INTRODUCCIÓN AL TESTING

1. Calidad

¿De qué hablamos cuando hablamos de calidad en testing? Del grado de satisfacción del cliente. Implica cumplir con sus expectativas y satisfacerlas, sin excedernos en tiempo y presupuesto.

Entonces, si cumplimos sus requerimientos por encima de sus expectativas, ¿hablamos de calidad? Bajo este escenario, no sería así, dado que estamos destinando más recursos de los realmente necesarios, disminuyendo la calidad.

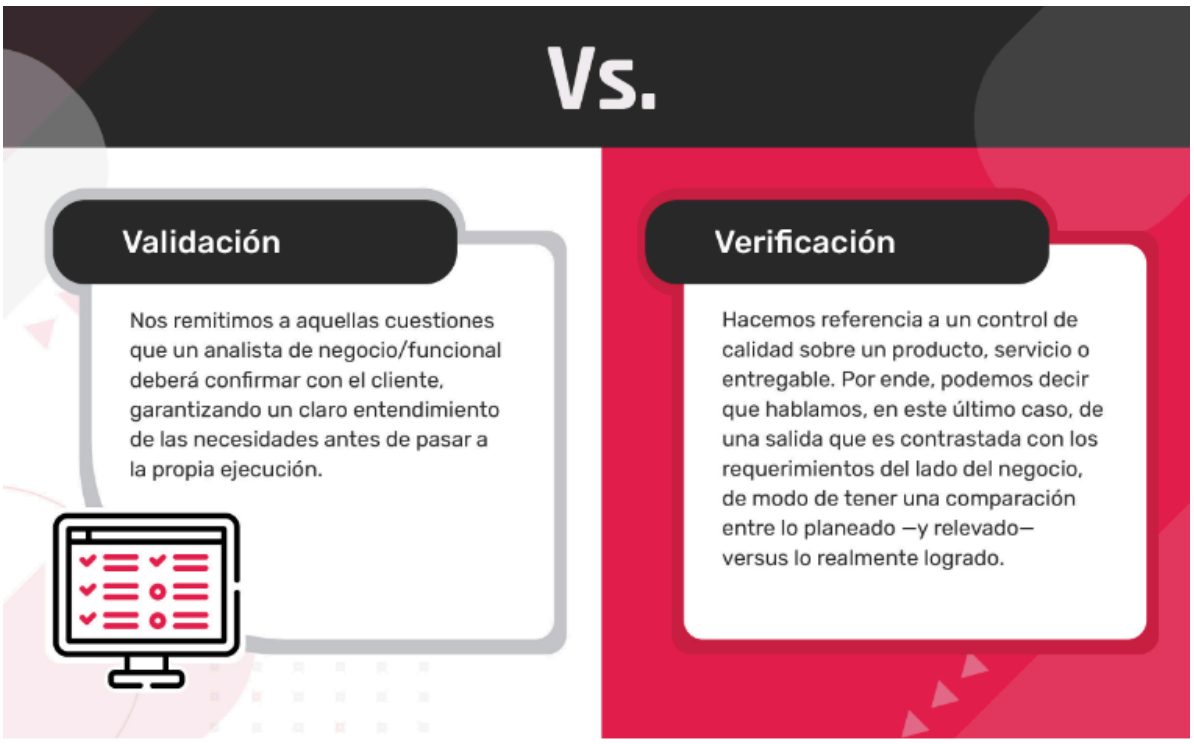
2. Evolución Histórica de la Calidad

1950 - 1960	Después de la Segunda Guerra Mundial, los importantes avances en el desarrollo de software tuvieron lugar en EE. UU. y se generaron en el ámbito de la industria militar. En aquella etapa de la evolución del software, las aplicaciones eran desarrolladas para un hardware dedicado, sistemas que contaban al software como una de sus partes. La calidad asociada a estos sistemas se lograba con pruebas exhaustivas una vez terminado de construir.

1960 - 1970	Con los avances en el hardware y la aparición de lenguajes de alto nivel, se estableció una nueva tendencia en el desarrollo: se comenzaron a producir sistemas no militares e independientes del hardware. Los avances estuvieron orientados a producir sistemas de propósito general. A partir de los inconvenientes de sobrepresupuesto y tiempo adicional necesarios en la terminación del proyecto de desarrollo del sistema operativo de IBM, se generó una alerta en el sentido de la necesidad de contar con métodos de desarrollo que garantizaran la calidad de los productos de software.
1980 - 1990	En la década de 1990, el crecimiento de los sistemas se acentuó, con protagonistas como Microsoft —ya convertida en líder mundial— Netscape y Oracle, entre otros. Además, se consolidaron las metodologías de desarrollo de tipo iterativas, las cuales van suplantando a las conocidas como cascada. Aparecieron algunas metodologías llamadas ágiles y el concepto de integración continua y también se sigue trabajando en IVV (Independent Verification Validation). Estas formas de trabajo tienen una fuerte influencia en la calidad del software.
1990 - 2000	El escenario establecido para el desarrollo de software está determinado por un hardware cada vez más poderoso, software de última generación, modelos de desarrollo y metodologías ágiles. Esta velocidad creciente impuesta por el mercado de productos de software tiene un impacto importantísimo en la calidad de los productos y servicios ofrecidos. Es de notar que estos cambios en la evolución de esta industria hicieron que la preparación de los desarrolladores de hoy día sea muy distinta a la que tenían aquellos programadores de sistemas integrados a un hardware dedicado y con requerimientos estables de los años cincuenta.

2000 - Actualidad	En esta época se afianza la integración entre la ingeniería del software y la ingeniería de sistemas destacándose el papel de los requisitos no funcionales y, sobre todo, de la seguridad; la importancia de la "ciencia, gestión e ingeniería de los servicios" que requiere un enfoque interdisciplinar —informática, marketing, gestión empresarial, derecho, entre otros— a la hora de abordar el diseño de los servicios; la necesidad de adaptar los métodos de desarrollo de software para trabajar en un "mundo abierto" —teniendo en cuenta la inteligencia ambiental, las aplicaciones conscientes del contexto, y la computación pervasiva)—; los *sistemas de sistemas intensivos en software* (SISOS) con decenas de millones de líneas de código, decenas de interfaces externas, proveedores "competitivos", jerarquías complejas, entre otros. También estamos viendo ya la implantación de la *ingeniería del software continua*, y su correspondiente tecnología y "filosofía" *DevOps*, que logran reducir el tiempo entre que se compromete un cambio en el sistema y se implementa en producción; lo que requiere un cambio cultural para aceptar la responsabilidad compartida —entre desarrollo y operación— de entregar software de alta calidad al usuario final.

3. Validacion y Verificacion



4. 7 Principios del Testing

La prueba muestra la presencia de defectos, no su ausencia
No puede probar que no hay defectos. Reduce la probabilidad de que queden defectos no descubiertos en el software, pero, incluso si no se encuentran, el proceso de prueba no es una demostración de corrección.

La prueba exhaustiva es imposible
No es posible probar todo —todas las combinaciones de entradas y precondiciones—, excepto en casos triviales. En lugar de intentar realizar pruebas exhaustivas se deberían utilizar el análisis de riesgos, las técnicas de prueba y las prioridades para centrar los esfuerzos de prueba.

La prueba temprana ahorra tiempo y dinero

Para detectar defectos de forma temprana, las actividades de testing, tanto estáticas como dinámicas, deben iniciarse lo antes posible en el ciclo de vida de desarrollo de software para ayudar a reducir o eliminar cambios costosos.

Los defectos se agrupan

En general, un pequeño número de módulos contiene la mayoría de los defectos descubiertos durante la prueba previa al lanzamiento o es responsable de la mayoría de los fallos operativos

Cuidado con la prueba del pesticida

Si las mismas pruebas se repiten una y otra vez, eventualmente estas pruebas ya no encontrarán ningún defecto nuevo. Para detectarlo, es posible que sea necesario cambiar las pruebas y los datos de prueba existentes.

La prueba se realiza de manera diferente según el contexto

Por ejemplo, el software de control industrial de seguridad crítica se prueba de forma diferente a una aplicación móvil de comercio electrónico.

La ausencia de errores es una falacia

El éxito de un sistema no solo depende de encontrar errores y corregirlos hasta que desaparezcan ya que puede no haber errores, pero sí otros problemas. Existen otras variables a tener en cuenta al momento de medir el éxito.