METODOLOGÍA DE DESARROLLO ÁGIL

CONTINUOUS INTEGRATION (INTEGRACIÓN CONTINUA).

Guillermo Alexander Cornejo Argueta

e-mail: guillermobrs@gmail.com

Lance Albeiro Peraza Cruz

e-mail: lap.cruz96 @gmail.com

Nathaly Marcela Amaya Agreda

e-mail: mamita.a@hotmail.com

Adonay Absalón Barahona Ramos

e-mail: adonay\_barahona@hotmail.com

Francisco Everardo Gómez Évora

e-mail: fendergx@live.com

**RESUMEN:** *El resumen se elaborará cuando toda la investigación esté completada.*

**PALABRAS CLAVE**: integración continua, metodología de desarrollo ágil.

# 1 OBJETIVOS

**1.1 OBJETIVO GENERAL**

Desarrollar una investigación descriptiva y aplicativa mediante el uso de diversas técnicas de búsqueda de artículos especializados, para comprender una de las prácticas de desarrollo ágil en el proceso de construcción de software como lo es la Integración Continua.

**1.2 OBJETIVOS ESPECÍFICOS**

1- Conceptuar todos los términos esenciales acerca de las metodologías de desarrollo ágil en el área de la práctica de integración continua, haciendo uso de la investigación aclarativa para lograr comprender y analizar los aspectos involucrados.

2- Analizar la importancia de la implementación de la integración continua en el marco del desarrollo de software, basándose en ciertas experiencias de aplicación, para demostrar el uso de esta metodología en el ámbito empresarial.

3- Definir los principios clave de la integración continua para lograr una efectiva implementación de dicha práctica al momento de desarrollar un software.

4- Analizar el grado de implicación que se tiene entre las prácticas posteriores a la integración continua, en el proceso de desarrollo de software haciendo uso de investigaciones sobre dichas prácticas, para comprender completamente la relación y dinámica de trabajo entre estas.

# 2 INTRODUCCIÓN

Pendiente.

# 3 METODOLOGÍAS ÁGILES

Las metodologías agiles son métodos para el desarrollo de software que permiten adaptar la forma de trabajo a las condiciones del proyecto, permitiendo gran flexibilidad e inmediatez en la respuesta para moldear el proyecto y su desarrollo a las circunstancias específicas del entorno. Se caracterizan por enfatizar la comunicación frente a la documentación y por el desarrollo evolutivo.

Los beneficios obtenidos de implementar metodologías agiles en una empresa son:

1. Mejoran la satisfacción del cliente, dado que este se ve involucrado a lo largo del desarrollo del proyecto. El cliente puede agregar su experiencia para optimizar las características del producto final. Esto reduce los malentendidos, ya que el cliente poseerá en todo momento una completa visión del estado del proyecto.
2. Mejora la motivación e implicación del equipo encargado del desarrollo del software. Como con el cliente, esto permite que todos los miembros del equipo estén al tanto del estado del proyecto en cualquier momento. Cualquier compromiso es negociado y aceptado por todos los integrantes del equipo, teniendo presente las ideas de cualquiera de ellos.
3. Ahorra tiempo y dinero. La forma de trabajo del desarrollo ágil es más eficiente y eficaz que otras metodologías.
4. El trabajo se realiza en menos tiempo. El proyecto se desarrolla realizando entregas parciales pero funcionales del producto, lo que permite entregar en el menor intervalo de tiempo posible una versión funcional del producto.
5. Elimina características innecesarias del proyecto, gracias a la participación activa del cliente y a las entregas centradas en las funcionalidades que más valor aportan al producto.
6. Mejora la calidad del producto. Gracias a la continua interacción entre el equipo de desarrollo y los clientes, se asegura que el producto final sea exactamente lo que el cliente desea y necesita.
7. Permite identificar rápidamente problemas o errores, ahorrando así el tiempo y costo relacionados a la solución de dichas dificultades.
8. Permite que la empresa rentabilice las inversiones más rápidamente, gracias a la realización de entregas tempranas, el cliente tiene rápido acceso a las funcionalidades que aportan valor, acelerando el entorno de la inversión.

**4 INTEGRACION CONTINUA**

La integración continua (CI) es una práctica donde un equipo de desarrolladores integra su código repetidas veces un lapso corto, generalmente varias veces al día.

El propósito es evitar el “Integration hell” llamado así el punto en el que los desarrolladores constituyen su código individual en un solo proyecto sin haberlo hecho durante toda la etapa de desarrollo. En entornos de desarrollo tradicional este proceso rara vez es llevado a cabo de forma fluida y sin contratiempos. El resultado de una mala integración puede derivar en horas o incluso días de retraso arreglando código.

Uno de los beneficios principales de la CI es ahorrarle tiempo al equipo de desarrollo durante el ciclo de trabajo identificando y corrigiendo conflictos tempranamente. Representa también una excelente forma de reducir el tiempo requerido para corregir errores.

Enfatiza el uso de un banco de pruebas y conocimiento compartido del código.

# 5 COMPARACION DE PRACTICAS DE DESARROLLO

Las prácticas que suceden a la integración continua (CI) son la Entrega continua y despliegue continuo. Dichas técnicas son utilizadas frecuentemente en el ámbito de desarrollo de software, sin embargo aunque ambas tienen mucho en común no se deben confundir ambos términos ya que las diferencias son significativas.

## 5.1 ENTREGA CONTINUA

La entrega continua es una extensión de la CI que se encarga de asegurar que nuevos cambios se entreguen en cualquier momento de forma rápida y sostenible. Esto requiere la automatización del proceso de lanzamiento (release). Una de las ventajas más grandes de esta práctica es que el lanzamiento puede ser diario, semanal o el tiempo que mejor se acople a las necesidades del negocio

.

Lo recomendable es entregar las mejoras lo más rápido posible para mantener los cambios entre versiones de un tamaño manejable, más fácil de corregir en caso de errores y en ciclos cortos de trabajo.

Podemos definir entonces a la entrega continua como: Una práctica de desarrollo de software mediante la cual se crean, prueban y preparan automáticamente los cambios en el código y se entregan para la fase de producción.

Se alcanza todo esto asegurando que el proyecto está siempre en un estado estable y desplegable,

## 5.2 DESPLIEGUE CONTINUO

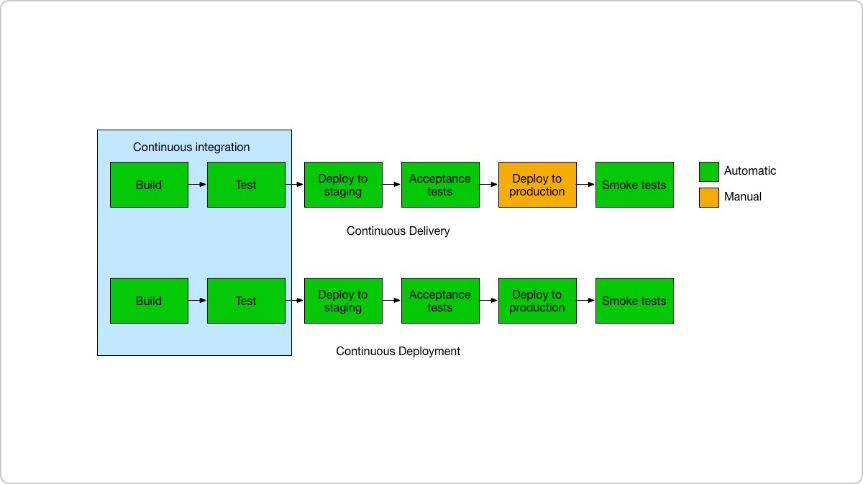
El despliegue continuo va un paso más allá de la entrega continua. Con esta práctica, cada cambio que pasa todas las etapas de pruebas para producción se libera a sus clientes. Todo se realiza automáticamente, y solo una prueba fallida evitará que se implemente un nuevo cambio en la producción.

La implementación continua es una excelente manera de acelerar el ciclo de retroalimentación con los clientes y de quitar presión al equipo de desarrollo, ya que no hay fecha de lanzamiento. Los desarrolladores pueden centrarse en la creación de software y ver cómo una mejora se despliega minutos después de que hayan terminado de trabajar en ella.

**5.3 RELACION ENTRE PRÁCTICAS CI y CD**

La integración continua es parte esencial de la entrega continua y despliegue continuo.

Para tener Despliegue continuo antes debe haber Entrega continua y en cualquiera de los casos debe haber una buena integración continua ya que es la base de la automatización en el proceso de desarrollo de software con metodologías ágiles.



**5.4 PASANDO DE INTEGRACION CONTINUA A DESPLIEGUE CONTINUO**

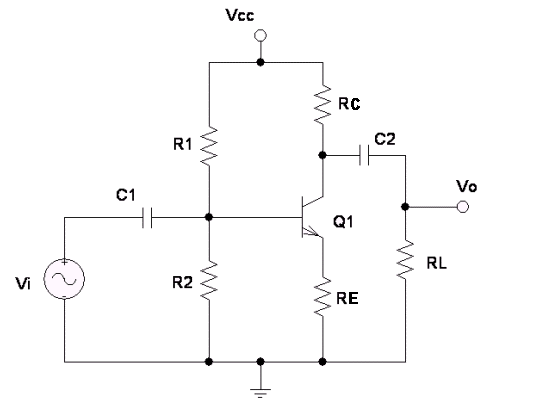


Figura 2. Configuración de emisor común.

### CITAS Y/O REFERENCIAS

Las citas y/o referencias se colocarán al final del manuscrito. Utilice Arial, 8 pts, espacio simple. Para ayudar a los lectores, evite notas a pie de página que incluyen las observaciones periféricas necesarias en el texto (dentro de paréntesis, si usted prefiere, como en esta oración). Las citas deberán de respetar el orden de aparición en las referencias.

Se colocarán entre corchetes Ej. [2].

Si es preciso mencionar los nombres de los autores deberán de aparecer todos los nombres exceptuando si el numero de éstos es más de cuatro, en tal caso se pondrá el nombre del primer autor y la leyenda ‘et al’.

Si la frase inicia citando la referencia entonces puede utilizar el formato Ref. [4], en otro caso utilice solo [4].

Las referencias electrónicas (URL) deben seguir el formato mostrado en [6].

# REFERENCIAS

[1] G. Obregón-Pulido, B. Castillo-Toledo and A. Loukianov, *“A globally convergent estimator for n frequencies”*, IEEE Trans. On Aut. Control. Vol. 47. No 5. pp 857-863. May 2002.

[2] H. Khalil, *”Nonlinear Systems”*, 2nd. ed., Prentice Hall, NJ, pp. 50-56, 1996.

[3] Francis. B. A. and W. M. Wonham, *“The internal model principle of control theory”*, Automatica. Vol. 12. pp. 457-465. 1976.

[4] E. H. Miller, *“A note on reflector arrays”*, IEEE Trans. Antennas Propagat., Aceptado para su publicación.

[5] *Control Toolbox* (6.0), User´s Guide, The Math Works, 2001, pp. 2-10-2-35.

[6] J. Jones. (2007, Febrero 6). Networks (2nd ed.) [En línea]. Disponible en: <http://www.atm.com>.

**Notas:**

1. En general una referencia debe de contener el nombre del autor(es), el Nombre del articulo o libro en *itálicas*, Edición y editorial ó nombre de la revista, volumen y número, paginas y finalmente el mes y año o solo el año si es un libro o comunicado.
2. Observe el formato del encabezado: Nombre Institución. Apellido Autor1, Apellido Autor2, etc. Título abreviado del artículo.
3. Es permitido utilizar tipo de letra Times New Roman en lugar de tipo Arial, pero debe utilizarse el mismo tipo de letra en todo el documento y aumentar en 1 punto el tamaño respecto de los que se señalan en el presente documento.

Adaptado por:

Ing. William Marín, para los cursos de laboratorio.

Escuela de Ingeniería Electrónica

Instituto Tecnológico de Costa Rica

2007