escuela técnica superior
de ingeniería informática

Gestión de la construcción e integración continua
Build engineering and continuous integration

Departamento de
Lenguajes y Sistemas Informáticos

EGC

Varias páginas de integración continua están adaptadas de Marty Stepp
http://www.cs.washington.edu/403/ y Bruce Altner y Brett Lewinski (NASA)

Ejemplo de otro dominio

ENSAMBLAR BICIS

¿Qué hay que hacer para “ensamblar” /”construir” software?

- Introducción
- Conceptos básicos
- Integración continua
- Resumen
- Bibliografía

¿Qué diferencia hay entre construir (*build*), integrar, empaquetar, entregar (*release*), desplegar (*deploy*) software?

- Introducción
- Conceptos básicos
- Integración continua
- Resumen
- Bibliografía

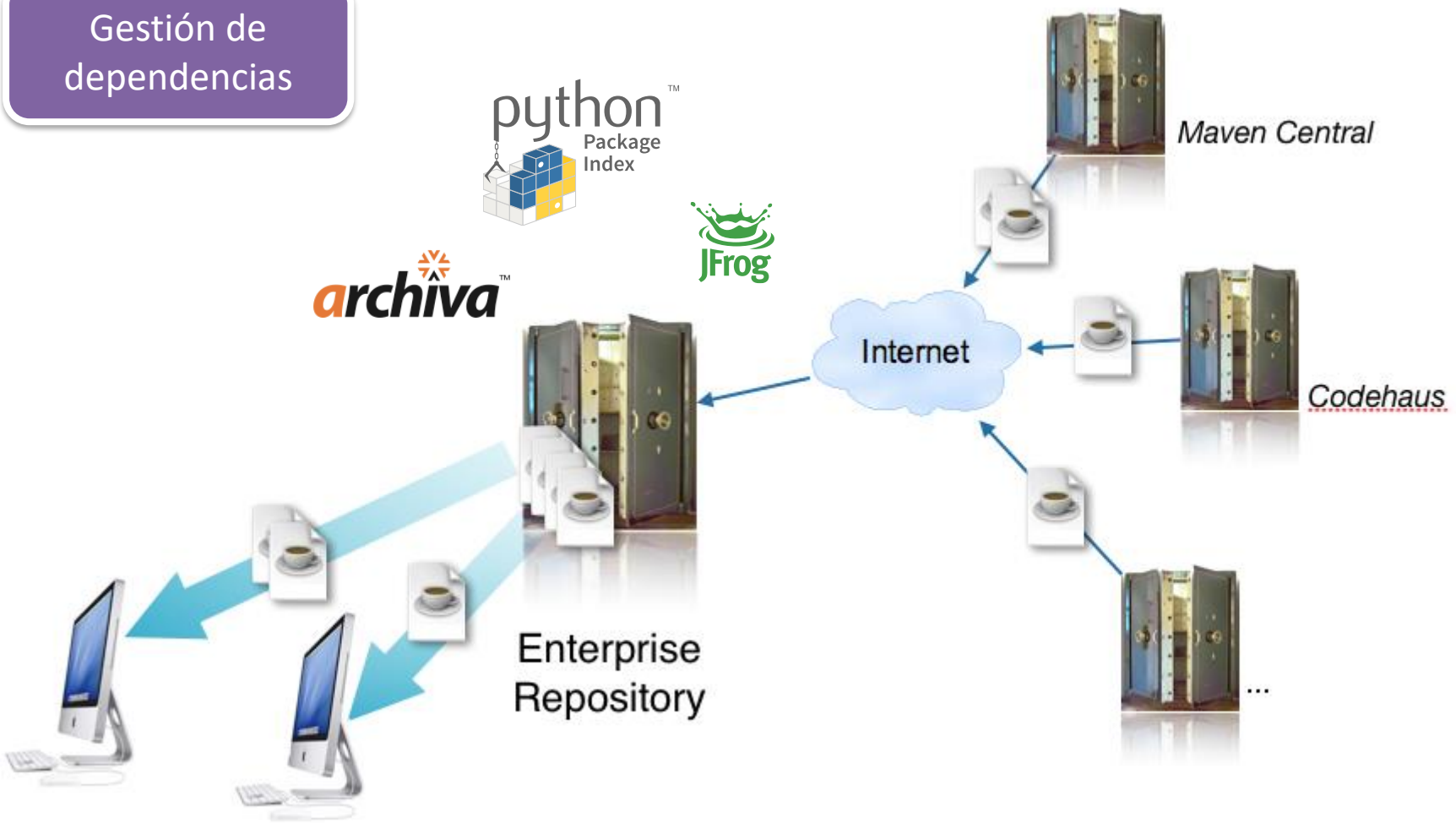
Construcción

Construcción

- Gestión de dependencias
- Empaquetado
- Versionado

Repositorio de artefactos

Gestión de dependencias



Compilaciones locales o remotas

Empaquetado

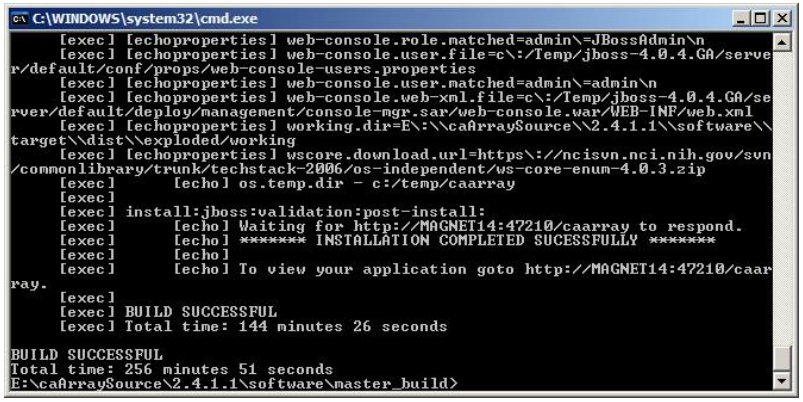
Puesto 1

Puesto 2

Puesto n

Repositorio de código

Servidor de compilación



Versionado de los resultados

- Basado en secuencias:
  - Importancia del cambio:
    - X.Y.Z
      - X: cambios sustanciales en funcionalidad
      - Y: cambios menores en funcionalidad
      - Z: cambios menores, no hay cambios de funcionalidad
    - Estado de la versión:
      - Alpha: primera liberación, alpha, alpha1, alpha2,..
      - Beta: fase inicial, beta, beta1, beta 2, ….
      - Release candidate: candidata a versión final, rc, rc1, rc2,..
      - Final release: liberación final
  - Fecha de liberación:
    - Ubuntu 5.10, 10.04, 17.10, etc..
    - Wine 20040505

Versionado

```
$>python3 --version
Python 3.8.5
```

Alternativas

Versionado

- Manuales:
  - https://github.com/EGCETSII/decide/releases
- Automáticas:
  - "automating semantic versioning" (http://semver.org/)

Integración

Integración

- **Integración**: Combinar 2 o más unidades de software.
  - A menudo un subconjunto del total del proyecto

¿Por qué hay que preocuparse de la integración?

Integración

- **Integración**: Combinar 2 o más unidades de software.
  - A menudo un subconjunto del total del proyecto

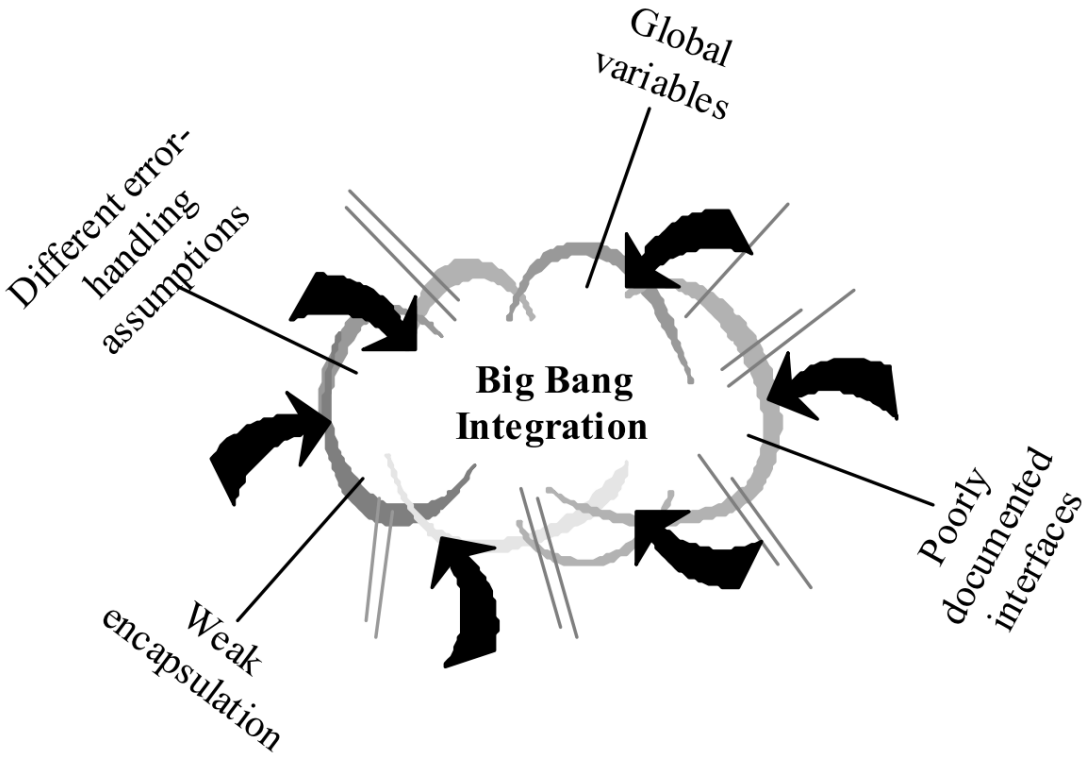
¿Por qué hay que preocuparse de la integración?

"Siempre que haya algo difícil ("painful"), hay que hacerlo cuánto antes mejor" [Humble]

¿Y cómo integramos?

Integración por fases

- **Integración por fases ("big-bang")**:
  - Diseño, códificación, pruebas, depuración, cada clase/unidad/subsistema de manera separada.
  - Lo combinamos todo
  - "rezamos"



Introducción

Conceptos básicos

Integración continua

- Definición
- Arquitectura
- Builds
- Pruebas y CI
- Inspección y análisis de código
- Feedback
- Lo esencial

Resumen

Bibliografía

Continuous Integration

- Idea nacida de Martin Fowler; parte de lo que se conoce como XP y métodos ágiles
- 10 principios:
  1. Mantener un único repositorio de código
  2. Automatizar los *build*
  3. Hacer que los *build* sean auto-testeables
  4. Todo el mundo hace *commit* a la línea principal *una vez al día*
  5. Todos los *commits* deben lanzar un trabajo de integración en una máquina de integración
  6. Haz que los *build* sean rápidos y cortos
  7. Haz las pruebas en un clone del entorno de producción
  8. Haz que sea fácil para todo el mundo acceder al último ejecutable
  9. Todo el mundo puede ver lo que está pasando
  10. Automatizar los despliegues

Introducción

Conceptos básicos

Integración continua

- Definición
- Arquitectura
- Builds
- Pruebas y CI
- Inspección y análisis de código
- Feedback
- Lo esencial

Resumen

Bibliografía

Aquitectura del sistema de CI

The key to fixing problems quickly is finding them quickly.
– (Fowler, 2006)

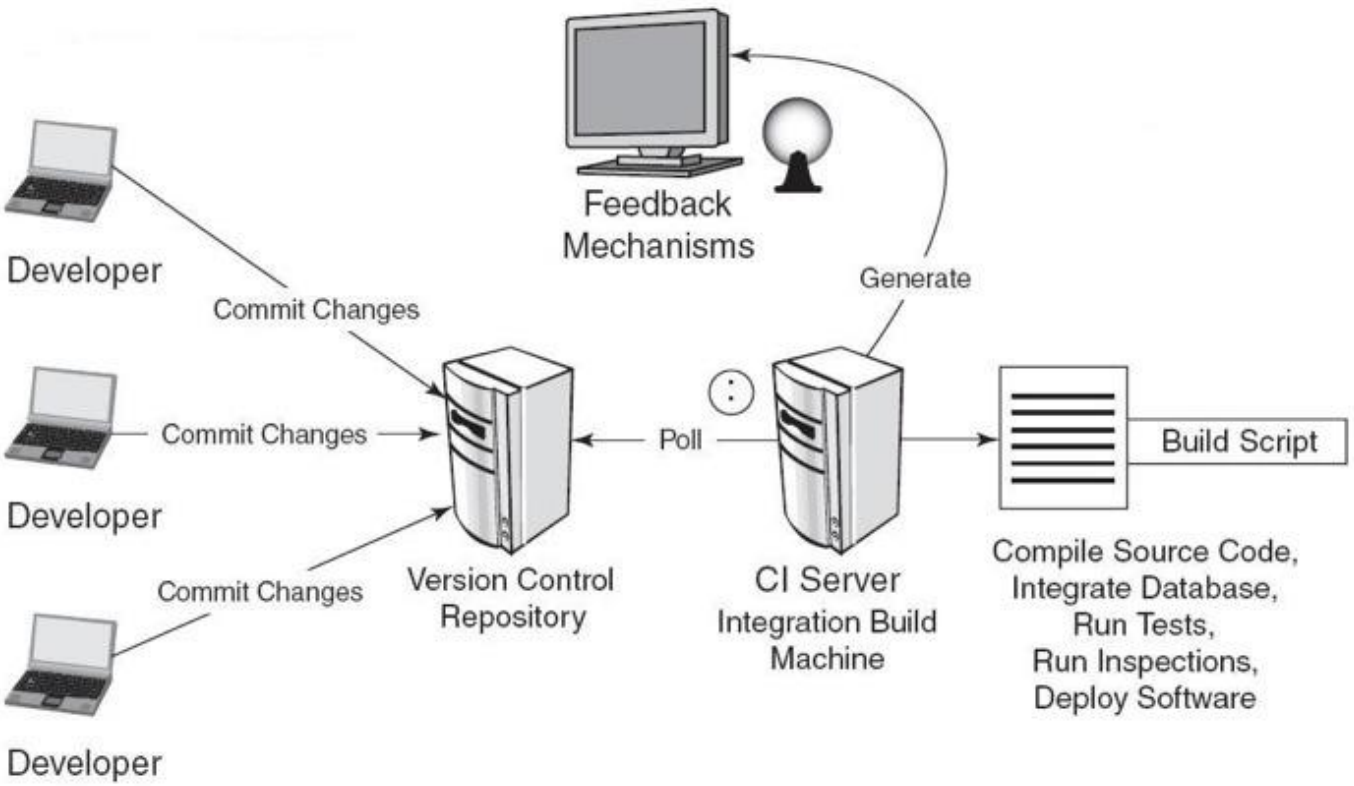
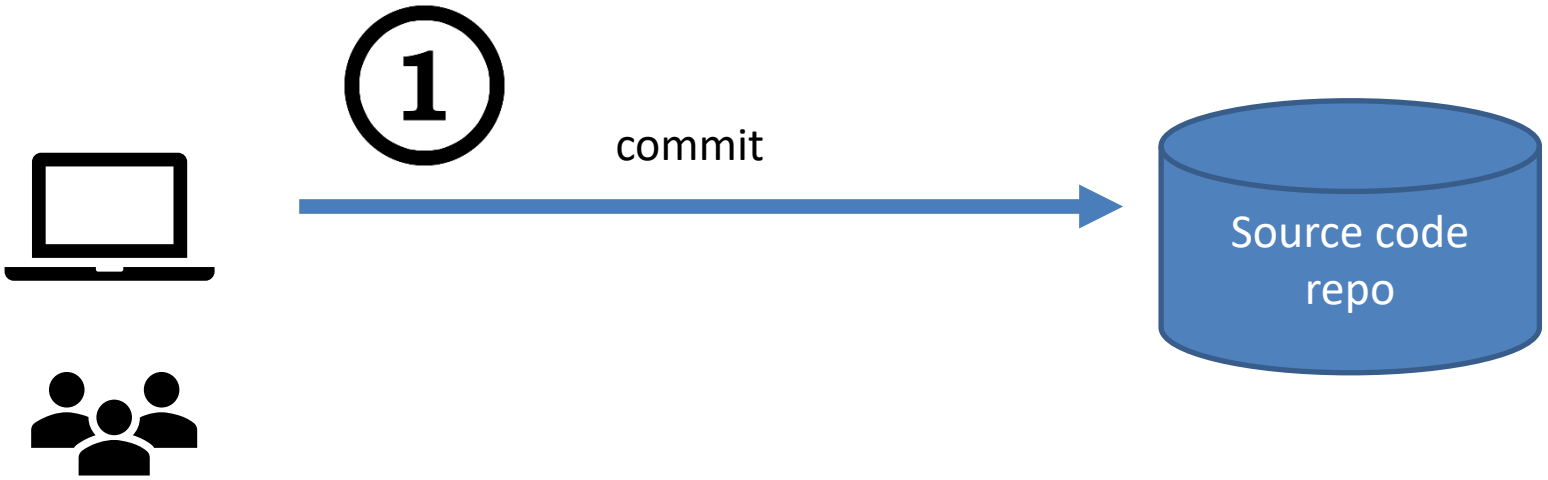


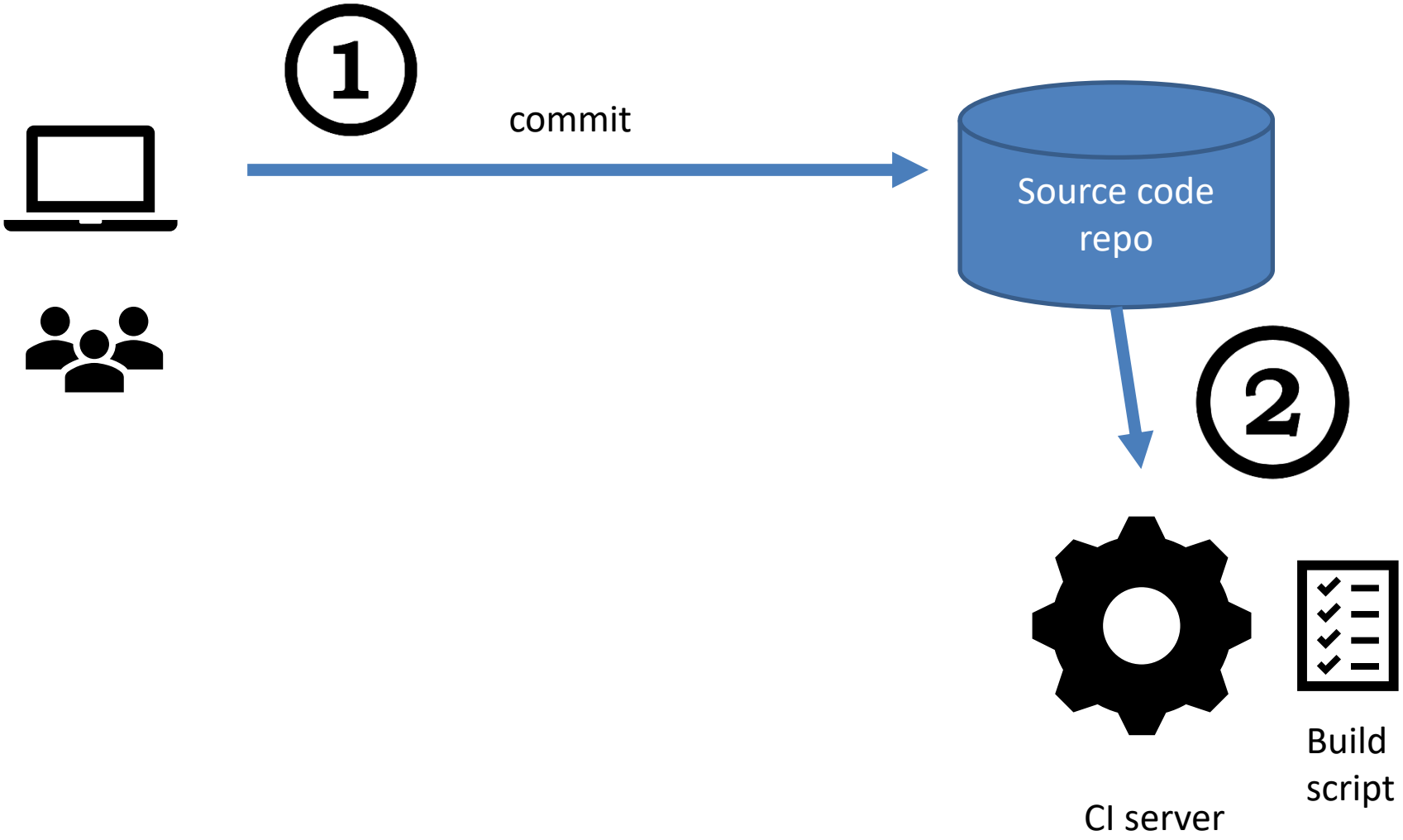
Figure 1: System and Software Architecture Supporting a CI Build

Definition: CI is the practice of regular, comprehensive, and automatic building and testing of applications in software development.

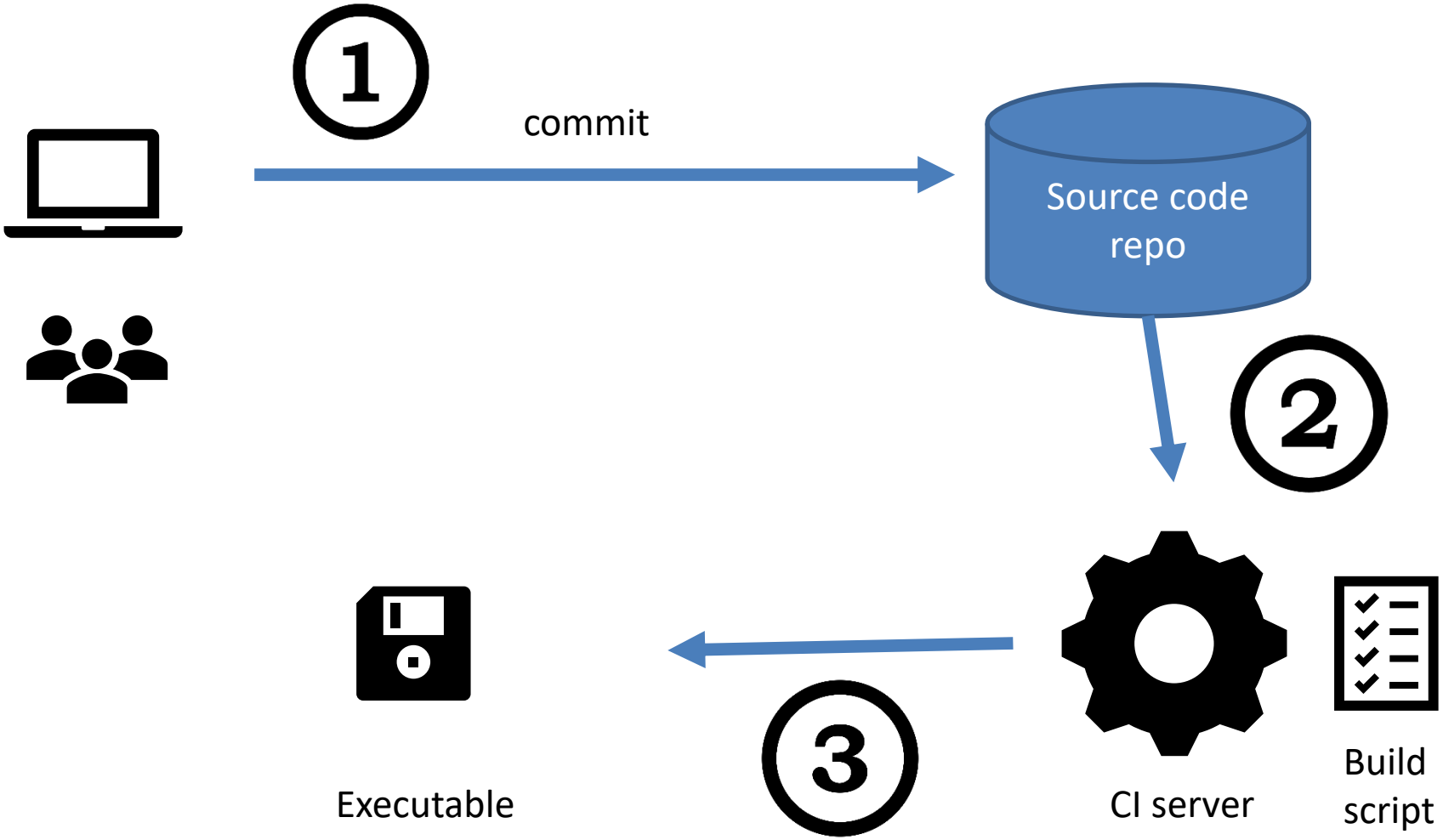
Ciclo de CI



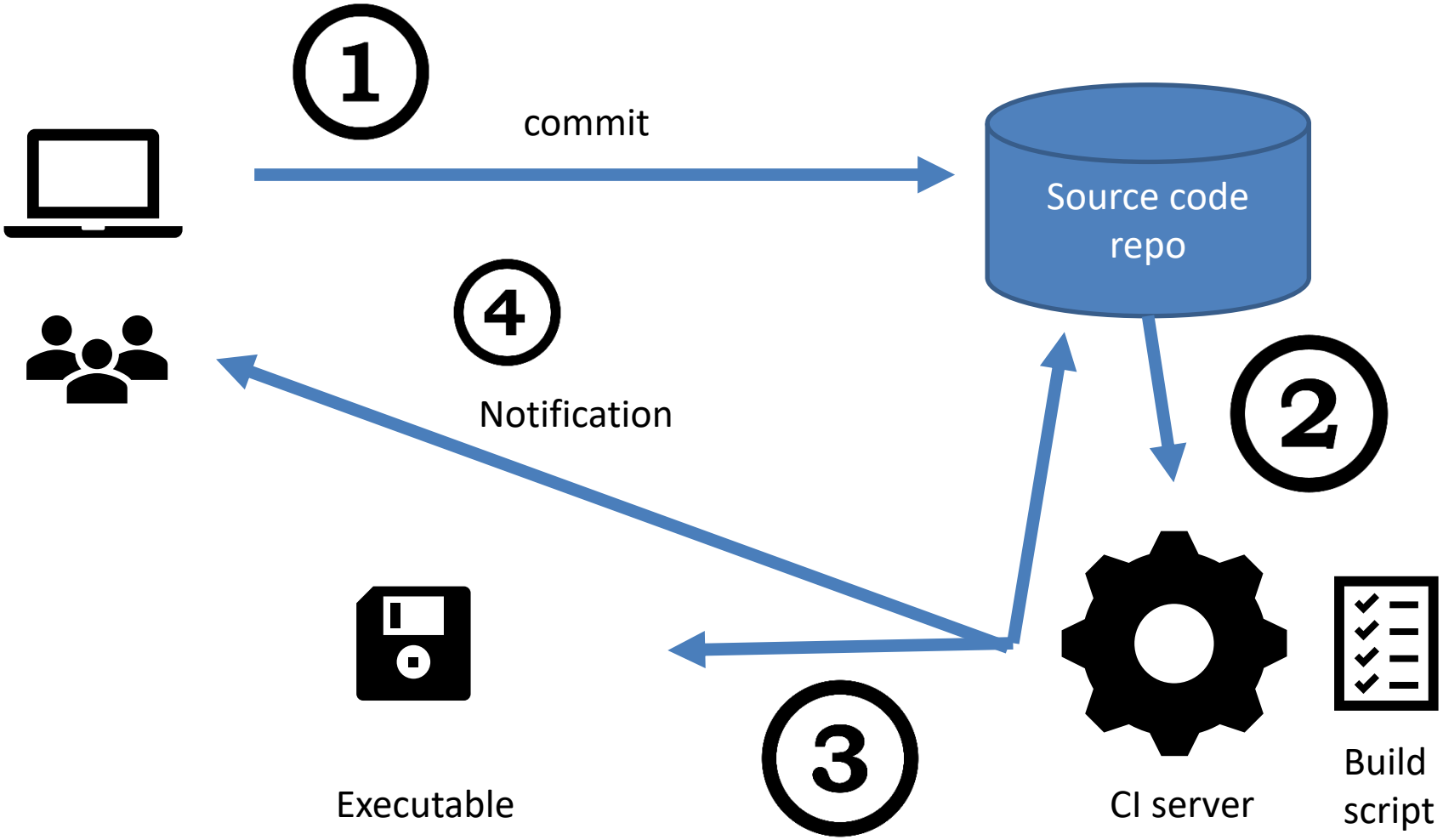
Ciclo de CI



Ciclo de CI


1
commit
Source code repo
2
3
Executable
CI server
Build script

Ciclo de CI


1
commit
Source code repo
4
Notification
2
Executable
3
CI server
Build script

La metáfora del botón de integración

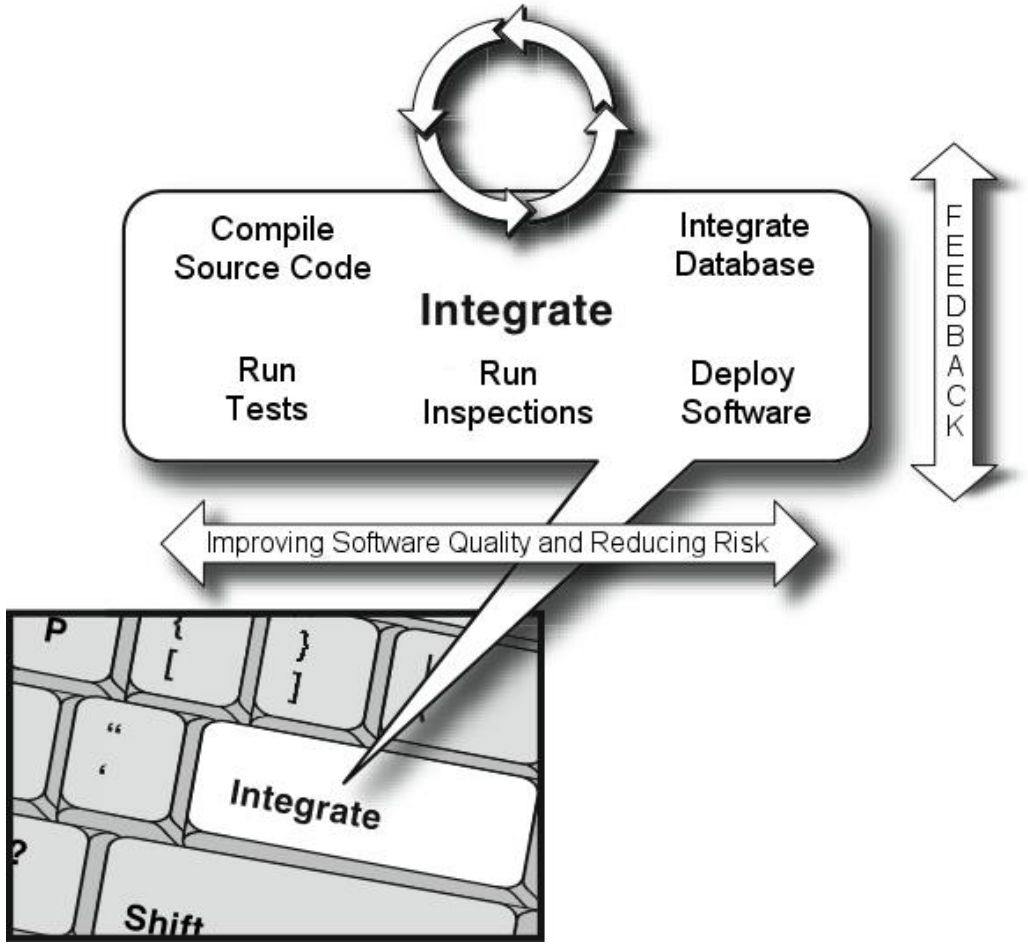


Figure 2: The “Integrate Button” Metaphor

Source: Duvall et al. (2007)

Artefactos necesarios

- Normalmente un build necesita algo más que solo source code. Otros posibles elementos:
  - Elementos de terceros o de otras partes del proyecto
  - Ficheros de configuración
  - Ficheros de datos o scripts de datos
  - Scripts de tests
  - Scripts de construcción

Todos los artefactos excepto los artefactos de terceros deben estar almacenados en el repositorio de Código
¿Qué hacemos con los binarios?

Introducción

Conceptos básicos

Integración continua

- Definición
- Arquitectura
- Builds
- Pruebas y CI
- Inspección y análisis de código
- Feedback
- Lo esencial

Resumen

Bibliografía

Daily commits

"Todo el mundo hace *commit* a la línea principal una vez al día."

daily commit

- Idea: Reducir los conflictos de merge; evitar problemas de integración.

Cuidado: No hagas commit de código con errores (que no compila, que no pasa las pruebas) solo por mantener esta práctica.

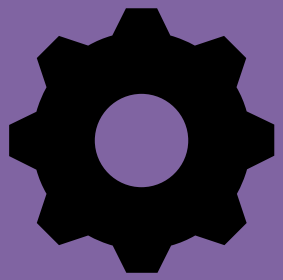
Si el código no está listo para enviarlo al final del día, o bien envía un subconjunto coherente del código o bien se más flexible con la planificación

Construcción

- Gestión de dependencias
- Empaquetado
- Versionado

Tipos de *builds*

Privados



CI server

Integración

Entrega

Daily builds

"Automate the build."

- ***Build* diario**: Construir ejecutables de trabajo diariamente:
  - permite probar la calidad de la integración hasta el momento
  - ayuda a la moral del equipo: el producto "funciona todos los días" y el progreso es visible
  - Fácil y más bien rápida detección de cualquier "issue" que rompa la compilación
- **Continuous Integration (CI) server**: Una máquina externa que toma automáticamente la última versión del código del repo y construye (en una serie de pasos)

* Kent Beck, en su libro Extreme Programming Explained (2004), sugierea 10-minutos como límite para integración básica. Incluso 10 minutos puede parecer mucho en algunos casos.

Introducción

Conceptos básicos

Integración continua

- Definición
- Arquitectura
- Builds
- Pruebas y CI
- Inspección y análisis de código
- Feedback
- Lo esencial

Resumen

Bibliografía

Automated tests

"Make your build self-testing."

- **Pruebas automáticas**: La importancia de la línea de cómandos para poder hacer pruebas
  - Distintos tipos de pruebas: test unitarios, de covertura, *static analysis / style checking*, ...

Introducción

Conceptos básicos

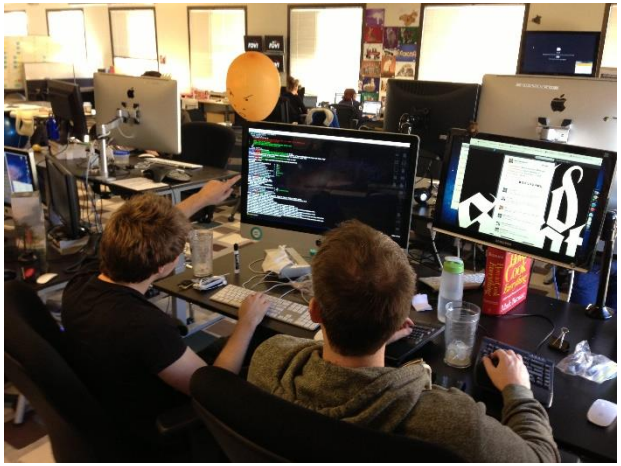
Integración continua

- Definición
- Arquitectura
- Builds
- Pruebas y CI
- Inspección y análisis de código
- Feedback
- Lo esencial

Resumen

Bibliografía

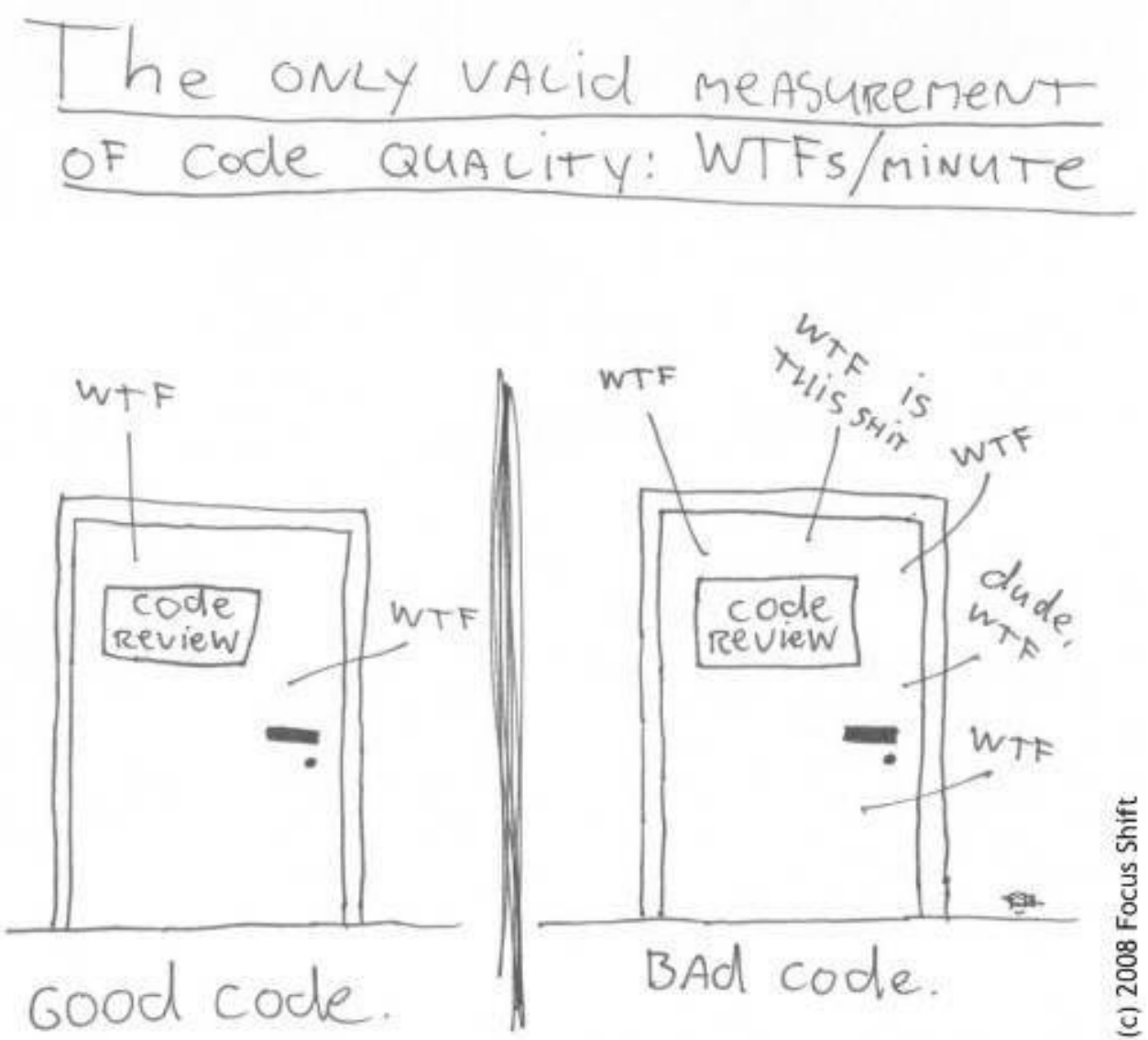
Inspección vs análisis de código



Código de calidad usando herramientas

Tanto la inspección como el análisis de código pueden servir para obtener un código de mejor calidad (recordar cosas dadas en PGPI). Este tipo de prácticas, podemos automatizarlas usando algunas herramientas de análisis estático de código o podemos institucionalizarlas llevando a cabo prácticas de inspección de código por pares, por ejemplo

La calidad del código



¿Qué hacer con inspección de código y qué con análisis estático?

¡Buscar ejemplos!

Introducción

Conceptos básicos

Integración continua

- Definición
- Arquitectura
- Builds
- Pruebas y CI
- Inspección y análisis de código
- Feedback
- Lo esencial

Resumen

Bibliografía

Feedback

¿Qué información se le puede dar al usuario? ¿Cómo? ¡Buscar ejemplos!

Introducción

Conceptos básicos

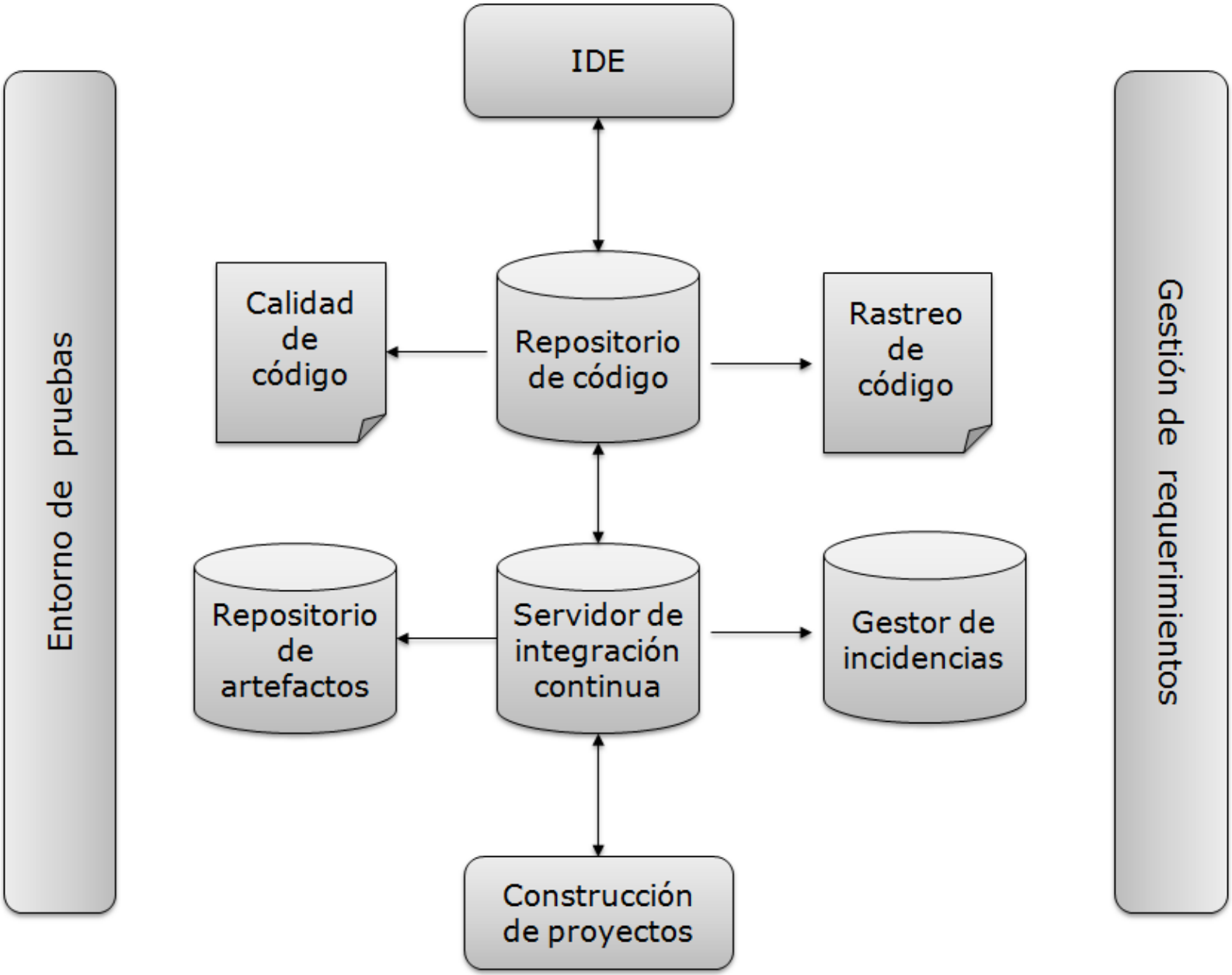
Integración continua

- Definición
- Arquitectura
- Builds
- Pruebas y CI
- Inspección y análisis de código
- Feedback
- Lo esencial

Resumen

Bibliografía

Ecosistema de desarrollo con CI



Índice

Resumen

- ¿Qué hemos aprendido?
  - Construir/ensamblar software es algo más que "compilar"
  - La integración es un problema que hay que tratar
  - La integración continua aboga por hacerlo desde el primer momento
  - Automatizar la construcción se convierte en fundamental
- ¿Qué veremos en las siguientes lecciones?
  - Gestión de las incidencias, pruebas, código…
  - En práctica herramientas de construcción, servidores de integración continua

Índice

Bibliografía