How we build software at Microsoft

Dinh Doan Van Bien
didoan@microsoft.com
Senior Software Engineer
February 2016

Microsoft

Our organization

- Part of Applications and Services Group (Bing and Office)
- European Development Center
- Office Marketplace EXperiences team (OMEX)
- We work on both services (office.com) and Office clients (Word, Excel, PowerPoint -> Windows, iOS, Android, Web, Mac)

Software Engineers @Microsoft

- Combined engineering, no separate testing/operations organization
- We design/code/test/release/run the software we build
- We manage the delivery project
- We are constantly learning and improving in a data-driven way
- We “obsess” about the customer

Always start with “Why”

- Pro tip: when you are new to a company don’t learn the tools first
- Focus on the “why”
- This will help you understand the “how”

Recent cultural shift

From 3-year ship cycle to high velocity delivery

Agile Manifesto – Principle #1

“Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.”

Our objective and challenge

Reduce the cost, time, and risk of delivering incremental value to users, whilst maintaining quality standards expected.

Do this for huge scale and volume – thousands of requests per second, from all over the world.

Done Means Deployed Live

“Code complete” is not done

Deployed live (or nearly live), even dark, makes us feel more confident

Sense of real, believable progress

Increasing confidence in quality

Prove that a code change works by running it through more and more difficult tests e.g.:

- Low level unit tests
- High level integration tests
- Performance tests

Until we are confident it's shippable and then we measure, measure, measure…

Environment increasingly PROD-like

The code change is also going through environments closer and closer to production

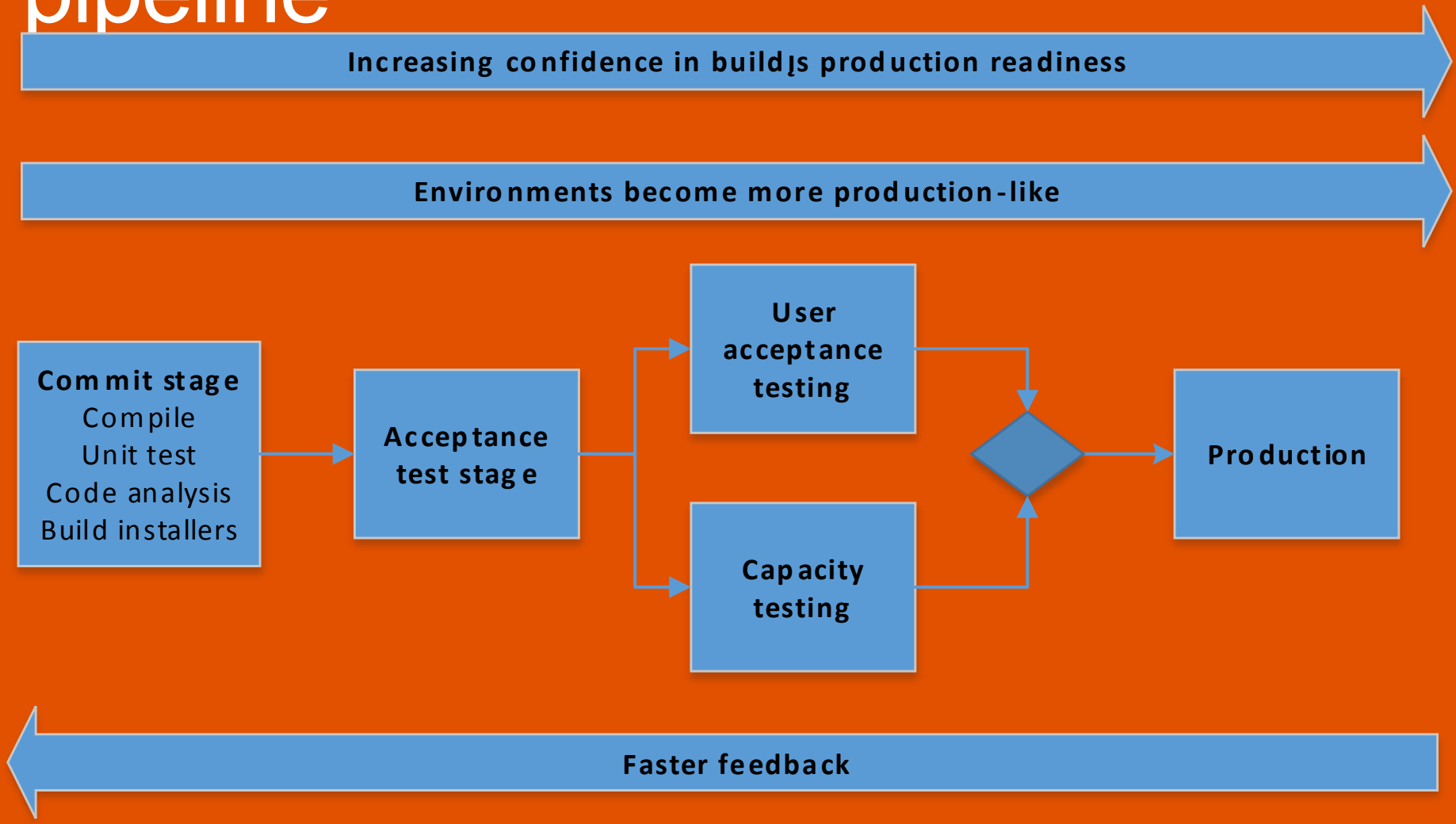
Last stage = exact mirror of production

No Place Like Production

“Production is truly the only place you can validate your code”

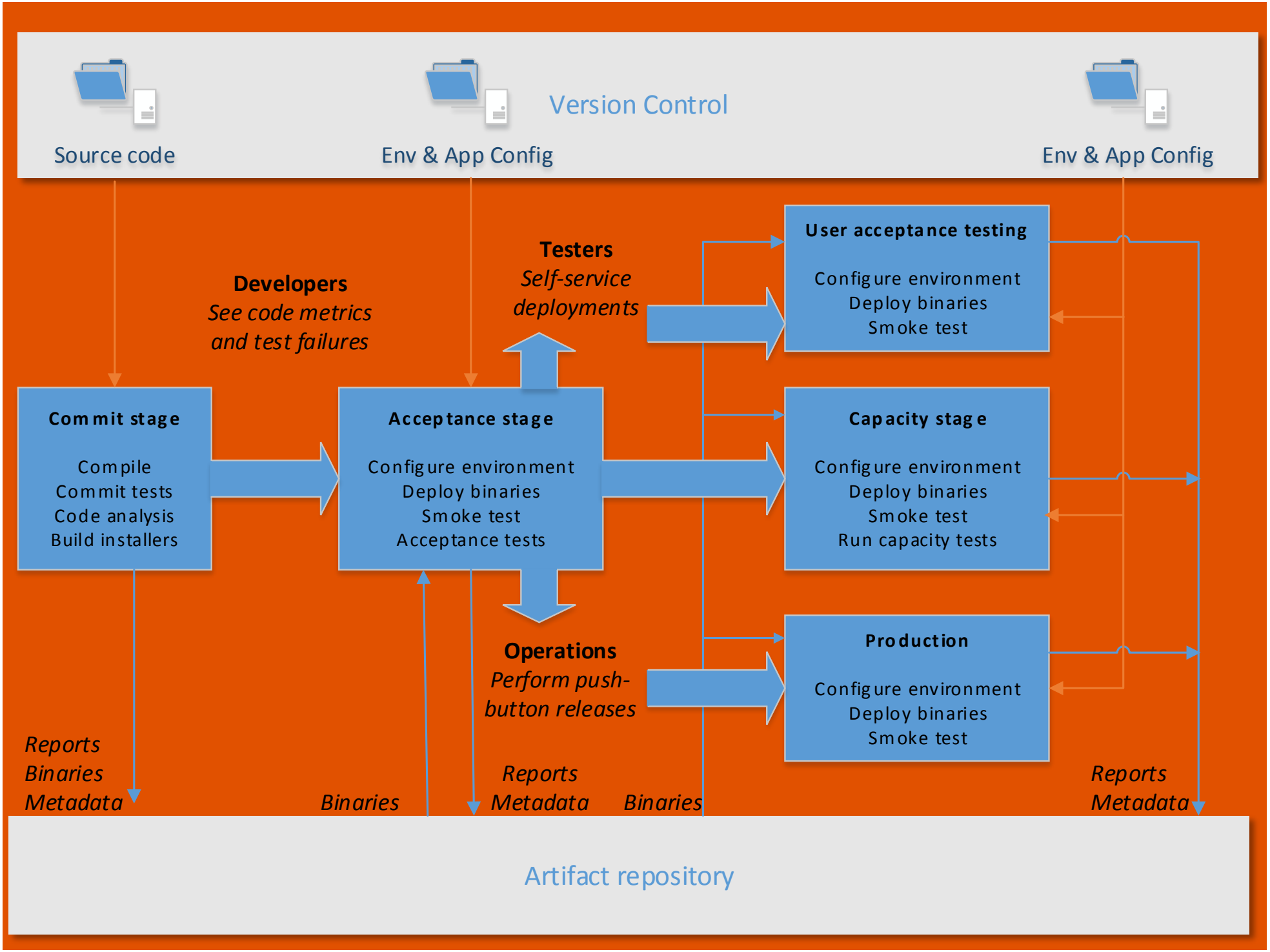
Michael Nygard, author of “Release It!”

Trade-offs in the deployment pipeline



Production vs Rest of the World

More database servers in prod
Bigger database hardware in prod
More web servers
Various replication schemes
Different versions of server and OS software
Schema changes applied at different time
Physical hardware in prod
More data in prod
Legacy data (X years of odd stuff)
More traffic in prod
Faster disks in prod


Version Control
Source code
Env & App Config
Env & App Config
Developers
See code metrics and test failures
Testers
Self-service deployments
User acceptance testing
Configure environment
Deploy binaries
Smoke test
Commit stage
Compile
Commit tests
Code analysis
Build installers
Acceptance stage
Configure environment
Deploy binaries
Smoke test
Acceptance tests
Capacity stage
Configure environment
Deploy binaries
Smoke test
Run capacity tests
Operations
Perform push-button releases
Production
Configure environment
Deploy binaries
Smoke test
Reports
Binaries
Metadata
Binaries
Reports
Metadata
Binaries
Reports
Metadata
Artifact repository

Our Engineering Pillars

Extreme automation

Solid configuration management (i.e. version control)

DevOps culture

Continuous integration

Continuous insights

Automate All the Things!

Build from scratch

Deploy automatically to an arbitrary environment

Infrastructure as code

Automated test all along

"*Automated tests transform fear into boredom*"

Eran Messeri, Google

Version Control All the Things!

Source code
Database schema
Service configuration
Environment data
Dependencies

Change metadata = What? When? Why?

Smaller Changes => Less Risk

By shrinking change size:

- Easier to identify problem
- Easier to roll back

Continuous Integration

Do you commit to trunk/main at least once a day?

Does each check-in trigger a build and a battery of automated tests?

"We don't optimize for being right. We optimize for quickly detecting when we're wrong"

Kellan Elliot-McCrea - Etsy CTO

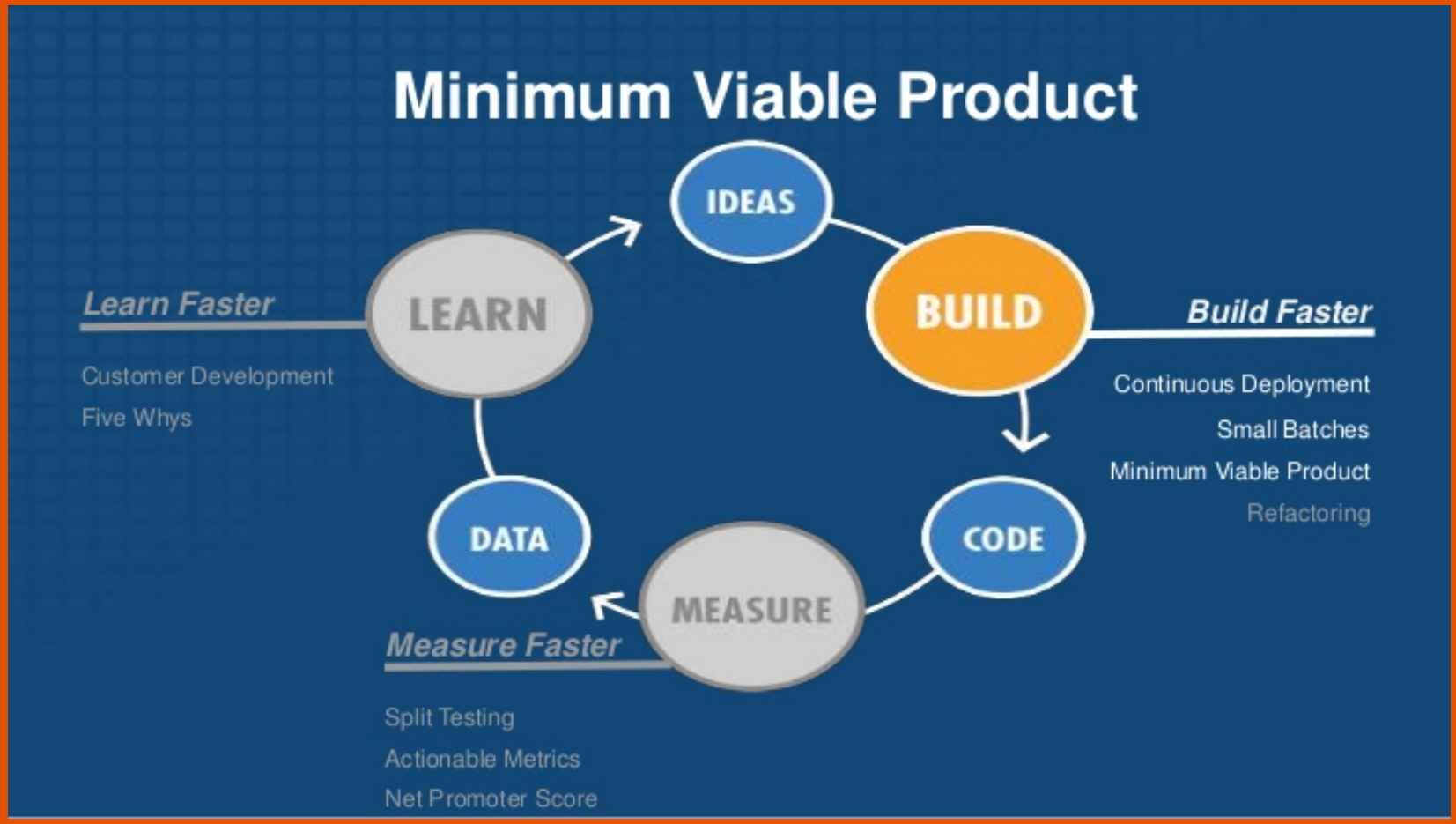
DevOps

Little bit weird
Sits closer to the boss
Thinks too hard

Pulls levers & turns knobs
Easily excited
Yells a lot in emergencies

Learn in Production

Be methodical about it and use Gating + A/B testing to run experiments



User Feedback

Do not listen to what customer SAY they want, find out what they **ACTUALLY DO** through telemetry and analytics

Are We Building the Right Thing?

“Success is a lousy teacher. It seduces smart people into thinking they can’t lose.“

Bill Gates

Questions?

Sources

[1] Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation (Addison-Wesley Signature Series (Fowler)), Jez Humble, David Farley, 2010, ISBN-10: 0321601912

[2] Lean Startup: http://www.slideshare.net/wealthfront/lean-startup-for-geeks-with-eric-ries