ENTORNOS DE DESARROLLO – UF2 (ACTIVIDAD A)

¿Cuáles son los dos aspectos fundamentales que debe de tener una batería de pruebas?

Cuando desarrollamos un software, una parte fundamental es la realización de pruebas para asegurarnos de su correcto funcionamiento. Cuando realizamos una batería de pruebas lo hacemos sobre dos puntos guía que nos ayudan en esta labor y nos permiten progresar en un entorno real de explotación con el máximo de garantías posibles.

* **Casuísticas habituales.** Donde se valoran todas aquellas circunstancias habituales que gestiona el programa.
* **Casuísticas críticas.** Aquí se da un enfoque sobre aquellos fallos que pueda tener el producto y que puede generar un problema grave.

¿Qué aspectos se deben de tener en cuenta en la planificación de un periodo de pruebas?

Cuando queremos realizar un periodo de pruebas, lo debemos hacer bajo una planificación de pruebas. Esta planificación se hace bajo unas pautas que nos sirven de guía, son las siguientes:

* CALENDARIO. Debemos saber cuándo y durante cuánto tiempo se realizará dicha planificación.
* NORMAS. Debemos saber el objetivo de la compañía, el ámbito de las pruebas, la metodología que se empleará, etc.
* RECURSOS. Saber qué hardware y qué software serán necesarios para la realización de los test, las herramientas que se van a utilizar y los perfiles de los técnicos y los usuarios que van a colaborar.
* PRODUCTOS. Aquí se incluyen todos los elementos que conforman la futura instalación en entorno real.
* CONTINGENCIA. Se valoran las alternativas de solución en case de que fallara la prueba principal del producto.

¿Cuáles son los tres principales tipos de pruebas que hay?

Cuando tenemos el producto finalizado, debemos pasar a comprobar su funcionamiento, este punto es esencial para su fiabilidad de cara a los usuarios. Existen tres tipos de pruebas para poder realizar este punto, las detallo a continuación:

* **Prueba funcional.** Se trata de pruebas que son ajenas al código del programa, es decir, miden los resultados que este origina y si estos se ajustan a la demanda del programa.
* **Prueba estructural.** Aquí se hace un estudio del código, de la estructura interna y del diseño técnico, pudiendo llegar a la conclusión de que el programa requiere alguna corrección, entrando a valorar el propio código del programa En este tipo de pruebas se utilizan casos relacionados con:
  + La lógica del código fuente.
  + Las estructuras de datos.
  + Las tablas.
  + Los tiempos de ejecución.
  + Control correcto de bucles de sentencias
* **Prueba de regresión.** Esta se aplica en el mantenimiento de un software que ya existe y al que se le requiere realizar una modificación porque, o bien contenía un error, o bien mejoraba el producto, o por ambas opciones. Si se trata de un producto nuevo, este tipo de prueba se realizará después de hacer el test principal y puede implicar la realización de cambios.

¿Qué son las herramientas de depuración?

Se trata de programas que tienen la tarea de probar el funcionamiento de otros programas informáticos, utilizando para ello las siguientes acciones:

* **Puntos de ruptura (breakpoint).** El programador puede poner en el código fuente unas señales que hacen que el depurador los interprete como punto de parada, así, cuando el programa llegue a ese punto, se detendrá temporalmente, pero mantendrá los valores de las variables, de las funciones y de los objetos en la memoria. Una vez analizados los datos, será posible continuar con la ejecución a partir de ese mismo punto.
* **Tipos de ejecución.** Según los intereses del programador, se podrá realizar la depuración de dos maneras:
  + Ejecución línea a línea. Cuando se desea que la depuración sea detallada.
  + Ejecución por grupos. El depurador se detendrá cuando termine una declaración completa.
* **Examinadores de variables.** Contamos con una ventana en el IDE al que le podremos seleccionar variables según nuestra conveniencia y nos irá señalando el valor que tienen estas justo en el punto en el que está la depuración.

¿Qué son las normas de calidad?

Se trata de documentos en los que se determinan los requisitos, las reglas y las formas de trabajo que una empresa está obligada a cumplir para que le puedan dar el certificado de esa norma. En cuanto a normas de calidad, las más destacables son:

* **ISO/IEC 15504 SPICE –** (Software Process Improvement Capability Determination). Esta facilita la mejora continua de los procesos de las organizaciones, especialmente los relacionados con el desarrollo y mantenimiento del software. Se basa en la evaluación, a través de niveles de madurez de la empresa. Hay 5 niveles de madurez:
  + Nivel 0 – inmadura.
  + Nivel 1 – Básica.
  + Nivel 2 – Gestionada.
  + Nivel 3 – Establecida.
  + Nivel 4 – Predecible.
  + Nivel 5 – Optimizado.
* **ISO 9001.** Es la solución más eficaz con la que pueden contar aquellas organizaciones que pretenden desarrollar e implantar un sistema de gestión de calidad. Esto abarca:
  + Desarrollo del software.
  + Servicios para el cliente.
  + Sistemas de mantenimiento y flujos administrativos.
* **ISO IEC 90003 2004.** Esta sirve de guía para una mejora constante. Para ello se gestiona sobre los siguientes ámbitos:
  + Requerimientos del sistema empresarial.
  + Requerimientos de gestión.
  + Requerimientos de recursos.
  + Requerimientos de control y realización.
  + Requerimientos de contingencia y reparación.
* **NORMATIVA IEEE 829-2008.** También es conocida como norma 829 para la documentación de test de sistemas y software.
  + Proporciona un conjunto estandarizado de documentos para las pruebas de software.
  + Las tareas del proceso de prueba se especifican para diferentes niveles de integridad.
  + Determinan la amplitud y profundidad apropiadas de la documentación de prueba.
  + El alcance de las pruebas abarca sistemas basados en software, hardware y sus interfaces.
  + Se aplica a los sistemas basados en software que se desarrollan, mantienen o reutilizan:
    - Elementos heredados.
    - Comerciales.
    - No evolutivos.

¿Qué es una prueba unitaria?

Una prueba unitaria es la que se realiza en un único componente del programa y lo hace de forma aislada. Dicho componente puede ser o bien un módulo, o bien una clase. Por diferentes motivos, el aislamiento del componente a probar se puede hacer complejo pero para ello existen unas herramientas que nos ayudan en esa labor, son las siguientes:

* **JUnit.** Se trata de un conjunto de clases desarrolladas en Java y sirven para comprobar si un programa funciona correctamente.
* **PHPUnit.** Esta herramienta está orientada a la creación de juegos para pruebas unitarias en lenguaje PHP.
* **NUNit.** Es un gestor de pruebas unitarias para los lenguajes .NET. Esta herramienta está escrita en C#.

7. Verdadero o falso.

Identificador del plan de pruebas – V

Descripción del plan de pruebas – V

Descripción del proyecto – F

Elementos del software que no se han de probar – V

Definición de la configuración del plan de pruebas – V

Requisitos del proyecto – F

Documentos a liberar – V

Responsables y responsabilidades – V

Partes interesadas – F

Calendario del plan de pruebas – V

Calendario de la fase de desarrollo – F

8 .- Relaciona

Son las encargadas de detectar los errores en la implementación de los requerimientos de usuario – D – TIPOS DE PRUEBAS FUNCIONALES

El método utilizado es de caja blanca, es de diseño descendiente y el de bottom-up – C- TIPOS DE PRUEBAS DE INTEGRACIÓN

Su finalidad es detectar errores en la obtención de los requerimientos – B – TIPOS DE PRUEBAS DE SISTEMAS

Su objetivo es la validación de la aplicación por parte de los usuarios – A - TIPOS DE PRUEBAS DE ACEPTACIÓN