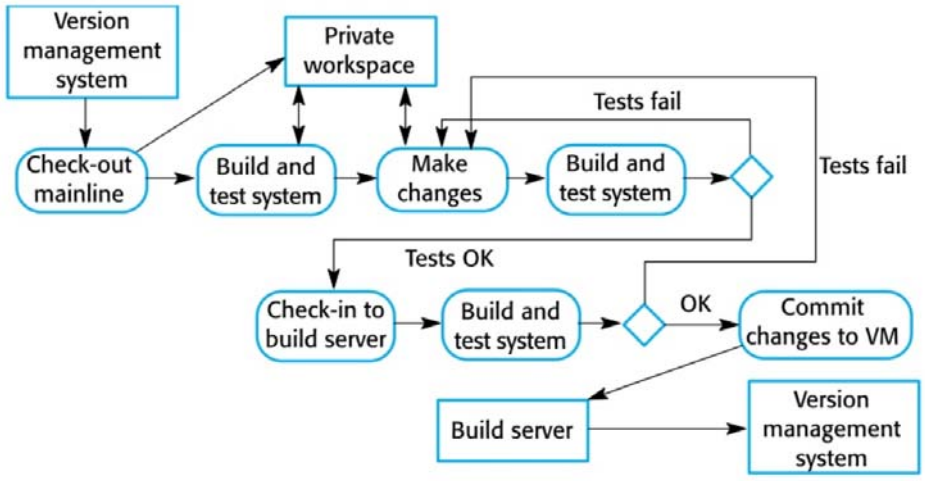
# PROCESOS DE LA INGENIERIA SOFTWARE
# Tema 2
## *Construcción Automatizada de Sistemas Software*

***Universidad de Cantabria – Facultad de Ciencias***
***Patricia López Martínez***

## Integración continua

- La integración continua es una práctica de desarrollo software donde los miembros del equipo integran su trabajo frecuentemente
  - Cada miembro  al menos una vez al día
  - Cada integración se verifica con un *build* automático, que incluye la compilación, enlazado y ejecución de pruebas unitarias y de integración
  - El objetivo es detectar errores de integración tan pronto como es posible
- Concepto introducido por Martin Fowler
  - www.martinfowler.com/articles/continuousIntegration.html
- Muy relacionado con metodologías ágiles
  - Una de las 12 prácticas de Extreme Programming

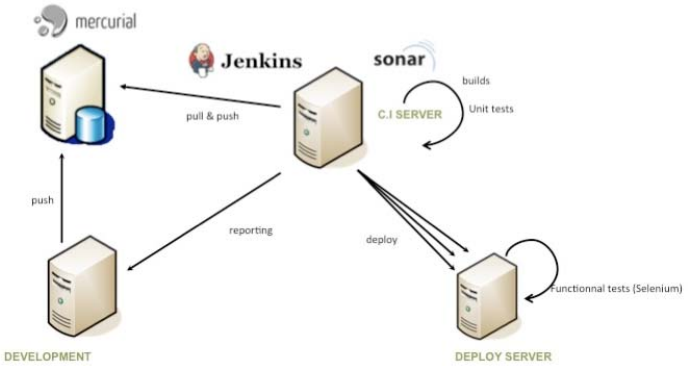
## Integración continua - Proceso




Sommerville, 2010



## Integración continua – Infraestructura




- Servidores de integración continua:
  - Java: Jenkins, Hudson, CruiseControl, Bamboo, Continuum, Anthill
  - .NET: CruiseControl.Net, Team FoundationBuild
  - Apoyados en herramientas de construcción automatizada:
    - Ant/Maven para Java o Nant/MSBUILD para .NET

## Integración continua – Ventajas

- Los desarrolladores pueden detectar y solucionar problemas de integración de forma continua, evitando el caos de última hora cuando se acercan las fechas de entrega
- Disponibilidad constante de una versión para pruebas, demos o lanzamientos anticipados
- Ejecución inmediata de las pruebas unitarias y de integración
- Monitorización continua de las métricas de calidad del proyecto
- En definitiva, reducción del riesgo y aumento de la calidad



## Integración continua – Buenas prácticas

1. Mantener un solo repositorio de código => Subversion, Git
2. Automatizar la construcción (build) => Ant, Maven
3. Incluir testing automatizado en el build (Self-testing build) => Xunit, Selenium
4. Todos los desarrolladores deben hacer un commit al mainline al menos una vez al día
5. Cada commit debe hacer un build en la máquina de integración (más fácil con el servidor de integración continua)
6. Corregir errores encontrados en el build inmediatamente
7. Minimizar el tiempo del build (testing en dos fases, p.e.)
8. Ejecutar los tests en un entorno lo más parecido posible al de producción
9. Proceso visible para todo el mundo involucrado