A picture containing text

Description automatically generated

Especificación y diseño de software

Prof. Daniel Eduardo Madriz Huertas.

Semestre I

2022

Investigación #1: Unit Testing

José Alejandro Chavarría Madriz 2019067306

Sebastián Mora Godínez 2019??????

Índice

**No table of contents entries found.**

# Pruebas unitarias

## Definición

## Beneficios y limitaciones

Las pruebas unitarias, así como muchos otros métodos de pruebas automatizadas, han tomado muchísima importancia en la escena del desarrollo de software. En muchas organizaciones las pruebas automatizadas son completamente obligatorias, el código y sus pruebas son uno solo. [lab] Y por supuesto, las pruebas unitarias son parte principal del proceso de testing.

Esta importancia que reciben las pruebas unitarias está completamente fundamentada en los beneficios que traen consigo. Dentro de los principales beneficios se puede citar [lab] [kralj]:

* Mejora la calidad del código: La simple necesidad que escribir código que debe ser testeado mas tarde obliga al programador de ser consciente de la manera en que escribe su código. Debe asegurarse de que su cogido tenga alta cohesión y poco acoplamiento, es decir, que las dependencias de su código sean mínimas y accesibles, así como mantener las clases pequeñas y específicas.
* Encontrar bugs tempranamente: Gracias a la retroalimentación que ofrecen las pruebas unitarias muchos errores pueden ser encontrados en durante las pruebas y no en producción. Los programadores, asumiendo que existe la cantidad necesaria de pruebas unitarias, son capaces de detectar un error básicamente en el momento en que es programado.
* Facilita el cambio: Con un soporte robusto de pruebas unitarias cambios grandes al código pueden ser realizados con mucha más facilidad y tranquilidad, ya que un error ocasionado por el cambio puede ser rápidamente detectado, es decir, el cambio puede ser probado muy rápidamente y asegurar que el código se mantiene funcional.
* Provee documentación: la existencia de las pruebas ofrece una plataforma a otros programadores para entender el funcionamiento individual de cada módulo. La serie de casos analizados en el código de la prueba unitaria ofrece un vistazo a la manera en que se espera que el código se comporte.
* Agilidad: La naturaleza de las pruebas automatizadas es aumentar la productividad, ahorrar tiempo de pruebas manuales y permitir que el código “se pruebe solo”. Las pruebas unitarias abren paso al “Test Driven development” y en general a un desarrollo más rápido de integración continua. Fragmentos de código ya probados son generados más rápida y frecuentemente.

Como se menciona anteriormente beneficios son suficientes para convertir a las pruebas en un procedimiento estándar en muchas ocasiones. Pero como cualquier otra inversión, las pruebas unitarias poseen un costo de oportunidad, o desventajas, que se deben de tener en cuenta, por ejemplo:

* Costo en recursos: La creación de pruebas unitarias puede resultar un proceso costo, tanto en tiempo como en personal. La naturaleza de las pruebas puede ser compleja y cada una debe ser programada manualmente. Y, lógicamente, se requiere de personal con las habilidades y conocimiento suficientes para crear las pruebas unitarias.
* Difícil de implementar en código de legado: La mejor forma de programar pruebas unitarias es durante del desarrollo del código mismo. Un tercero al que se le entrega un código desconocido sin ninguna prueba, e inclusive con pobre documentación (o ninguna), va a tener un reto enorme intentando crear pruebas unitarias para este.
* NO SIRVE EN GUI
* NO ES INFALIBLE

# Frameworks de pruebas unitarias en ¿?

# Frameworks de pruebas unitarias en ¿?

# Ejemplo aplicado en ¿?

# Ejemplo aplicado en ¿?

# Conclusiones

# Bibliografía

<https://methodpoet.com/unit-testing-advantages-and-disadvantages/>

<https://performancelabus.com/unit-testing-importance/#1>