
Order Tracking
Build Pipeline
git
master
Compile
Package
Automated unit tests
Automated UI tests
Continuous Integration
Package
Release Pipeline
Operations Team
Automated Scripts
Test Environment
Automated Acceptance Testing
Public Website
Continuous Delivery