Aseguramiento de Calidad de Software

**Carrera: Ingenieria en Software**

**Año: 2022**

**Profesores:**

**- Julieta Barrionuevo**

**- Juan Manuel Ojeda**

Añade encabezados (Formato > Estilos de párrafo) y aparecerán en el índice.

Contenido TP2

**Clase N°3**

**Herramientas para el testing Manual**

Para el testing manual existen un numero importante de herramientas que podemos utilizar para testear “features” y poder asegurar un nivel de calidad aceptable.

Podemos pensar en Feature como un Conjunto de funcionalidades

Dentro de las herramientas podemos mencionar las siguientes

* Casos de Uso
* Diagramas O.O ( Clase , ER , Secuencia)
* Wireframes
* SQL – Bases de Datos
* Documentación del sistema bajo prueba

Además de las mencionadas existe una herramienta que hoy por hoy es la herramienta por excelencia utilizada por persona que se dedica a la calidad de software. Esta herramienta se llama Caso de prueba (Test Case )

**Definición:**

Existen muchas definiciones que intentan describir lo que es un buen test case y muchas de ellas son válidas.

Para fines prácticos podemos decir que un test case (Caso de prueba) es un conjunto de pasos/acciones que al seguirlas de forma ordenada e ingresando datos validos cuando sea necesario permiten verificar el correcto funcionamiento ( por lo tanto desarrollo ) del feature que está siendo probado.

• ***Que es un Test Case?***

Podemos definir un test case como una mera secuencia de acciones ( pasos ) que son realizados para poder VERIFICAR que el sistema nos da el resultado ( outcome ) que es esperado.

**Clase N°4 - Test Cases**

• ***Que es un escenario y como se relaciona con un test case?***

Cuando nos encontramos en etapas de análisis y de definición/Creación de test cases podemos comenzar a dudar de si es una buena idea generar un test case por cada uno de lo que nosotros podemos entender como “casos de prueba”

La pregunta que generalmente sale a la luz es… “Cuantos test cases necesito para cubrir una funcionalidad que se encuentra siendo testeada?”

Podemos concebir a un escenario como una situación hipotética dentro de nuestro sistema. Es decir, una combinación de variables dentro las posibilidades del feature que estamos evaluando en un momento determinado.

Un escenario ideal valido es aquel que es creíble, fácil de evaluar y a su vez complejo.

Se diferencia a los escenarios de testing con los test cases en el sentido de que un escenario puede ser validado por más de un test case

**https://wiki.evolveum.com/display/midPoint/Test+Scenarios**

***Por ejemplo:***

***Escenario:*** *Usuario ingresa sus credenciales para ingresar al sistema*

***Validaciones necesarias:***

*Existencia de componentes de pantalla, Valor de las variables presentes en la pantalla, internacionalización, seguridad, cifrado de los datos, etc.*

*Supongamos que queremos testear el escenario en donde el usuario desea ingresar al sitio con sus credenciales. Realice el siguiente ejercicio*

• ***Por que son necesarias las Pruebas?***

Sea de nuestro agrado o no, la actualidad nos muestra que el SOFTWARE es moneda corriente y esta en nuestras vidas todo el tiempo ( de hecho estamos leyendo y escribiendo en pantallas mas tiempo del que interactuamos con personas ) tenemos en promedio mas de 10 pestañas abiertas cada vez que trabajamos , estudiamos o hacemos alguna investigacion.

No muy lejos suyo , debe estar su telefono y quizas una tablet ( uno o mas ) la cual se usa para leer, ver series, y otras actividades.

No podemos concebir vida productiva sin Software ( al menos en nuestro rubro ) casi en ninguna actividad productiva del siglo XXI

Dicho esto… No le parece frustrante cuando la app/pagina/tool que esta usando deja de responder o funcionar como normalmente hace? que pasa cuando su sitio preferido demora mas tiempo en cargar? que pasa cuando queremos enviar una impresion y el duende de las impresiones no nos deja hacerlo?

No podemos negar que es mas que frustrante y hoy por hoy, la paciencia con el software es practicamente nula.

**Esta catedra pretende mostrar que el area de Calidad (QA) debe ser parte integral en cualquier proyecto de software.**

El truco detrás de las pruebas QA está en identificar con precisión la calidad del software con el objetivo de asegurar que el software funcione como se espera que funcione en todo momento.

El término se refiere a diferentes métodos y procesos para probar software y garantizar su calidad.

Por eso las pruebas QA son su mejor aliado en la relación costo beneficio, y para poder garantizarse software de excelente calidad.

**Algunos beneficios del Area de QA en general - Recap.**

| Topico | Descripción |
| --- | --- |
| Ahorrar Dinero y evitar problemas | ¿Cuánto dinero le cuesta un proyecto de software defectuoso? Les cuesta usuarios y a los clientes. Y es bien sabido que cuanto más tiempo pasa inadvertido un error en su software, más difícil y costoso se vuelve solucionarlo. Al emplear pruebas de aseguramiento de la calidad durante todo el proceso de desarrollo del software, ahorrará tiempo y dinero después de la implementación. |
| Evitar errores en produccion y por ende impacto negativo en su negocio | Con el software corporativo, lo que está en juego es aún más importante. Los errores en el software corporativo pueden ocasionar apagones del sistema, datos faltantes y fallas de comunicación. Si va a utilizar software en una empresa o para manejar información sensible, entonces debe asegurarse de que el software funcionará exactamente como debe funcionar. No hay margen de error. |
| Mas confianza traen mas usuarios | Al hacer que el Software Testing para Aseguramiento de la Calidad tenga una clara prioridad para el desarrollo de software, está enviando un mensaje a sus clientes indicando que desea que su software tenga el mayor éxito posible. Esto es extraordinariamente importante cuando se trata de ofrecer calidad y forjar relaciones a largo plazo. |
| Mejorar la experiencia del usuario en su aplicacion | Cada vez es más claro que la experiencia del usuario creará o romperá un producto. Si el software está fallando o es lento, entonces interrumpe la experiencia del usuario con el producto. La mala experiencia del usuario resulta en insatisfacción y frustración. Una buena experiencia de usuario, lo que obtiene cuando ha probado meticulosamente un producto de software, resulta en un usuario satisfecho, que es mucho más probable que recomiende el producto y su negocio a otros. |
| Organizar al equipo de desarrollo y empoderar la productividad y eficiencia | Nadie quiere lidiar con software defectuoso, comunicación frenética, ni correcciones apresuradas. Organizarse con pruebas de control de calidad desde el comienzo de su estrategia de desarrollo le permitirá trabajar en paz y ser más productivo con su tiempo. Al utilizar metodologías ágiles, donde los desarrolladores de software crean y entregan pequeños fragmentos de  un producto en una línea de tiempo clara, se puede comenzar a probar el software a medida que se crea, en lugar de esperar siempre hasta el final. Cuando las pruebas de software son una parte integral de su estrategia de software, usted gana, su cliente gana y sus usuarios ganan. |

**Cuando Automatizar?**

A veces caemos dentro de la falsa idea que automatizar nuestras pruebas sera la solucion definitiva a nuestros problemas de Calidad de Software.

Esto es comunmente un error que no es dificil evitar, pero ciertamente se comente mas veces de lo que uno supone.

Cuando se recomienda empezar a PENSAR en automatizar?

* Hay procesos repetitivos que no varian en el tiempo
* Hay flujos extensos que requieren actualizaciones y es necesario probar en regresión
* Hay casos complejos que consumen mucho tiempo y no se tiene el bandwidth necesario
* Es necesario correr múltiples pruebas y muchas veces
* Es fundamental minimizar los errores humanos
* Se requiere probar componentes críticos de forma exhaustiva
* Se precisa realizar pruebas en múltiples plataformas y no es humanamente posible

La siguiente pregunta es ¿Cómo automatizar?

Existen diferentes maneras de hacerlo. A través de herramientas complejas o con alto grado de programación, puede ser algo open source, o bien, pagar una licenciadas. En este sentido podemos afirmar que existe gran variedad de herramientas simples y amigables.

Lo importante al generar su estrategia de automatizacion es:

* emplear soluciones ágiles, fáciles de utilizar y escalables
* los cambios pueden incorporarse sencillamente.
* Validar el ROI cuando la misma se ejecuta repetidas veces a lo largo de la vida del proyecto.

Con algunas herramientas cada cambio en la aplicación requerirá obligatoriamente el mantenimiento del codigo de la automatizacion y la dedicación de recursos para esto.

Dado lo anterior, han surgido tecnologías que permiten una asimilación de uso más generalizado y facilitan la creación de las automatizaciones a personas con bajo o nulo conocimiento de programación.

Hay herramientas muy potentes que son cero código, como STELA, con motores de IA. Se trata de una solución que además de simplificar la generación de automatizaciones a través de drap and drop, le da una vida útil más alta a las automatizaciones, maximizando su rentabilidad. No demanda gran dedicación en la construcción de robots, ni requiere personal técnico especializado, así como tampoco necesita de mantenimiento exhaustivo, ni frecuente al aplicar técnicas de Visión Artificial para el reconocimiento de elementos en la pantalla aunque cambien de posición.

**Ventajas de automatizar**

| Objetivo | Descripcion |
| --- | --- |
| **Mayor capacidad para la ejecución de pruebas.** | Es posible realizar un gran número de pruebas en un breve período de tiempo. Pueden ejecutarse las 24 horas, los 7 días de la semana, incluso cuando el equipo de test no esté en la oficina. Pueden realizarse en forma paralela y sobre distintos servidores, maximizando su capacidad.  También permiten mayor cubrimiento de casos, por lo que se logran pruebas muy exhaustivas y precisas de la funcionalidad y más condiciones y escenarios probados. |
| **Integración continua y Devops.** | Devops es una importante tendencia en la construcción de software moderno. Tiene como objetivo agilizar el proceso de liberación de software, buscando responder rápidamente a las exigencias de los negocios. En ese sentido, que las pruebas estén automatizadas, resulta vital para acelerar todo el proceso de entrega, ya que permite, de manera temprana realizar pruebas críticas antes de la salida a producción. |
| **Incrementa indicadores de calidad y confiabilidad.** | Las pruebas automatizadas son ideales para detectar fallas de regresión, de integración y funcionales, entre otras. El papel de los líderes o responsables de testing será crucial y más participativo, al reportar e informar a tiempo y con mayor precisión oportunidades de mejora y fallas detectadas para que el software cumpla con los estándares de calidad establecidos, entregando versiones confiables. |
| **Ahorro sustancial de tiempo.** | Las pruebas automatizadas multiplican la capacidad de los equipos de testing. Minimizan bastante la ejecución de las pruebas manuales, necesitando estas últimas solo en escenarios especiales |
| **Mejora la productividad y la satisfacción laboral de los recursos.** | Al optimizar el tiempo y liberar a los recursos humanos de tareas repetitivas, se gana en productividad y se mejora su motivación, ya que estarán más dispuestos a realizar otro tipo de tareas que sigan aportando a la mejora del negocio.  Empleando una herramienta como STELA, los recursos tendrán mayor participación, la construcción de las automatizaciones es tan simple que en promedio se incrementa un 60% la productividad. La evidencia y los reportes facilitan la labor. Cada persona va a poder mantener y reusar las automatizaciones, escalando sus capacidades y el aporte a los proyectos. |
| **Mayor precisión para la corrección de fallas de software.** | Las pruebas automatizadas facilitan la precisión a la hora de diagnosticar la falla detectada. Proveen la evidencia en una serie de reportes, con algunas herramientas, es posible obtener un video que permite analizar en profundidad la situación, y el paso a paso del bug detectado. Al utilizar distintos juegos de datos en el mismo caso de prueba, es posible ser muy certero en el diagnóstico de las situaciones problemáticas. |