DuocUC® INFORMÁTICA Y TELECOMUNICACIONES

Obtención de evidencias y registro de resultados

Calidad de Software - CSY4111

CONTENIDO

DuocUC®

Refrescando conocimiento

DuocUC®

REFRESCANDO CONOCIMIENTOS

Un caso de prueba es un conjunto de condiciones bajo las cuales un analista determinará si una aplicación, es parcial o completamente satisfactoria

Se pueden crear muchos casos de prueba para determinar que un requisito es completamente satisfactorio.

Debe haber al menos un caso de prueba para cada requisito.

DuocUC®

REFRESCANDO CONOCIMIENTOS

Estructura de un caso de prueba

Nombre del proyecto: Nombre de la aplicación
Ambiente de pruebas: Versión web o de escritorio
Autor del caso de prueba: Nombre del analista que diseña el caso de prueba.
ID: caso de prueba: Número que identifica el caso de prueba
ID Historia de usuario: Número con el que identifica la historia de usuario que hace referencia al caso.
Propósito: Cual es la finalidad del caso de prueba.
#: Nro. De la acción.

Los casos de pruebas buscan verificar que el software haga lo que se supone que debe hacer.

DuocUC®

01

CICLO DE VIDA DE LAS PRUEBAS DE SOFTWARE (STLC)

DuocUC®

CICLO DE VIDA DE LAS PRUEBAS DE SOFTWARE (STLC)

El ciclo de vida de las pruebas de software (STLC)

Es un conjunto de acciones que se realizan de forma coherente, sistemática y planificada con el fin de averiguar si un producto de software funciona correctamente y si existen opciones para mejorarlo.

En general, el procedimiento de prueba consta de 8 fases, que comprenden 3 tareas: preparación para la ejecución, ejecución en sí, conclusión.

La parte de la conclusión es trascendental, ya que los conceptos de pruebas de software prevén que cada prueba ejecutada tiene que ayudar a resolver futuros errores.

CICLO DE VIDA DE LAS PRUEBAS DE SOFTWARE (STLC)

Fase de requisitos

Fase de planificación

Fase de análisis

Fase de diseño

Fase de implementación

Fase de ejecución

Fase de conclusión

Fase de cierre

DuocUC®

CICLO DE VIDA DE LAS PRUEBAS DE SOFTWARE (STLC)

Fase de requisitos

Es el primer paso de todo el ciclo de vida de la garantía de calidad, en el que un equipo de pruebas define cómo debe comportarse exactamente el producto final de acuerdo con los requisitos de las partes interesadas.

La recopilación de esta información ayuda a optimizar el proceso de trabajo en el marco del proyecto desde el principio.

En esta fase, el equipo define los tipos de pruebas que se van a realizar y el entorno en el que se espera que se lleven a cabo.

CICLO DE VIDA DE LAS PRUEBAS DE SOFTWARE (STLC)

Fase de planificación

Incluye la definición de los requisitos para las pruebas, la estimación de los riesgos que pueden surgir en los procedimientos adecuados, la elección de la estrategia de pruebas, la programación y la definición de las secuencias, y la creación del plan de pruebas.

Aquí también se determinan los objetivos y el alcance del proyecto para calcular el trabajo y el coste totales del mismo.

Criterios de salida: Estrategia de pruebas, plan de pruebas, documento de estimación de trabajo de pruebas.

CICLO DE VIDA DE LAS PRUEBAS DE SOFTWARE (STLC)

Fase de análisis

Este paso se realiza para definir con exactitud los procedimientos que deben llevarse a cabo en todas las fases posteriores de las pruebas de software.

Para hacer una lista exhaustiva y tener en cuenta todas las condiciones necesarias que hay que proporcionar, el equipo tiene que estudiar a fondo todos los aspectos del proyecto y considerar todos los detalles con precisión.

Este proceso es muy importante ya que tiene un impacto directo en el éxito del cumplimiento de las siguientes fases.

Criterios de salida: Estrategia de pruebas, plan de pruebas.

CICLO DE VIDA DE LAS PRUEBAS DE SOFTWARE (STLC)

Fase de diseño

En este punto el equipo prepara todos los casos de prueba necesarios. Un ejemplo sencillo sería «probar el inicio de sesión del usuario en un sistema introduciendo datos válidos/no válidos».

En función de los requisitos de las partes interesadas, se crean los casos adecuados.

Las pruebas pueden diseñarse tanto para las pruebas de caja negra como para las de caja blanca.

CICLO DE VIDA DE LAS PRUEBAS DE SOFTWARE (STLC)

Fase de diseño

Es importante crear una lista suficiente de casos que cubra todos los aspectos necesarios.

Si procede, también se escriben en esta fase las secuencias de comandos para las pruebas automatizadas.

Criterios de entrada: Plan de pruebas, casos de prueba de humo, datos de prueba;

Criterios de salida: Entorno de prueba, resultados de prueba de humo.

CICLO DE VIDA DE LAS PRUEBAS DE SOFTWARE (STLC)

Fase de implementación

Este es el último paso antes de la etapa de ejecución. Es necesario asegurarse de que todos los requisitos previos necesarios están ajustados. Analista QA tiene que comprobar si el entorno está preparado y tiene una configuración correcta de acuerdo con las directrices escritas para este proyecto.

Criterios de entrada: Plan de pruebas, casos de prueba de humo, datos de prueba;

Criterios de salida: Entorno de prueba, resultados de prueba de humo.

CICLO DE VIDA DE LAS PRUEBAS DE SOFTWARE (STLC)

Fase de ejecución

De acuerdo con el plan de pruebas y los casos de prueba que han sido desarrollados en los pasos anteriores, se ejecutan las pruebas correspondientes.

Cuando se detecta un fallo, el analista o Tester qa lo marca con el estado correspondiente (fallido, bloqueado, no ejecutado, etc.) y asigna un ID para todos ellos.

CICLO DE VIDA DE LAS PRUEBAS DE SOFTWARE (STLC)

Fase de ejecución

Después de arreglar todos los fallos, el equipo vuelve a realizar pruebas para ver si todo está bien, ya que arreglar un fallo en un sitio puede provocar que aparezca otro en el otro lado.

Cuando la construcción está lista, es el momento de la fase de conclusión.

Criterios de entrada: Plan de pruebas, casos de prueba, datos de prueba, entorno de prueba;

Criterios de salida: Informe de ejecución de casos de prueba, informe de defectos.

CICLO DE VIDA DE LAS PRUEBAS DE SOFTWARE (STLC)

Fase de conclusión

Como su nombre indica, en esta etapa previa a la finalización del STLC se documentan todas las pruebas ejecutadas y sus resultados, y se registran los casos fallidos y sus defectos.

Toda esta información se envía al jefe de equipo de forma semanal o mensual. Posteriormente la analiza o revisa y la prepara para el informe de cierre.

Criterios de entrada: Informe de ejecución de casos de prueba, informe de defectos;

Criterios de salida: Informe de cierre preparado.

CICLO DE VIDA DE LAS PRUEBAS DE SOFTWARE (STLC)

Fase de cierre

Cuando todo está hecho, el equipo organiza una reunión en la que se subrayan los momentos clave, se discuten las nuevas formas posibles de evitar fallos similares en el futuro y se preparan métricas para el proyecto (por ejemplo, tiempo total invertido, costes, calidad, cobertura de las pruebas, etc.).

Toda la información útil se registra y documenta, de modo que, si en el futuro se produjera un error parecido, el probador podría solucionarlo rápidamente.

Criterios de entrada: Informe de cierre preparado;

Criterios de salida: Informe de cierre de pruebas, métrica de pruebas.

02

EJECUCIÓN DE PRUEBAS

DuocUC®

EJECUCIÓN DE PRUEBAS

Ejecución de pruebas, recopilación de evidencias y documentación de defectos son actividades clave en el proceso de pruebas de software.

A continuación, se presenta una definición detallada de cada una de ellas:

EJECUCIÓN DE PRUEBAS

Ejecución de pruebas:

La ejecución de pruebas es el proceso de ejecutar los casos de prueba diseñados para evaluar el comportamiento y la calidad del software.

Implica seguir un plan de pruebas predefinido, ejecutando los casos de prueba y **registrando los resultados.**

Durante la ejecución de las pruebas, se verifican las funcionalidades, se evalúa el rendimiento, se comprueba la interoperabilidad y se analiza el cumplimiento de los requisitos establecidos.

EJECUCIÓN DE PRUEBAS

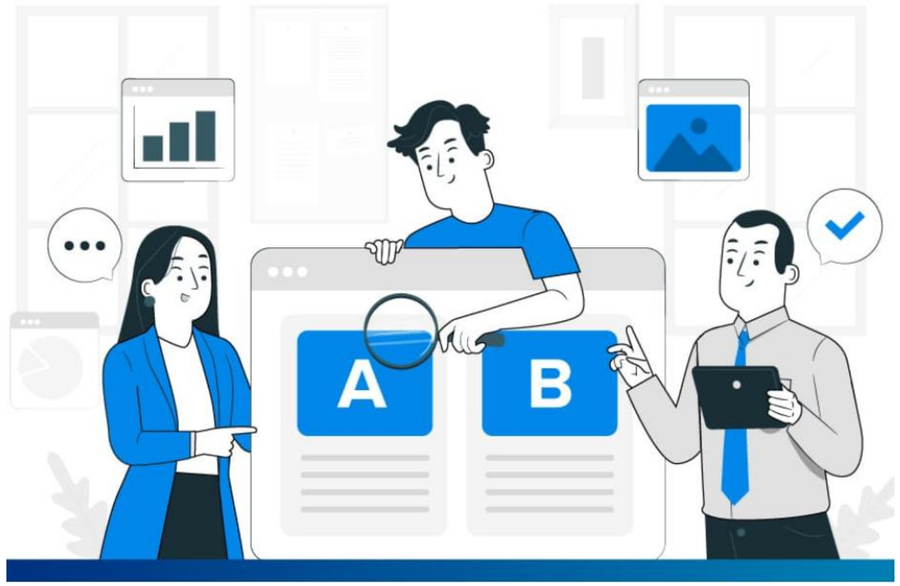
Ejecución de pruebas:

El objetivo principal de la ejecución de pruebas es identificar defectos o problemas en el software y garantizar que funcione correctamente antes de su lanzamiento o implementación.

EJECUCIÓN DE PRUEBAS

Recopilación de evidencias:

La recopilación de evidencias implica capturar y documentar los resultados y datos relevantes durante la ejecución de las pruebas.



Esto puede incluir capturas de pantalla, registros de eventos, mensajes de error, informes de resultados, registros o trazas de logs, entre otros.

Es de vital importancia, documentar en el orden de ejecución del plan de pruebas

DuocUC®

EJECUCIÓN DE PRUEBAS

Recopilación de evidencias:

La recopilación de evidencias es fundamental para respaldar los hallazgos, facilitar el análisis de problemas y proporcionar información detallada a los equipos de desarrollo y gestión.

Estas evidencias sirven como base para documentar y comunicar los resultados de las pruebas y respaldar la toma de decisiones posteriores.

Existen planillas y formatos ad-hoc para estos fines.

DuocUC®

EJECUCIÓN DE PRUEBAS

Recopilación de evidencias:

La recopilación de evidencias es fundamental para respaldar los hallazgos, facilitar el análisis de problemas y proporcionar información detallada a los equipos de desarrollo y gestión.

Estas evidencias sirven como base para documentar y comunicar los resultados de las pruebas y respaldar la toma de decisiones posteriores.

Existen planillas y formatos ad-hoc para estos fines.

DuocUC®

EJECUCIÓN DE PRUEBAS

Documentación de defectos:

La documentación de defectos implica registrar y describir de manera precisa y detallada los problemas o defectos encontrados durante la ejecución de las pruebas.

Esto incluye identificar y describir claramente cada defecto, proporcionar información sobre cómo reproducirlo, indicar su gravedad o impacto en el software.

DuocUC®

EJECUCIÓN DE PRUEBAS

Documentación de defectos:

La documentación de defectos es esencial para comunicar los problemas encontrados a los equipos de desarrollo y permitir su corrección.

Además, ayuda a llevar un registro de los problemas identificados, su estado de resolución y su impacto en el producto final.

DuocUC®

EJECUCIÓN DE PRUEBAS

Documentación de defectos:

Para la realización de un informe de defecto es interesante recopilar los siguientes datos de cada defecto:

- Identificador único
- Título del defecto
- Fecha de detección del defecto
- Entorno o ambiente dónde se produce
- Caso de prueba donde se localizó
- Fase del ciclo de vida del proyecto
- Descripción completa indicando datos, pasos seguidos, condiciones, etc.
- Capturas de pantallas, grabaciones.

EJECUCIÓN DE PRUEBAS

Documentación de defectos:

Para la realización de un informe de defecto es interesante recopilar los siguientes datos de cada defecto:

- Resultado esperado
- El Resultado obtenido.
- Grado de impacto sobre el sistema (leve, mediano, grave)
- Estado en el que se encuentra el defecto. (abierto, cerrado, solucionado, para re-test, rechazado)
- Historial de cambios
- Impactos en otros componentes o sistemas
- Observaciones

EJECUCIÓN DE PRUEBAS

Documentación de defectos:

Quizás es mucha la cantidad de datos a definir, pero si se utiliza herramientas, muchos de estos campos se generarán de forma automática, y se ahorra tiempo sin escatimar en información.

En todo caso, aunque se registren de forma manual es menor el tiempo dedicado al registro inicial, que el necesario para recabar la información a posteriori.

EJECUCIÓN DE PRUEBAS

Registrar la solución antes de pasarlo a "resuelto“

No basta con registrar todos los datos del defecto en sí, también es importante llevar a cabo un registro de la solución realizada.

Hay desarrolladores que indican “ya está resuelto”. Es importante saber qué ocurrió y qué solución se ha aplicado.

De esta forma tendremos una base de datos donde consultar las distintas soluciones aplicadas a los defectos registrados.

EJECUCIÓN DE PRUEBAS

Para archivar la solución se deben tener en cuenta los siguientes valores:

- Causa del defecto
- Breve descripción de la solución aplicada
- Descripción ampliada
- Sistema que ha generado el defecto
- Origen del defecto
- Pasos seguidos para su resolución
- Pasos para poder evidenciar que ya no ocurre el defecto

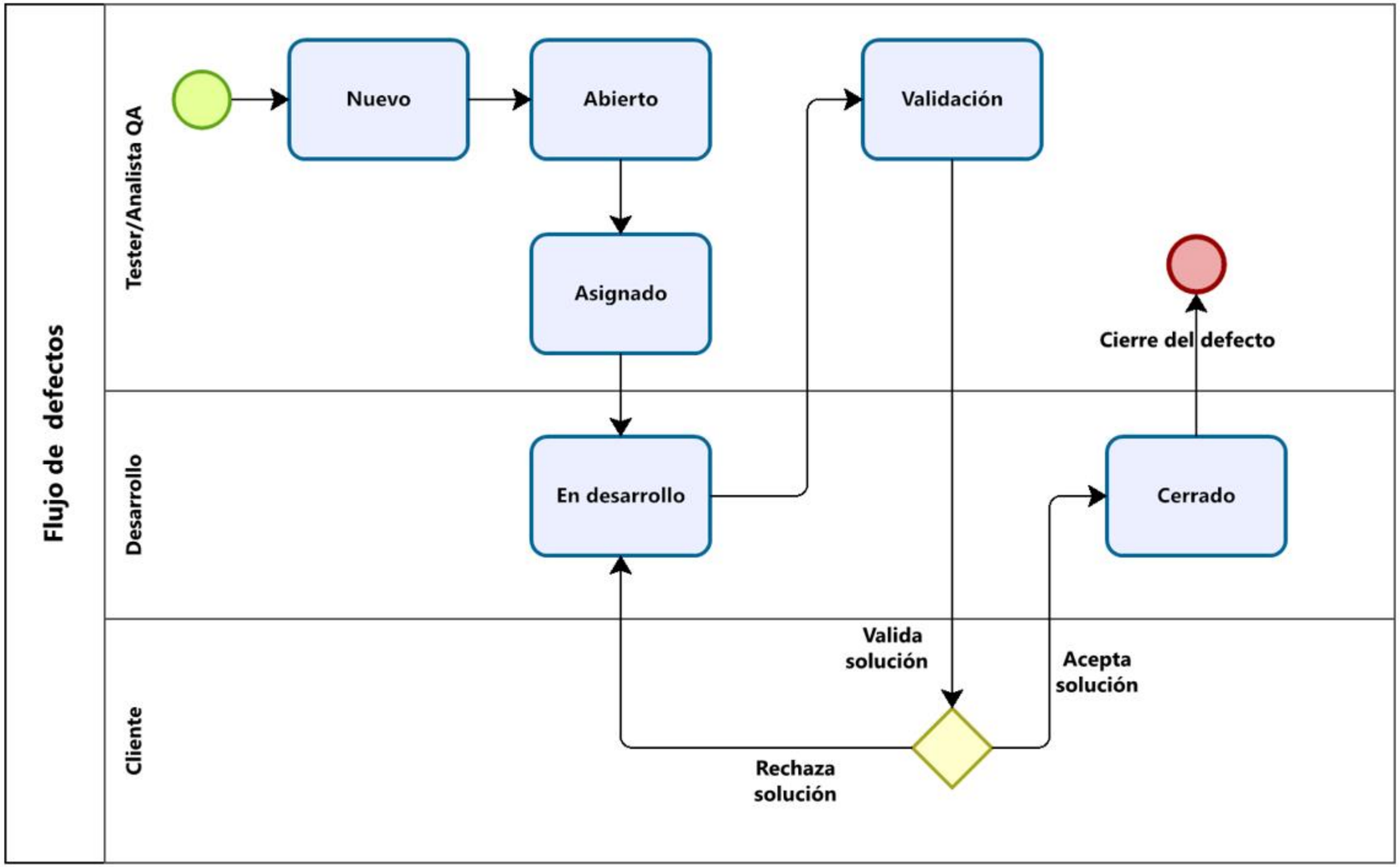
EJECUCIÓN DE PRUEBAS

Iteración con el equipo de desarrollo:

Posterior al ciclo de ejecución, se deben listar los defectos encontrados, este listado de defectos debe ser validado por el equipo de desarrollo para garantizar que realizamos correctamente la prueba, porque puede ser que estemos ejecutando el software con una variable desconocida hasta ese momento.

Después de ser validado se debe asignar una tipificación del defecto, es decir, si es leve, si es de gravedad menor o si es invalidante o bloqueante para continuar con las pruebas.

Flujo de defecto



DuocUC®

03

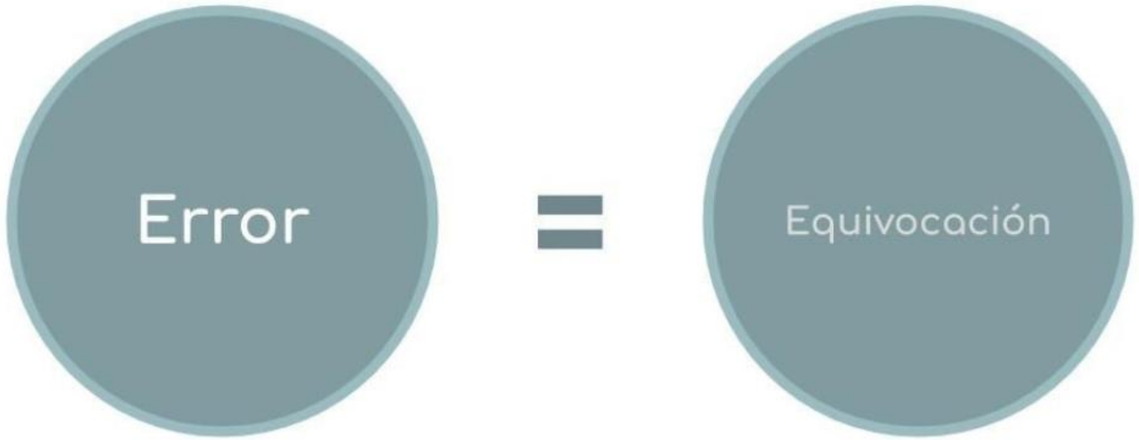
ERROR DEFECTO Y FALLO

DuocUC®

ERROR DEFECTO Y FALLO

¿Cuál es la diferencia entre los tres?

Un error es lo mismo que una equivocación, que por lo general es cometida por una persona, ya sea en el código del software o en algún otro producto de trabajo.



Cuando se comete un error, este introduce un defecto.

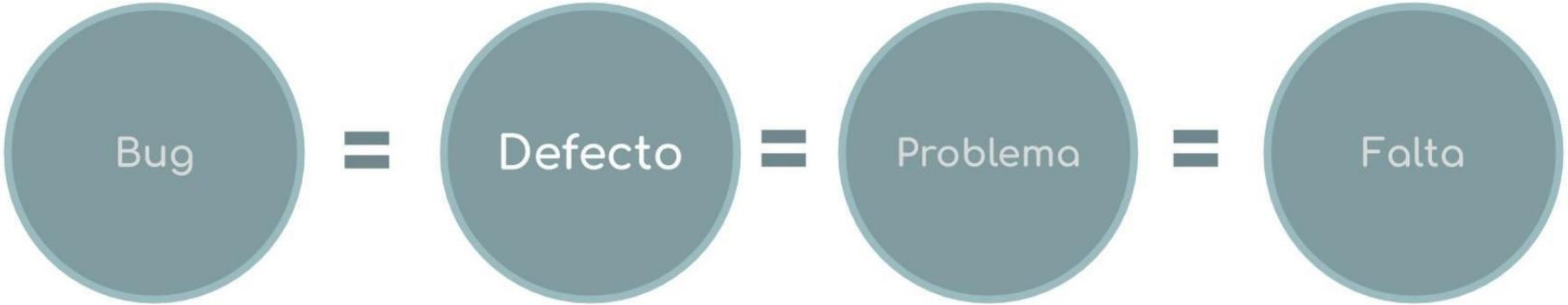
Un defecto, es lo mismo que un problema o falta y también es conocido como bug por los desarrolladores.

DuocUC®

ERROR DEFECTO Y FALLO

¿Cuál es la diferencia entre los tres?

Un defecto puede definirse como una imperfección en un componente o sistema que puede causar que este no desempeñe las funciones requeridas.

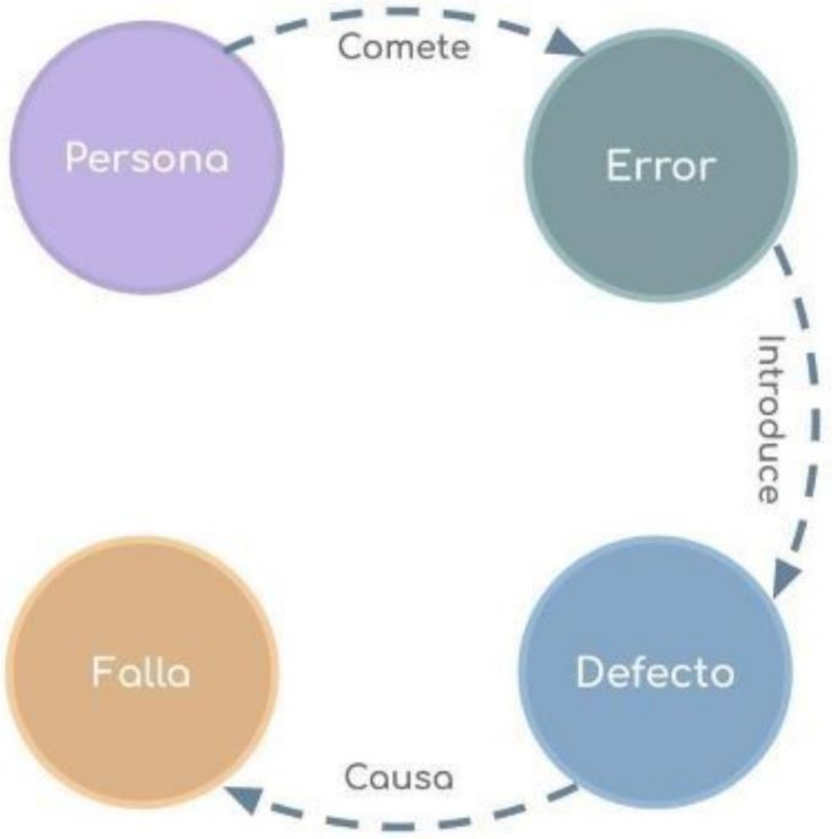


DuocUC®

ERROR DEFECTO Y FALLO

¿Cuál es la diferencia entre los tres?

Si un fragmento de código que contiene un defecto se ejecuta, es posible que cause un fallo, aunque esto no siempre pasa en todas las circunstancias, a veces se requiere de situaciones y datos muy específicos para que ocurra un fallo.



DuocUC®

ERROR DEFECTO Y FALLO

¿Ejemplo de la diferencia entre los tres:

Imagine que se le ha pedido al desarrollador que construya un sitio web en el cual solo pueden acceder personas de 15 años o más y él desarrolla un algoritmo parecido al siguiente:

```
if (edad > 15) {
  alert("Bienvenido a nuestro sitio web");
} else {
  alert("No puede acceder a este sitio");
}
```

La especificación incluye la edad de 15 años como una edad permitida y el algoritmo la excluye, por lo que el desarrollador cometió un error e introdujo un defecto en la condición.

ERROR DEFECTO Y FALLO

Causando el siguiente resultado en los casos de prueba:

AÑOS	RESULTADO ESPERADO	RESULTADO ACTUAL
14	No puede acceder a este sitio	No puede acceder a este sitio
15	Bienvenido a nuestro sitio web	No puede acceder a este sitio
16	Bienvenido a nuestro sitio web	Bienvenido a nuestro sitio web

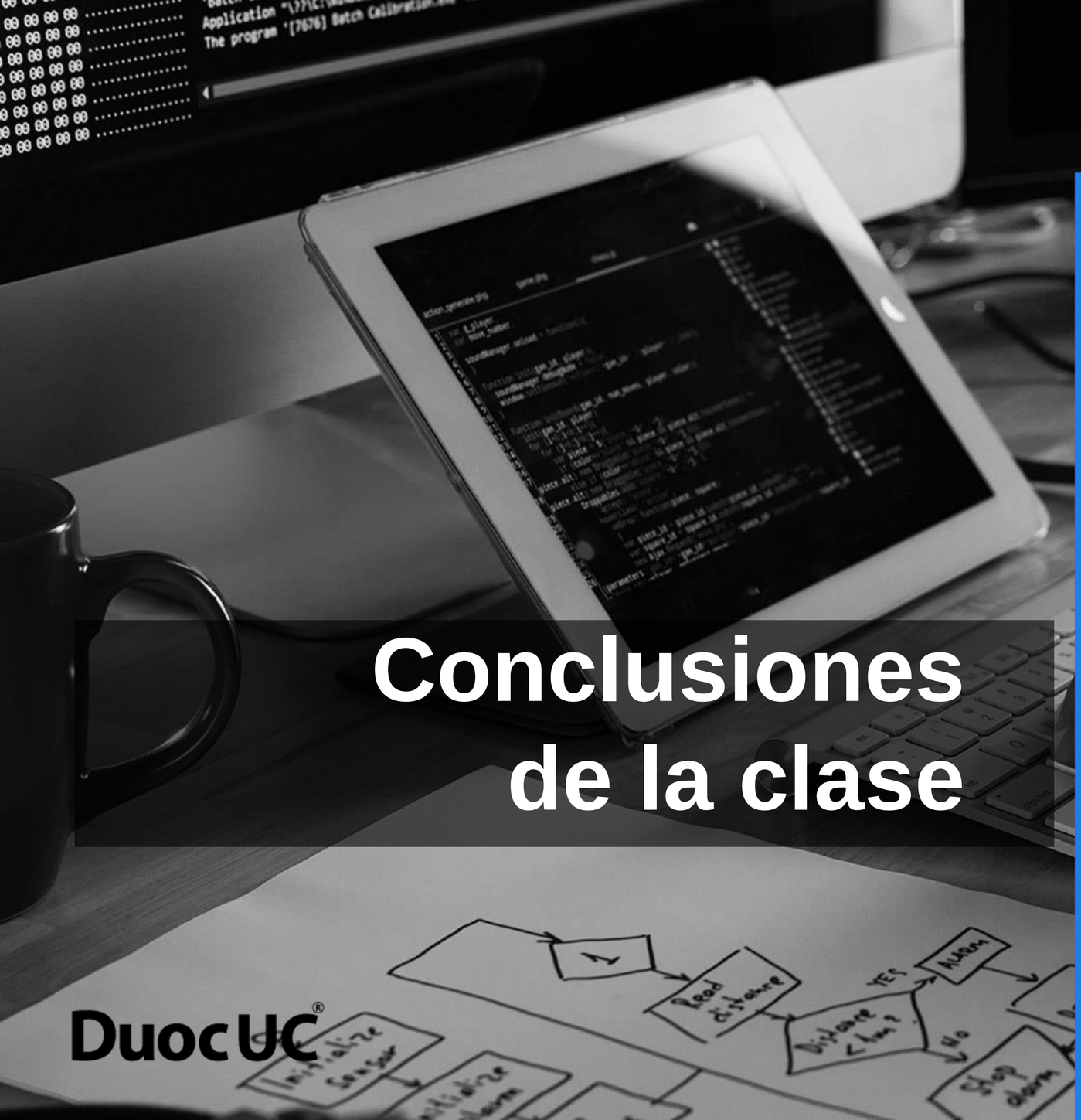
Con el defecto en el código, al introducir la edad 15 en la plataforma, aparecerá el siguiente mensaje:

No puede acceder a este sitio

Este es el fallo ocasionado por el defecto del código que fue causado por el error cometido por el desarrollador.

DuocUC®

Conclusiones de la clase



DuocUC®

Conclusiones

- ✓ Un buen registro de defectos produce una mejor gestión de la calidad

- ✓ No hay una forma única, para la gestión de defectos.

Bibliografía

- ✓ Reyes Sánchez García. (15 de mayo 2015). Testing: La importancia del Registro de defectos y de su corrección. https://elminimoviable.es/ Recuperado de https://elminimoviable.es/testing-la-importancia-del-registro-de-defectos-y-de-su-correccion/

- ✓ Stephen J. Bigelow. (31 de enero 2020). Cómo escribir un buen reporte de un error de software. https://www.computerweekly.com/ Recuperado de https://www.computerweekly.com/es/consejo/Como-escribir-un-buen-reporte-de-un-error-de-software

DuocUC®

DuocUC® INFORMÁTICA Y TELECOMUNICACIONES

Obtención de evidencias y registro de resultados

Calidad de Software - CSY4111