

# CONTINUOUS INTEGRATION

# Agenda

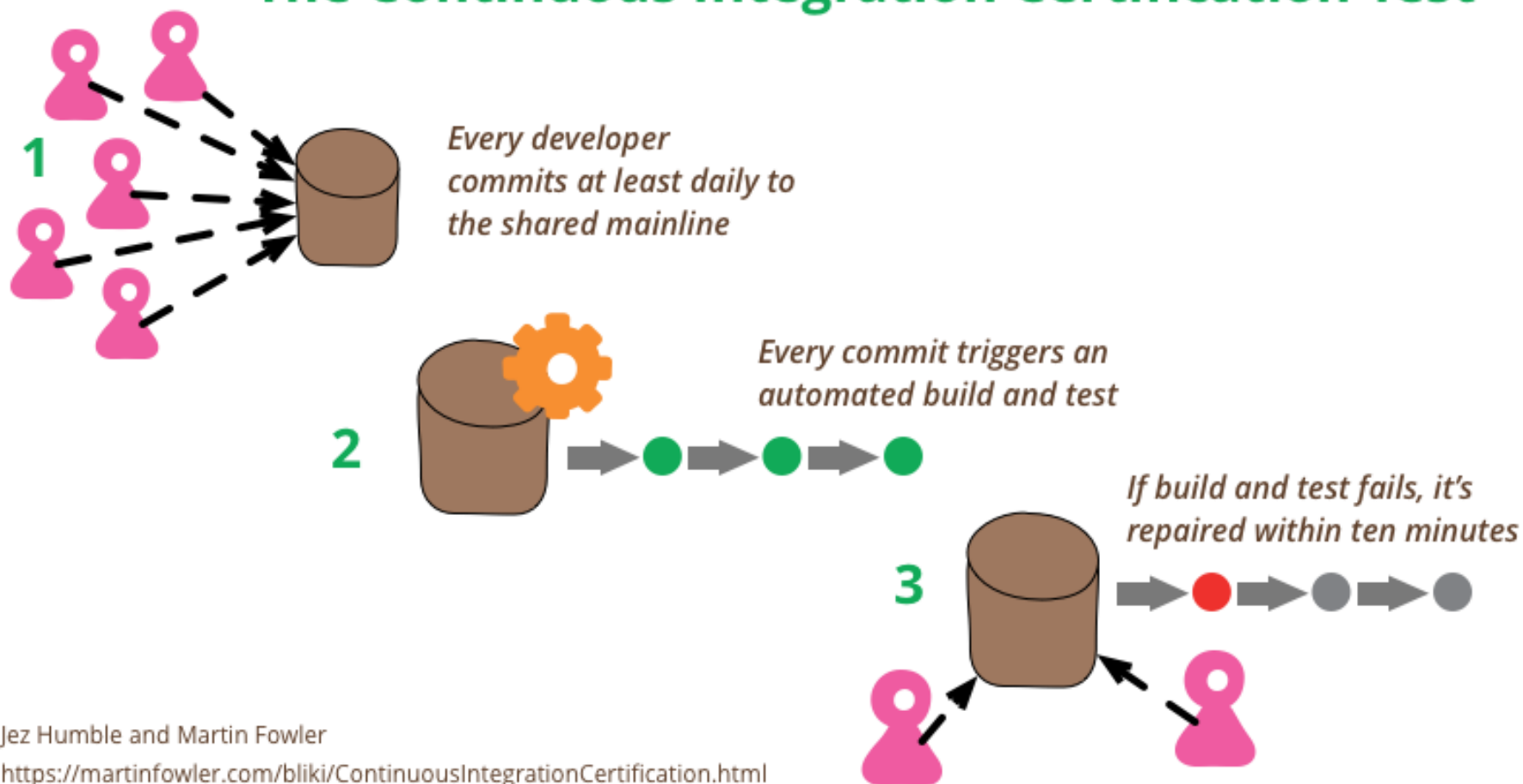
- What is Continuous Integration?
- CI keys
- Continuous Delivery
- Continuous Deployment
- Flow
- CDMM
- Demo

# Qué es Integración Continua? (Continuous Integration)

- Práctica de desarrollo
- Integrar frecuentemente código en un repositorio común
- Cada integración se verifica por un *build* automatizado

# Qué es Integración Continua?

## The Continuous Integration Certification Test



Jez Humble and Martin Fowler

<https://martinfowler.com/bliki/ContinuousIntegrationCertification.html>

# Qué es Integración Continua? (Continuous Integration)

- Se considera un *best practice*
- Debería incluir tests automatizados
- Reduce riesgos, rutinas manuales (pro error)
- Ayuda a identificar problemas pronto

# Qué es Integración Continua? (Continuous Integration)

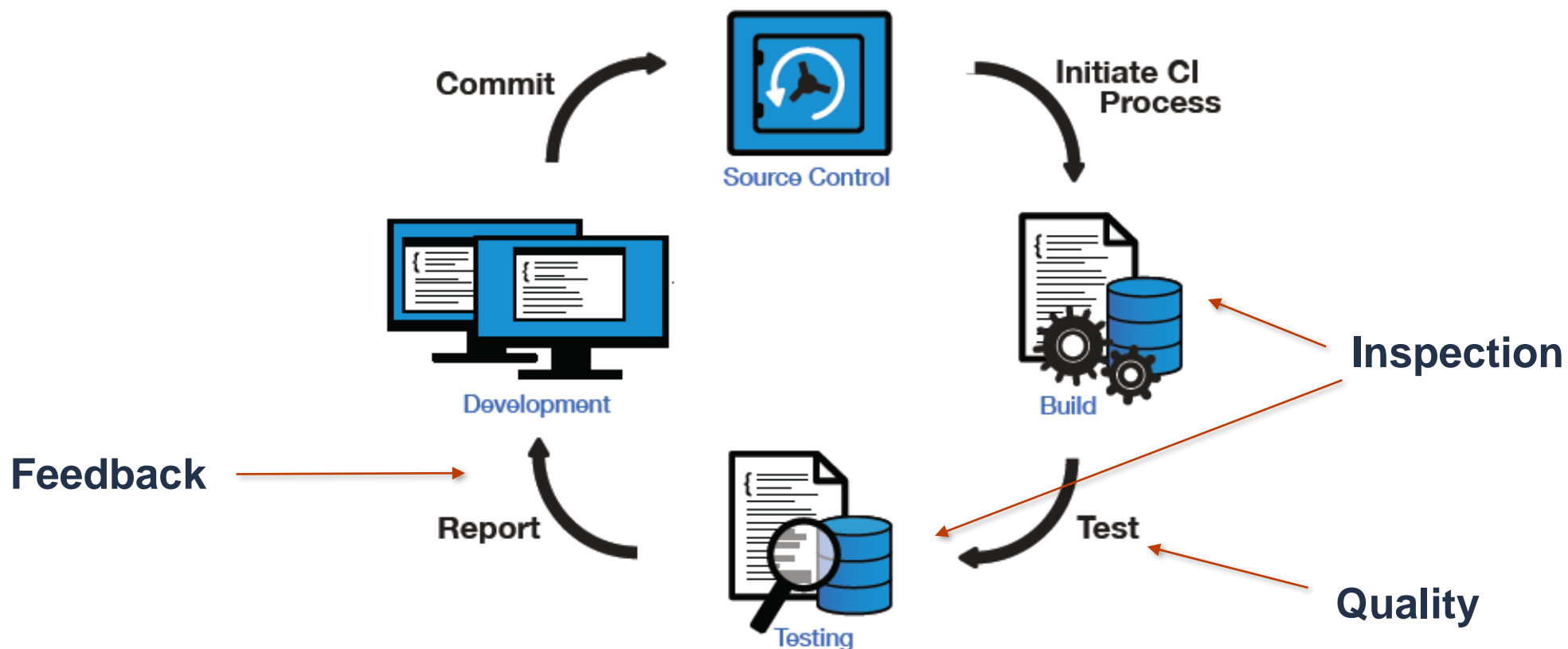
“Continuous Integration doesn’t get rid of bugs, but it does make them dramatically easier to find and remove.”

— Martin Fowler, Chief Scientist, ThoughtWorks

## CI Keys

- **Quality** → Unit Tests
- **Inspection** → Compilation errors, code metrics
- **Feedback** → Fast notifications

# CI Keys





# Continuous Delivery

- Es una extension de CI
- Vamos a través de fases de testing en diferentes ambientes
- Tenemos un software confiable, listo para ser puesto en producción

Necesitamos:

- Buen manejo de ambientes
- Ambientes dedicados para cada fase

# Continuous Delivery

**READY TO DEPLOY TO PRODUCTION AT ANY TIME**

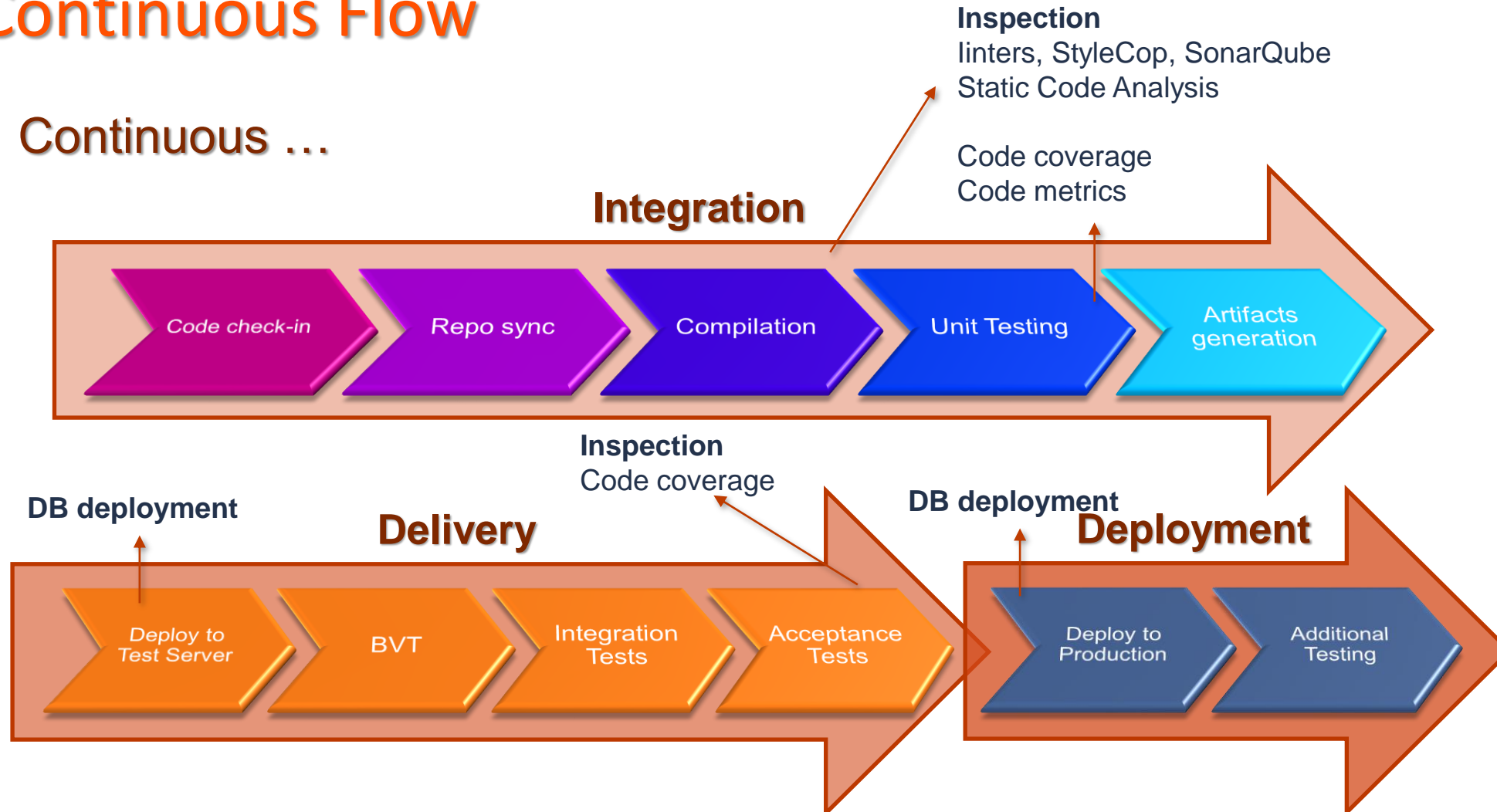
# Continuous Deployment

## CADA BUILD EXITOSO LLEGA A PRODUCCIÓN

- ✓ Retroalimentación rápida sobre *features*
- ✓ Fixes para defectos desplegados inmediatamente (excelente servicio al cliente)
- ✓ Respuesta rápida a situaciones extraordinarias

# Continuous Flow

Continuous ...



# Continuous Delivery Maturity Model

	Novice	Beginner	Intermediate	Advanced	Expert
Build	Automated builds	Artifacts are managed	Automated release notes	Full traceability	Deliver y pipeline
Test	Unit testing, mocks, stubs and proxies	Automated functional tests	Maintain test data	Adaptive test suites	Test in production
Version Control	Commits are tied to tasks	Release train branching strategy	Version numbers matter	Use distributed VCS	Pristine integration branch
DevOps	One Team	Automated deployment	Access to production-like environments	Infrastructure as code	Live monitoring and feedback
Architecture & Design	Code metrics	Testable code	Dependencies are managed	Individually releasable components	Full audit trail in production
Organization & Culture	Agile process	Buy-in from management	Tasks are groomed	Designated roles	Explicit knowledge transfer



**FOUNDATION**

Platform for CD 3.0 available, however the development cycle is still poorly automated

**NOVICE**

CD 3.0 with basic automation on a reactive level

**INTERMEDIATE**

Average CD 3.0 technologies adopted with proactive elements

**ADVANCED**

Advanced CD 3.0 technologies adopted, that are quantitatively managed

**EXPERT**

Decision making and execution is increasingly handed over to machine learning algorithms

**INTELLIGENCE**

- Customer behaviour and feedback server

- Basic monitoring of app usage and handling customer feedback

- Advanced customer monitoring
- A/B testing in place

- All metrics and reports are predefined
- Decision making based on detailed analytics

- Realtime data collection, analysis & reporting using AI

**PLANNING**

- Centralized backlog management server

- All work managed by means of digital backlog

- Automatic backlog item creation

- Automatic proposed backlog prioritization

- backlog creation and prioritization using AI

**INTEGRATION**

- Centralized version control
- Centralized build server

- Nightly builds
- Workflow Orchestrator
- C-Integration reporting

- Automatic build on commit
- One build for all environments

- Staged integrations
- Usage of micro-services
- Realtime integration reporting

- Continuous integration services are automatically up- and downscaled

**TESTING**

- Centralized unit-test server
- Unit tests start manually

- Unit tests run in CD- pipeline
- Automated integration tests that are manually started

- Integration tests in pipeline
- Automated acceptance tests that are manually started

- Acceptance tests in pipeline
- Automated performance and security tests that are manually started
- Behaviour driven development

- Continuous Testing pipeline including end-2-end regression tests

**DEPLOYMENT**

- Deployment server

- Basic deployment scripts
- Automatic deployment to test environment after successful build

- Automatic deployment pipeline from build to production

- Zero downtime deployments

- Deployments on endless scalable platforms

# Continuous Delivery Maturity Model

# DEMO