

TALLER DE TESTING Y CALIDAD DE SOFTWARE

Semana 1





ESCUELA DE CONSTRUCCIÓN E INGENIERIA

Director: Marcelo Lucero

ELABORACIÓN

Experto disciplinar: Aída Villamar Gallardo.

Diseño instruccional: Carla Silva Alvarado.

VALIDACIÓN

Experto disciplinar: Andrés del Alcázar

Jefa de Diseño Instruccional: Alejandra San Juan Reyes.

EQUIPO DE DESARROLLO

AIEP

AÑO

2021



Tabla de contenidos

Aprendizaje esperado de la semana	4
Introducción	4
1. Conceptos asociados a pruebas de sistema para proceso de certificación y testeo de proyecto de software	5
1.1. Conceptos	5
1.2. Objetivos de la prueba	6
1.3. Pruebas de sistema, tipos de pruebas de sistema	8
2. Verificación y validación de software, considerando análisis de casos	10
2.1. Verificación y validación.....	10
3. Proceso de testing de software o de sistema informático, considerando mejora de calidad.	14
3.1. Las pruebas y el aseguramiento de calidad	14
4. Software testeado y software no testeado, según calidad de producto final.	16
4.1. Diferencias entre software testeado y software no sometido a pruebas de software.....	16
Ideas clave	19
Conclusiones	21
Referencias bibliográficas	22



Aprendizaje esperado de la semana

Diferencian conceptos asociados a calidad de sistemas en proyectos de desarrollo de software.

Introducción

- ¿Cómo saber si un software es de calidad?
- ¿Cuál es el proceso de pruebas de software?
- ¿Qué tipos de pruebas de software existen?
- ¿Qué es validar y qué es verificar software?
- ¿Cuándo hablamos de probar el software cuál es la diferencia entre verificación y validación?

Dentro de nuestro trabajo, el desarrollo de software, nos encontramos con ciertos elementos que, si bien es cierto, no están relacionados directamente con el proceso de programar, sí juegan un rol muy importante en el resultado que vamos a obtener, ya que un software de calidad no es producto del azar.

Obtenemos un software de calidad porque hemos incorporado el concepto de calidad en el proceso de desarrollo y nos hemos asegurado de que nuestro trabajo se ajusta a lo que el cliente requiere y, además, entrega los resultados esperados.



1. Conceptos asociados a pruebas de sistema para proceso de certificación y testeo de proyecto de software

1.1. Conceptos

Las **pruebas de sistema** son una fase de prueba de investigación, un conjunto de actividades dentro del desarrollo de un sistema, por medio de los cuales se asegura que el módulo o componente interactúa con otros componentes o módulos como fue diseñado. Se comprueba de una manera global la integración del sistema de información, se verifica el adecuado funcionamiento de las interfaces y la comunicación con el resto de los sistemas de información y diversos subsistemas que lo componen, disponemos de diversas pruebas que pueden ser implementadas en las distintas etapas del proceso de desarrollo según sea necesario.

De acuerdo al sitio Pragma.com.co (s. f.),

Cuando se habla de certificación, se abarcan distintos tipos de pruebas como funcionales y no funcionales. **Entre las funcionales se encuentran las más importantes: las manuales que se complementan con las automatizadas**, ambas tienen como objetivo ir probando los diferentes escenarios que podría tener un usuario final, y así, localizar los errores o vulnerabilidades que tiene un sistema en su vida de desarrollo.

Por otra parte, de acuerdo a Villaumbrales (2019),



El **testing de software, o software QA**, es una disciplina en la ingeniería de software que permite tener procesos de ejecución de un programa o aplicación y una metodología de trabajo con el objetivo de localizar errores de software. También puede describirse como el proceso de validación y verificación de un programa de software o una aplicación. Algo importante que debemos considerar durante el proceso de desarrollo de un sistema, es **el testeo que se debe realizar de forma paralela**, es decir, que en cada una de las etapas es necesario incorporar la realización de tareas de testing. Con esto, podremos prevenir situaciones inesperadas en cuanto a funcionalidad y corregir desviaciones en el software antes de que sea enviado a producción.

1.2. Objetivos de la prueba

Las pruebas son un elemento fundamental que nos permite asegurar el correcto funcionamiento del sistema. Dentro de sus objetivos encontramos:

- La detección de defectos en el software.
- Verificar que los componentes se han integrado adecuadamente.
- Verificar que todos los requisitos se han implementado correctamente.
- Identificar y asegurar que antes de la entrega del software al cliente, todos los defectos encontrados han sido corregidos.
- Diseñar casos de prueba que, de forma sistemática, permitan evidenciar diferentes clases de errores; haciéndolo con la menor cantidad de tiempo y esfuerzo.



La finalidad de las pruebas de sistema es poder encontrar errores, definiendo así situaciones en las que encontraremos que está sucediendo algo inesperado o bien no está sucediendo cuando se espera. La palabra que mejor describe las pruebas de sistema es **detección**.



Figura 1. Bug

Fuente: Rodríguez (2011, p. 4)

Para la ejecución de las pruebas, se deben realizar ciertas actividades antes y después de la ejecución. Algunas de las actividades son las siguientes:

- Se debe realizar planificación y control.
- Dependiendo de las características de lo que queremos testear, seleccionar las condiciones de las pruebas.
- Es importante diseñar y ejecutar casos de pruebas, generalmente en base a los requerimientos iniciales.
- Comprobar los resultados sobre el sistema probado. Los resultados no deben ser eliminados, ya que servirán para tener un registro y, además, para que los analicemos y podamos entregar recomendaciones de mejora o de optimización en base a ellos, lo que se debe realizar sobre el sistema que se está testeando.



- Una vez terminada la fase de prueba, se deben finalizar o complementar las actividades de cierre.

La revisión de los documentos, del código fuente, es decir, todo lo relacionado con un análisis estático, está incluido dentro del proceso de pruebas.

1.3. Pruebas de sistema, tipos de pruebas de sistema

Objetivos de las pruebas de sistema: confirmar la adecuada navegación por el sistema, tanto en los ingresos, como en el tratamiento y recuperación de los datos.

Se definirá qué pruebas de sistema confirmarán que la aplicación conseguirá sus objetivos de negocio, dentro de las pruebas están: usabilidad, carga, rendimiento, entre otras.

La prueba de sistema incluye Fuente: (Montecinos, 2020)

- **Pruebas funcionales:** dirigidas a asegurar que el sistema de información realiza correctamente todas las funciones que se han detallado en las especificaciones dadas por el usuario del sistema. Fuente: (Montecinos, 2020)
- **Pruebas de comunicaciones:** determinan que las interfaces entre los componentes del sistema funcionan adecuadamente, tanto a través de dispositivos remotos, como locales. Asimismo, se han de probar las interfaces hombre/máquina. Fuente: (Montecinos, 2020)
- **Pruebas de rendimiento:** determinan que los tiempos de respuesta están dentro de los intervalos establecidos en las especificaciones del sistema. Fuente: (Montecinos, 2020)
- **Pruebas de volumen:** examinan el funcionamiento del sistema cuando está trabajando con grandes volúmenes de datos, simulando las cargas de trabajo esperadas. Fuente: (Montecinos, 2020)



- **Pruebas de sobrecarga:** comprueban el funcionamiento del sistema en el umbral límite de los recursos, sometiéndole a cargas masivas. El objetivo es establecer los puntos extremos en los cuales el sistema empieza a operar por debajo de los requisitos establecidos. Fuente: (Montecinos, 2020)
- **Pruebas de disponibilidad de datos:** demuestran que el sistema puede recuperarse ante fallos, tanto de equipo físico como lógico, sin comprometer la integridad de los datos. Fuente: (Montecinos, 2020)
- **Pruebas de facilidad de uso:** comprueban la adaptabilidad del sistema a las necesidades de los usuarios, tanto para asegurar que se acomoda a su modo habitual de trabajo, como para determinar las facilidades que aporta al introducir datos en el sistema y obtener los resultados. Fuente: (Montecinos, 2020)
- **Pruebas de operación:** verifican la correcta implementación de los procedimientos de operación, incluyendo la planificación y control de trabajos, arranque y re arranque del sistema, etc. Fuente: (Montecinos, 2020)
- **Pruebas de entorno:** verifican las interacciones del sistema con otros sistemas dentro del mismo entorno. Fuente: (Montecinos, 2020)
- **Pruebas de seguridad:** verifican los mecanismos de control de acceso al sistema para evitar alteraciones indebidas en los datos. Fuente: (Montecinos, 2020)

Si consideramos sistemas web, vemos que lo mínimo que se debe realizar en cuanto a pruebas de sistema son las siguientes:

- I. Pruebas de humo
- II. Pruebas de usabilidad



III. Pruebas de performance

IV. Pruebas de funcionalidad

2. Verificación y validación de software, considerando análisis de casos.

Pasaremos a revisar estos dos conceptos muy importantes dentro de nuestro proceso, **verificación** y **validación**, los que muchas veces son confundidos o se piensa que representan lo mismo.

Veremos que en realidad son diferentes, pero que se complementan en el resultado por medio de la aplicación de ambos, asegurando que el sistema que se está desarrollando se ajusta a sus especificaciones iniciales, pero además está cubriendo las necesidades del cliente.

2.1. Verificación y validación

Los procesos de verificación y validación corresponden a la unión de procesos de comprobación y análisis que garantizan que el software que se desarrolla está de acuerdo con su especificación y que además cumple las necesidades de los clientes.

Proceso de verificación: corresponde al proceso de evaluación de un sistema (o de uno de sus componentes) para determinar si los productos de una fase dada satisfacen las condiciones impuestas al comienzo de dicha fase. Fuente: (Montecinos, 2020)



Proceso de validación: corresponde al proceso de evaluación de un sistema (o de uno de sus componentes) durante o al final del proceso de desarrollo para determinar si satisface los requisitos marcados por el usuario. Fuente: (Montecinos, 2020)

Para cada etapa del desarrollo de software contamos con actividades de V&V asociadas. Es un proceso de ciclo de vida completo. Fuente: (Montecinos, 2020)

Este comienza con las revisiones de los requerimientos, sigue con las revisiones del diseño y las inspecciones del código, y llega hasta la prueba del producto. Fuente: (Montecinos, 2020)

Se consideran revisiones estáticas y revisiones dinámicas dependiendo de la etapa en que se encuentre el proceso de desarrollo en que se apliquen.

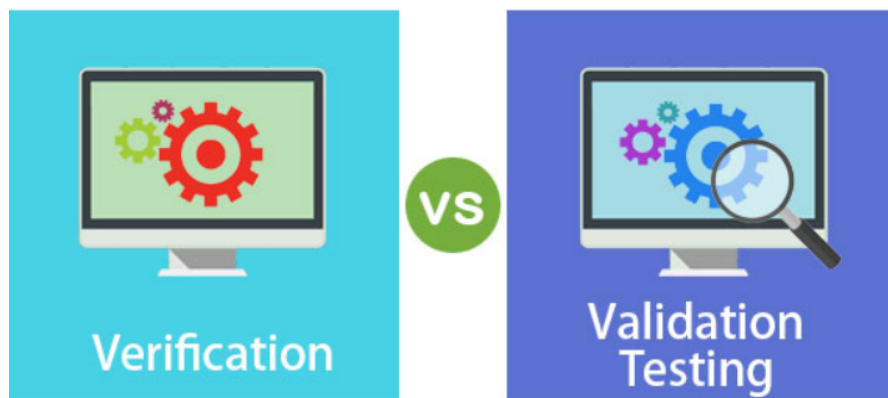


Figura 3. Verification vs Validation Testing, en español: Verificación vs Pruebas de validación

Fuente: (Gomez, s.f.)

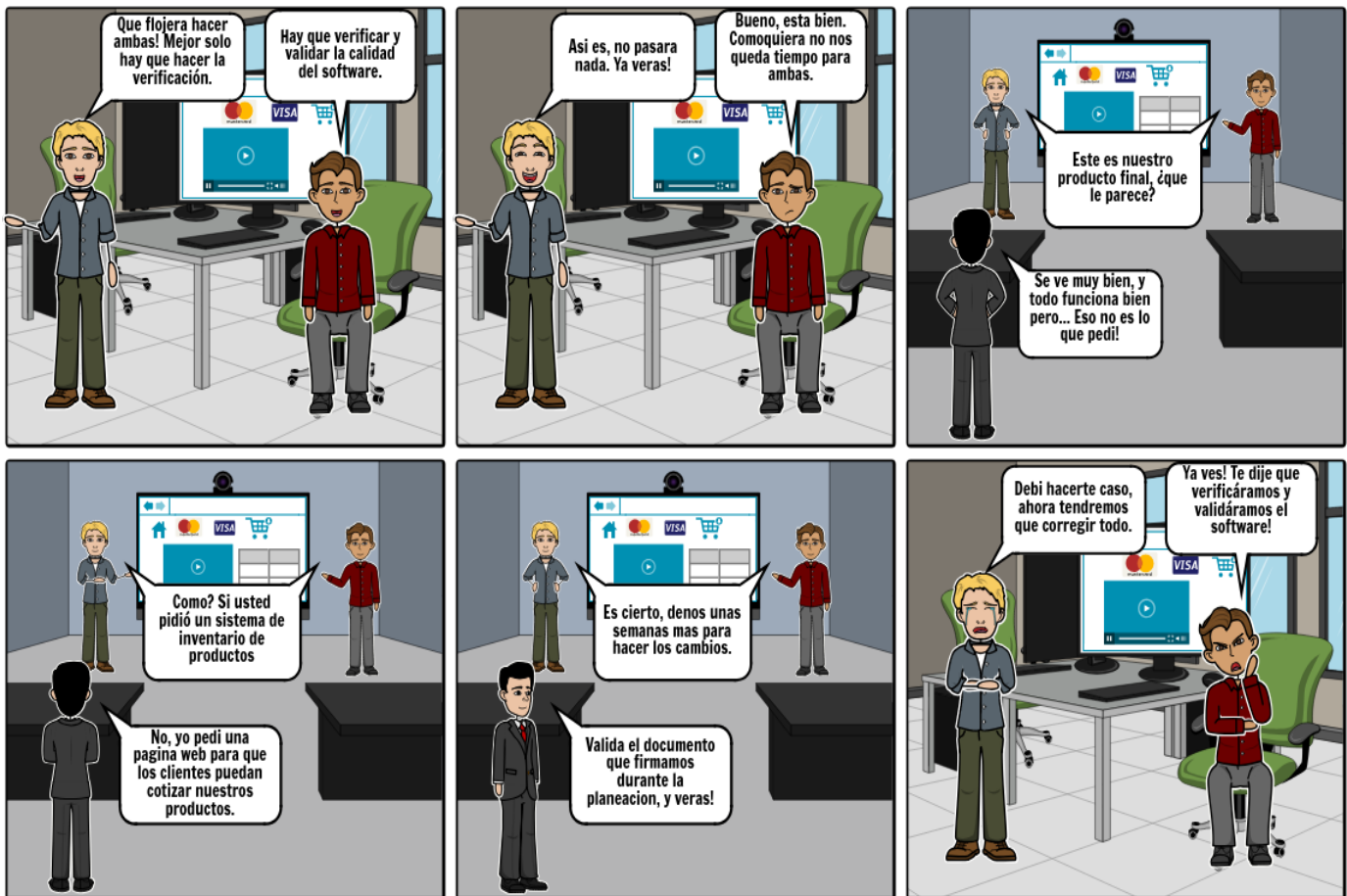
Verificación: ¿estamos construyendo el producto correctamente?, es decir, ¿el software está de acuerdo con su especificación?

Se comprueba que el software cumple los requisitos funcionales y no funcionales de su especificación.

Validación: ¿Estamos construyendo el producto correcto?, es decir, ¿el software cumple las expectativas del cliente?

Comprueba que el software cumple las expectativas que el cliente espera. Fuente: (López, 2009)

Si no se realizan estas dos actividades, puede pasar lo siguiente:



Cree sus propios en Storyboard That

Figura 3. Ejemplo de no realizar ambas actividades

Fuente: (Gómez, s.f.)

Si revisamos y analizamos la situación que se presenta en la imagen anterior, figura 3, podemos apreciar lo importante que es *no dejar de realizar tanto la verificación como la validación del software*.



Que se realicen los procesos de validación y de verificación no implica que el sistema quede libre de defectos, lo que hacemos es reducirlos.

Dentro del proceso de verificación y validación se utilizan dos técnicas de comprobación y análisis de sistemas Fuente: (UNAM, 2018)

Las **inspecciones del software** y los **análisis automatizados** son técnicas de verificación y validación estáticas puesto que no requieren que el sistema se ejecute.

- Las **inspecciones del software** analizan y comprueban las representaciones del sistema como el documento de requerimientos, los diagramas de diseño y el código fuente del programa. Se aplican a todas las etapas del proceso de desarrollo. Las inspecciones se complementan con algún tipo de análisis automático del texto fuente o de los documentos asociados. Fuente: (López, Verificación y Validación - Ingeniería Software, 2009)
- Las **pruebas del software** consisten en contrastar las respuestas de una implementación del software a series de datos de prueba y examinar las respuestas del software y su comportamiento operacional, para comprobar que se desempeñe conforme a lo requerido. Llevar a cabo las pruebas es una técnica dinámica de la verificación y validación ya que requiere disponer de un prototipo ejecutable del sistema. Fuente: (López, Verificación y Validación - Ingeniería Software, 2009)

3. Proceso de testing de software o de sistema informático, considerando mejora de calidad.

De acuerdo a Pragma.com.co (s. f.),

¿Qué es calidad de software? La calidad de software es un complemento necesario en la vida de desarrollo de un sistema porque garantiza que el producto final tenga valor y cumpla con las expectativas esperadas por el usuario final.



Figura 4. Calidad de software

Fuente: Pragma.com.co (s. f.)

La calidad del software es sinónimo de eficiencia, por ello, las empresas dedicadas a desarrollos de tecnologías deben tener expertos conocidos como QA o analistas de pruebas, quienes conforman los equipos de Certificación para analizar plataformas antes de que salgan a producción o se actualicen.

3.1. Las pruebas y el aseguramiento de calidad

Las pruebas en el software avalan la calidad del producto validando que cumpla las especificaciones para las que fue diseñado, cada una con objetivos específicos y cuya responsabilidad recae sobre diferentes roles. Existen diferentes tipos de pruebas, uso e importancia en el proceso de calidad.

Debido a que incluir la realización de pruebas durante el proceso de desarrollo tiene asociado un costo, muchas veces puede llevar a tomar la decisión de no incluirlas, pero no hacerlo también tiene un costo asociado, la dificultad es poder establecer cuál de estos costos es más alto.

Se tienen propuestos los siguientes objetivos:

- Menores costos
- Menores tiempos de desarrollo
- Mayor satisfacción del cliente

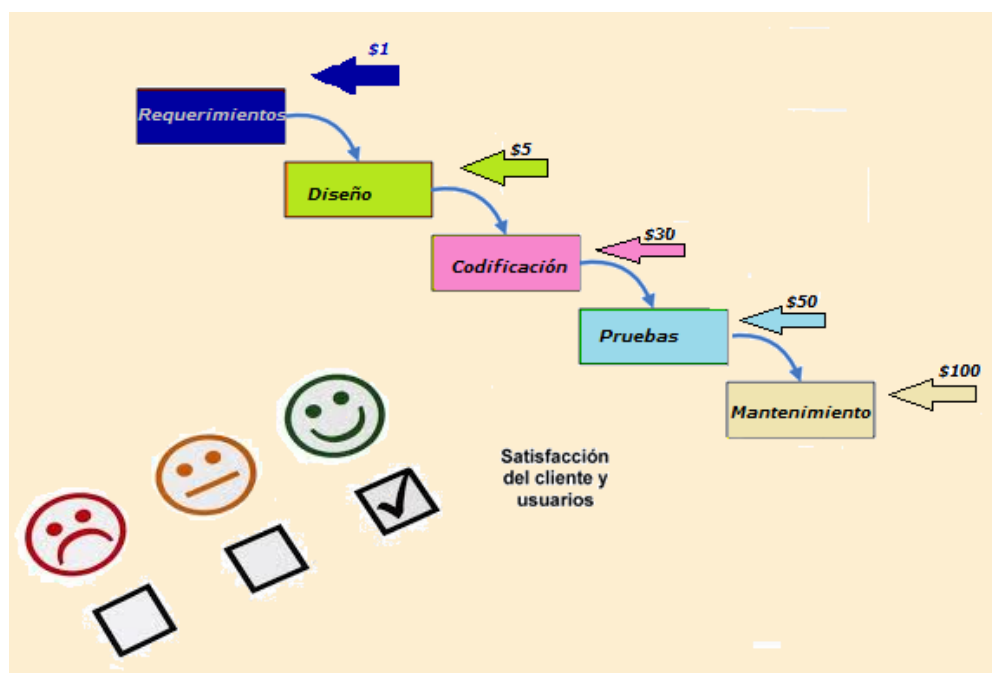



Figura 5. Incremento de los costos de las fallas detectadas en las distintas fases del desarrollo de un sistema



4. Software testeado y software no testeado, según calidad de producto final.

El **testing** de software es una disciplina de la ingeniería de software que brinda procesos, métodos de trabajo y herramientas gracias a las cuales es posible detectar defectos en el software alcanzando un proceso de estabilidad de éste.

El testeo no es una actividad que se deba pensar al final del desarrollo del software, va en paralelo al proceso de desarrollo, ya que, al incorporarlo, permite que aquello que se está desarrollando se efectúe de manera correcta, es decir, vaya de acuerdo con lo que necesita el usuario final. Es tan importante porque es un modo de prevenir e incluso de corregir posibles desviaciones del software antes de que sea entregado a operación. Fuente: (Montecinos, 2020)

Una idea equivocada que se tenía con respecto al testing es que se debía realizar al final, cuando el software estaba codificado, previamente a la entrega para operación. Hoy en día, en cambio, el testing de software se debe incorporar desde el inicio del proceso.

4.1. Diferencias entre software testeado y software no sometido a pruebas de software

Como se indica en el sitio web Yunbitsoftware.com (2016), con el software testeado se evita que en un proyecto de desarrollo de software puedan aparecer errores en cualquiera de las etapas del ciclo de vida.

Con el software testeado se pueden llegar a descubrir incluso errores que permanecerían sin ser descubiertos normalmente.

Tanto en un software testeado como un software no testeado, hay una gran probabilidad de que el código final tenga errores de requerimientos, como de diseño o de funcionalidad.

Para identificar estos problemas antes de que ocurran en un entorno crítico, es necesario realizar pruebas de software, una parte muy importante del proceso, pero también muy costosa.

Fuente: (TELLO, 2011)

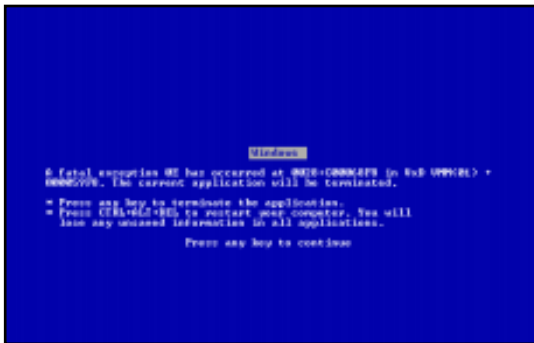


Figura 6. Error – bsod de Windows

Fuente: Rodríguez (2011, p. 5)

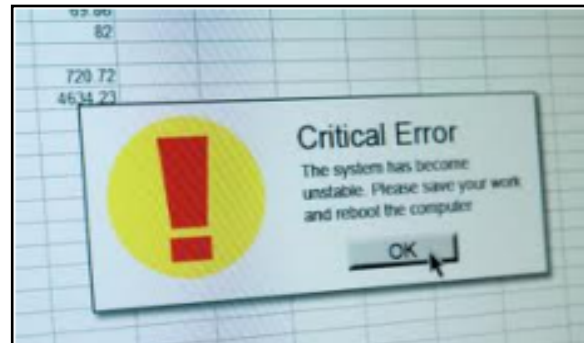


Figura 7. Error

Fuente: Rodríguez (2011, p. 5)

De acuerdo a Rodríguez (2011),

Las pruebas de software pueden llegar a significar entre el 30% y 50% del costo total del desarrollo. No obstante, hay que tener en consideración que los costos ocasionados por fallas en un software en operación pueden llegar a ser mucho más altos y ser considerados catastróficos. [...]

Algunos de los peores errores de la historia según, (TELLO, 2011)

- Se colapsa el aeropuerto de Los Ángeles (2007)

Más de 17 mil personas se quedaron en tierra debido a un problema de software que generó conflictos en una tarjeta de red que bloqueó toda la red informática.



gura 8: Aeropuerto

Fuente: Rodríguez (2011, p. 7)



gura 9: Cancelados

Fuente: Rodríguez (2011, p. 7)

- El lanzamiento comercial y la producción del Airbus A380 se retrasa más de un año (2006)

Diferencias entre versiones de las herramientas CAD (Computer Aided Design) usadas en las fábricas de Hamburgo y Toulouse provocaron un problema en el cableado (530km de cables)



Figura 11: Airbus A380

Fuente: Rodríguez (2011, p. 8)



Figura 12: Cableado

Fuente: Rodríguez (2011, p. 8)

- Sobredosis radiológica en el Instituto Nacional del Cáncer de Panamá (2000)

Errores en los procedimientos y un fallo de software causan que se apliquen dosis erróneas de radiación. 8 personas murieron y 20 tuvieron

problemas de salud graves. Los médicos responsables del hecho fueron acusados de asesinato



Figura 13. Pabellón radiológico

Fuente: Rodríguez (2011, p. 9)

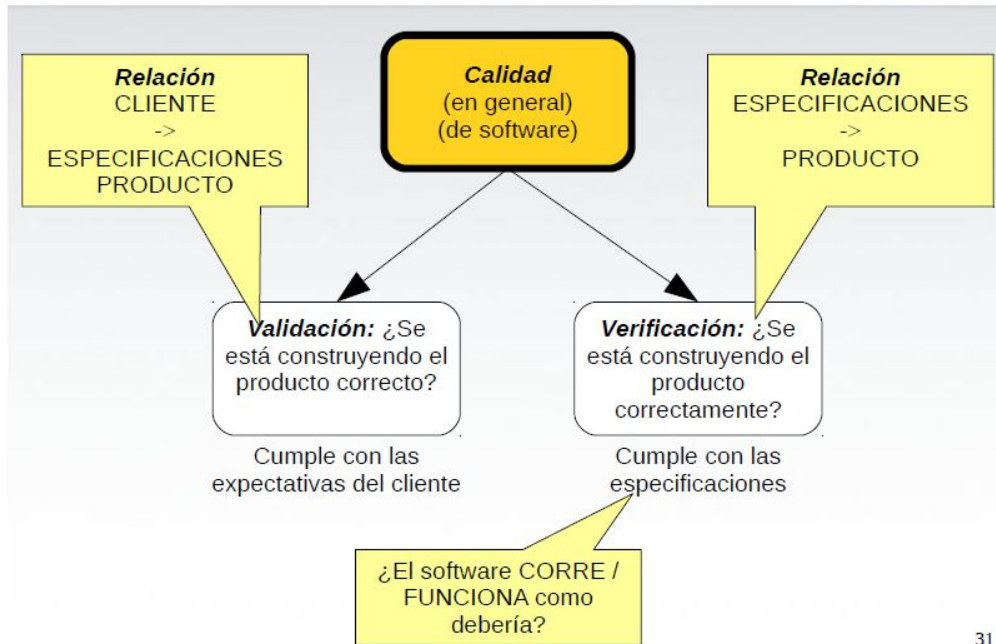


Figura 14: Muerte de personas

Fuente: Rodríguez (2011, p. 9)

Ideas clave

Verificación y Validación



31

Figura 15. El proceso de verificación y validación

Fuente: (Villamar, 2021)

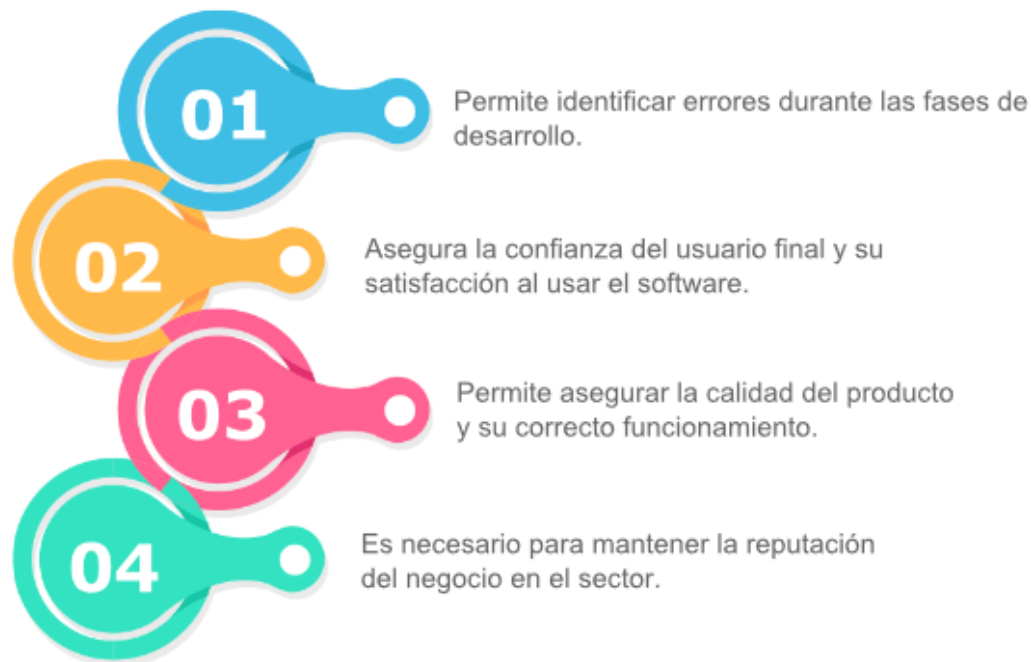


Figura 16. ¿Qué es el Testing de Software y por qué es importante?

Greensqa.com. <https://greensqa.com/que-es-el-testing-de-software-y-por-que-es-importante/>

Conclusiones

Se tiende a pensar que el proceso de pruebas se dedica solamente a la ejecución del software, pero hemos visto que la ejecución de pruebas es una parte del proceso y no representa la totalidad de las actividades que comprende.

Si omitimos la incorporación de algo tan importante como es el testeo en nuestro desarrollo, estamos afectando de manera directa la calidad de nuestro trabajo. Es por ello que, desde el principio, debe



ser considerado y aplicado en cada una de las etapas de nuestros proyectos.

Referencias bibliográficas

Pragma.com.co. (s. f.). Todo lo que debes saber sobre certificación de software. Recuperado de

<https://www.pragma.com.co/academia/conceptos/todo-lo-que-debes-saber-sobre-certificacion-de-software>

Rodríguez, E. (2011). Importancia de las pruebas de software. [Presentación]. Recuperado de

<https://www.tamps.cinvestav.mx/~ertello/swe/swTestingTecZacatecas.pdf>

Testinglat.com. (s. f.). Fundamentos de testing. Recuperado de <https://testinglat.com/definicion-de-testing/>

Villaumbrales, I. (2019, 26 de junio). Testing, la importancia sobre la fase del testeo de software. [Entrada de blog]. Recuperado de <https://www.hiberus.com/crecemos-contigo/testing-fase-de-testeo-de-software/>

Yunbitsoftware.com. (2016, 9 de febrero). Importancia de las pruebas de desarrollo de software. [Entrada de blog]. Recuperado de <https://www.yunbitsoftware.com/blog/2016/09/02/importancia-de-las-pruebas-en-desarrollo-de-software/>