# Fundamentos de Testing

## Mesa de 3 patas

Se refiere a tres roles específicos en el proceso de desarrollo de software, especialmente cuando se trabaja con metodologías ágiles como Scrum o Extreme Programming (XP). Estos roles son:

Desaroolador:

Descripción: La persona o grupo de personas que escribe el código del software. Son responsables de diseñar, codificar y asegurarse de que el software funcione correctamente.

Importancia: El desarrollador es esencial porque es quien transforma los requisitos y especificaciones en un producto de software funcional. Su visión técnica es crucial para identificar posibles dificultades y soluciones durante el proceso de desarrollo.

Tester:

Descripción: El tester se encarga de probar el software para encontrar y reportar defectos o problemas. Su objetivo es asegurarse de que el software cumpla con los requisitos y funcione correctamente en todos los escenarios posibles.

Importancia: El rol del tester es fundamental para asegurar la calidad del software. A través de la identificación de errores o fallos, garantiza que el producto final sea robusto y confiable. Su perspectiva es esencial para visualizar cómo los usuarios finales interactúan con el software y cuáles podrían ser sus problemas o inquietudes.

Product Owner:

Descripción: Este rol representa la voz del cliente o del usuario. Es quien tiene una visión clara de lo que se espera del software y sus funcionalidades. Proporciona requisitos, prioriza características y se asegura de que el producto final cumpla con las expectativas del negocio o del usuario final.

Importancia: El Product Owner es el puente entre el equipo de desarrollo y los stakeholders o interesados. Asegura que el software sea relevante y valioso para los usuarios, y que se alinee con los objetivos comerciales.

## ¿Por qué es importante la colaboración entre estos tres roles?

El concepto de la "mesa de 3 patas" enfatiza que, al igual que una mesa necesita todas sus patas para estar equilibrada y funcional, el proceso de desarrollo de software necesita la contribución activa y equilibrada de estos tres roles para ser exitoso.

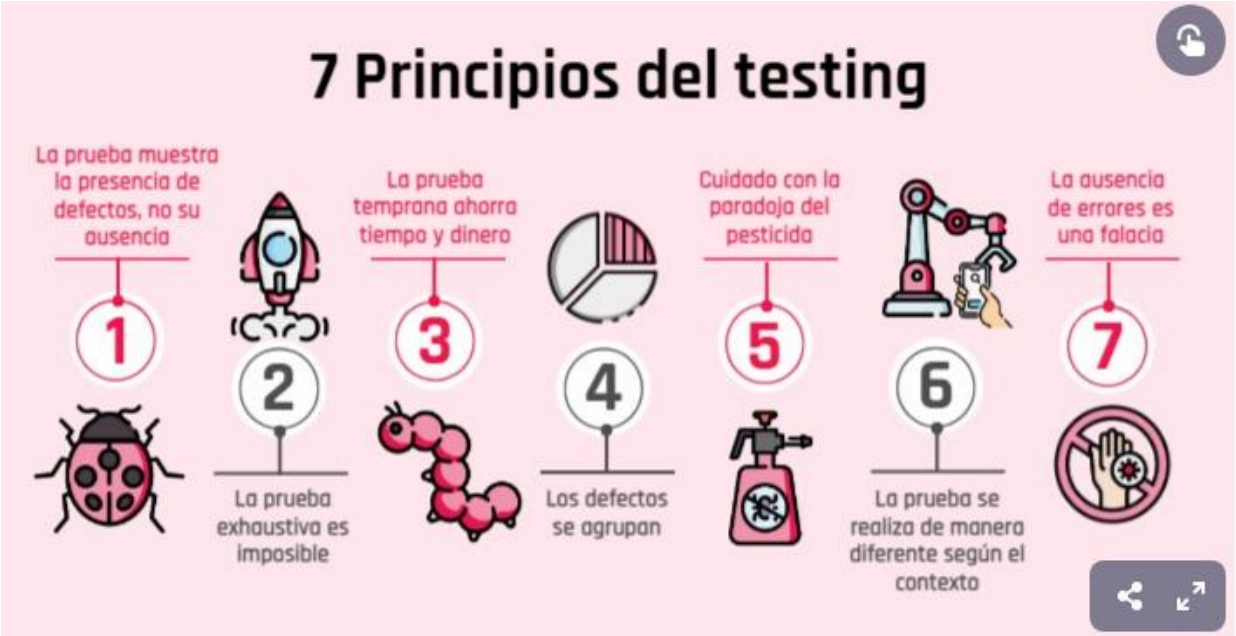
Cuando estos tres roles colaboran estrechamente:

Se garantiza que todas las perspectivas (técnica, de calidad y de negocio) sean consideradas.

Se aumentan las posibilidades de entregar un producto que no solo funcione correctamente, sino que también satisfaga las necesidades reales de los usuarios y aporte valor al negocio.

Se minimizan los malentendidos y las brechas de comunicación, lo que puede llevar a un desarrollo más rápido y eficiente.

## 7 Principio de Testing



Pruebas no demuestran la ausencia de defectos: Si bien las pruebas pueden encontrar defectos, nunca se puede probar que un software esté completamente libre de errores. Lo que se busca es reducir el riesgo de defectos críticos.

Testing temprano: Es fundamental comenzar las pruebas lo más temprano posible en el ciclo de vida del desarrollo. Cuanto antes se detecten los defectos, más barato y fácil será solucionarlos.

Testing exhaustivo es imposible: No se puede probar todas las combinaciones de entradas, estados y escenarios en un software. En lugar de eso, es crucial priorizar y enfocarse en los escenarios de prueba más relevantes y críticos.

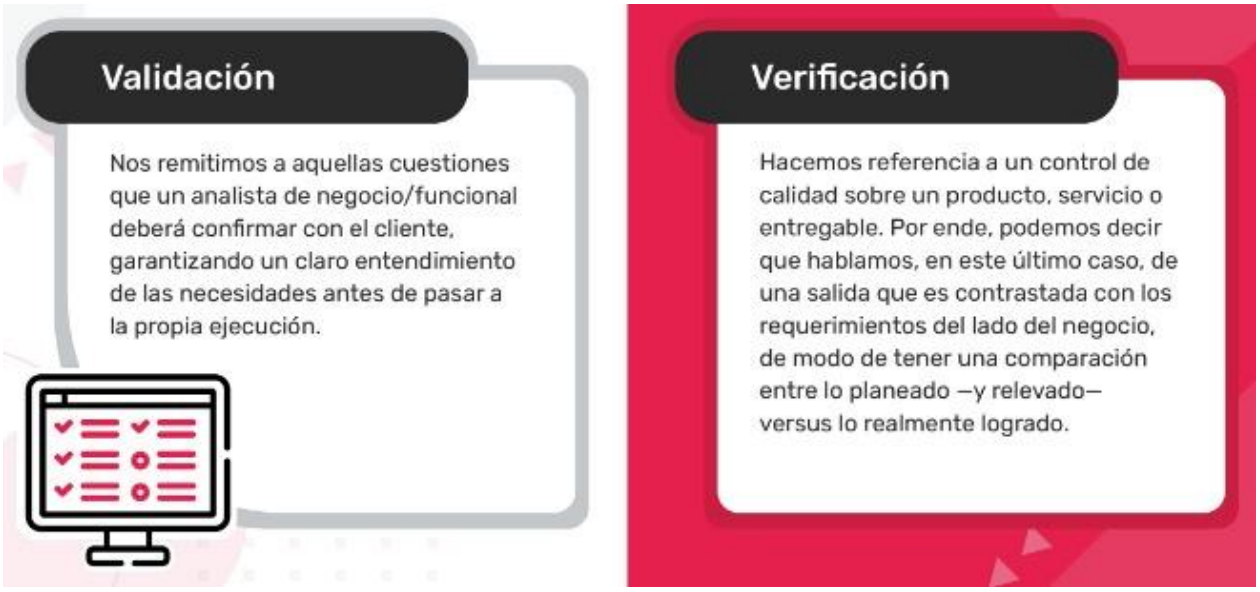
Los defectos tienden a agruparse: En muchas ocasiones, un pequeño segmento del código puede ser responsable de la mayoría de los defectos. Identificar estas áreas "calientes" puede hacer que el testing sea más efectivo.

El paradigma de la paradoja del pesticida: Si se usan las mismas pruebas una y otra vez, con el tiempo estas pruebas dejarán de encontrar nuevos defectos. Es esencial revisar y actualizar regularmente las pruebas para adaptarse a los cambios y evoluciones del software.

Dependencia del contexto: Lo que funciona en una situación o para un producto en particular puede no ser adecuado para otro. Las pruebas deben ser diseñadas considerando el contexto específico del software bajo examen.

La falacia de la ausencia de errores: Sólo porque un software pase todas las pruebas no significa que esté listo para el lanzamiento. Las pruebas solo pueden evaluar lo que se ha definido como criterio de éxito, y no pueden evaluar la calidad global desde la perspectiva del usuario final.

# La diferencia entre verificación y validación




Verificación y validación son dos conceptos fundamentales en el proceso de aseguramiento de la calidad del software, pero tienen objetivos y enfoques distintos.

Verificación: Se refiere al proceso de comprobar si el producto está siendo desarrollado correctamente. Es decir, si estamos construyendo el producto de la manera que se especificó. Se puede considerar como "estamos construyendo el producto correctamente".

Validación: Se refiere al proceso de comprobar si el producto que hemos desarrollado es el correcto. Es decir, si cumple con las necesidades y requerimientos del usuario o cliente. Se puede considerar como "estamos construyendo el producto correcto".

Ejemplo:

Imagina que estás construyendo una aplicación para reservar entradas de cine online.

Verificación: Aquí te fijarías en aspectos como:

¿La página de reserva carga en menos de 3 segundos?

¿El sistema procesa correctamente las transacciones y reserva el asiento elegido?

¿Los datos ingresados por los usuarios se almacenan de manera segura y conforme a las especificaciones técnicas?

En otras palabras, en la verificación estás asegurando que la aplicación funcione técnica y específicamente como se había planeado y documentado.

Validación: En este paso te preguntarías:

¿Los usuarios encuentran fácil usar la aplicación?

¿La aplicación satisface las necesidades de los usuarios al reservar entradas de cine?

¿Hay características esenciales que los usuarios esperaban y que no están presentes?

En la validación, el foco está en asegurar que la aplicación cumpla con las necesidades y expectativas reales de los usuarios y stakeholders.

En resumen, mientras que la verificación se ocupa de "hacer las cosas correctamente" en términos de especificaciones técnicas y diseño, la validación se asegura de que estás "haciendo las cosas correctas" desde la perspectiva de las necesidades y expectativas del usuario.