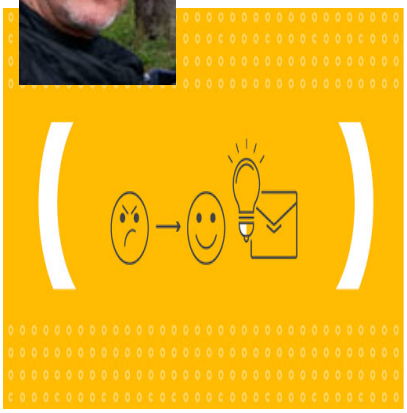

Pruebas de software y la diferencia entre queja y recomendación

Descripción

[vc_row][vc_column width=”1/6?][/vc_column][vc_column width=”2/3?][vc_column_text]

Este es un tema imperativo en el proceso de puesta en producción que llevo realizando desde el desarrollo de mi primer programa en Fortran en el año 1984 y gracias a la visión extraoridnaria de uno de mis profesores de programación en mi primer paso por el Politecnico JIC.

Posteriormente en una conversación con mi primo Pablo Zapata que en esa epoca se desempeñaba como director de IBM para Latinoamerica y uno de los responsables de mi amor por la tecnología me dijo una frase que siempre recuerdo: “La diferencia entre una queja y una recomendación es la fase de pruebas con el cliente, se claro con el cliente y tendrás el mejor probador de tu programa”.

En aquella epoca no exitia el termino Beta Tester. He aqui un artículo que puede aydar a aquello que desean llevar a buen puerto un lanzamiento de su aplicación.

“En un proceso de pruebas formal, suelen confundirse con mucha facilidad, los niveles de pruebas con los tipos de prueba, y a pesar de que se encuentren íntimamente relacionadas, tienen connotaciones diferentes en el proceso. Para entender un poco más, vamos a partir del hecho de que las pruebas pueden ejecutarse en cualquier punto del proceso de desarrollo de software, y es aquí donde los niveles de prueba nos permiten entender con claridad los diferentes puntos o etapas en donde pueden ejecutarse ciertos tipos de prueba. Por lo anterior, es común que algunas personas se refieran a los niveles de pruebas o intenten clasificarlos como: pruebas de desarrollador, pruebas funcionales y pruebas de usuario final. Sin embargo, la terminología apropiada para referirse a los diferentes niveles corresponde a la siguientes cuatro (4) clasificaciones que son: pruebas unitarias, pruebas de integración, pruebas de sistema y pruebas de aceptación. En cada uno de estos niveles de prueba, se podrán ejecutar diferentes tipos de prueba tales como: pruebas funcionales, no funcionales, de arquitectura y asociadas el cambio de los productos.

A continuación una breve descripción de cada nivel de prueba:

1. **Pruebas Unitarias o de Componente:** este tipo de pruebas son ejecutadas normalmente por el

o de desarrollo, básicamente consisten en la ejecución de actividades que le permitan ar al desarrollador que los componentes unitarios están codificados bajo condiciones de robustez, esto es, soportando el ingreso de datos erróneos o inesperados y demostrando así la capacidad de tratar errores de manera controlada. Adicionalmente, Las pruebas sobre componentes unitarios, suelen denominarse pruebas de módulos o pruebas de clases, siendo la convención definida por el lenguaje de programación la que influye en el término a utilizar. Por último, es importante que toda la funcionalidad de cada componente unitario sea cubierta, por al menos, dos casos de prueba, los cuales deben centrarse en probar al menos una funcionalidad positiva y una negativa.

2. **Pruebas de Integración:** este tipo de pruebas son ejecutas por el equipo de desarrollo y consisten en la comprobación de que elementos del software que interactúan entre sí, funcionan de manera correcta.
3. **Pruebas de Sistema:** este tipo de pruebas deben ser ejecutadas idealmente por un equipo de pruebas ajeno al equipo de desarrollo, una buena práctica en este punto corresponde a la tercerización de esta responsabilidad. La obligación de este equipo, consiste en la ejecución de actividades de prueba en donde se debe verificar que la funcionalidad total de un sistema fue implementada de acuerdo a los documentos de especificación definidos en el proyecto. Los casos de prueba a diseñar en este nivel de pruebas, deben cubrir los aspectos funcionales y no funcionales del sistema. Para el diseño de los casos de prueba en este nivel, el equipo debe utilizar como bases de prueba entregables tales como: requerimientos iniciales, casos de uso, historias de usuario, diseños, manuales técnicos y de usuario final, etc. Por último, es importante que los tipos de pruebas ejecutados en este nivel se desplieguen en un ambiente de pruebas / ambiente de pre-producción cuya infraestructura y arquitectura sea similar al ambiente de producción, evitando en todos los casos utilizar el ambiente real del cliente, debido principalmente, a que pueda ocasionar fallos en los servidores, lo que ocasionaría indisponibilidad en otros servicios alojados en este ambiente.
4. **Pruebas de Aceptación:** Independientemente de que se haya tercerizado el proceso de pruebas y así la firma responsable de estas actividades haya emitido un certificado de calidad sobre el sistema objeto de prueba, es indispensable, que el cliente designe a personal que haga parte de los procesos de negocio para la ejecución de pruebas de aceptación, es incluso recomendable, que los usuarios finales que participen en este proceso, sean independientes al personal que apoyó el proceso de desarrollo. Cuando las pruebas de aceptación son ejecutadas en instalaciones o ambientes proporcionados por la firma desarrolladora se les denominan pruebas Alpha, cuando son ejecutadas desde la infraestructura del cliente se les denomina pruebas Beta. En los casos en que las pruebas de aceptación del producto se hayan ejecutado en el ambiente del proveedor, el aplicativo no podrá salir a producción, sin que se hayan ejecutados las respectivas pruebas Beta en el ambiente del cliente, de lo anterior es importante concluir, que las pruebas Alpha son opcionales, pero las pruebas Beta son obligatorias.

Metodología de Pruebas

Los procesos de aseguramiento de calidad de un producto de software suelen dividirse en lo que respecta a su componente analítico en pruebas estáticas y dinámicas. La diferencia fundamental

e tipos de pruebas, radica en que las pruebas estáticas se centran en evaluar la calidad con la tá generando la documentación del proyecto, por medio de revisiones periódicas, mientras que las pruebas dinámicas, requieren de la ejecución del software con el fin de medir el nivel de calidad con la que este fue codificado y el nivel de cumplimiento en relación con la especificación del sistema.

Realizar pruebas dinámicas a un producto de software, suele en la mayoría de los casos confundirse con una simple actividad de ejecución de pruebas y reporte de incidencias, sin embargo, para productos de complejidad media en adelante, lo recomendable es implementar de manera formal una metodología de pruebas que se ajuste y acople uniformemente con la metodología de desarrollo seleccionada por la firma desarrolladora.

Para procesos de desarrollo basados en la metodología RUP o métodos tradicionales, implementar una metodología de pruebas es totalmente viable, teniendo en cuenta que estas metodologías están orientadas a la documentación y a la formalización de todas las actividades ejecutadas. Si por el contrario, la firma desarrolladora guía su proceso bajo lineamientos basados en metodologías ágiles, será necesario reevaluar la conveniencia de ejecutar todas las actividades que implica un proceso de pruebas formal, lo que en la mayoría de los casos, conlleva a reducir al mínimo las actividades relacionadas con un proceso de pruebas, circunstancia que naturalmente puede desencadenar en la liberación de productos con bajos niveles de calidad.

Un proceso de pruebas formal, está compuesto, cuando menos por las siguientes 5 típicas etapas:

1. Planeación de pruebas.
2. Diseño de pruebas.
3. implementación de pruebas.
4. Evaluación de criterios de salida.
5. Cierre del proceso.

Planeación de Pruebas.

Es la etapa en donde se ejecutan las primeras actividades correspondientes al proceso de pruebas y tiene como resultado un entregable denominado plan de pruebas el cual debe estar conformado en cuando menos por aspectos tales como:

- **Alcance de la prueba:** determina que funcionalidades del producto y/o software serán probadas durante el transcurso de la prueba. Este listado de funcionalidades a probar se extrae con base a un análisis de riesgos realizado de manera previa, que tienen en cuenta variables tales como el impacto que podría ocasionar la falla de una funcionalidad y la probabilidad de falla de una funcionalidad. Producto de este análisis, se cuenta con información adicional que permite determinar además del alcance detallado del proceso de pruebas, la prioridad con la que las funcionalidades deben probarse y la profundidad de las pruebas.
- **Tipos de Prueba:** en este punto se debe determinar qué tipos de pruebas requeriría el producto. No todos los productos de software requieren la aplicación de todos los tipos de pruebas que existen, por esta razón, es estrictamente necesario que el líder de pruebas se plantee preguntas que le permitan determinar qué tipos de prueba son aplicables al proyecto en evaluación. Los posibles tipos de prueba a aplicar son: pruebas de stress, pruebas de rendimiento, pruebas de

pruebas funcionales, pruebas de usabilidad, pruebas de regresión, entre otros.
egia de Pruebas: teniendo en cuenta que no es viable probar con base a todas las posibles combinaciones de datos, es necesario determinar a través de un análisis de riesgos sobre que funcionalidades debemos centrar nuestra atención. Adicionalmente, una buena estrategia de pruebas debe indicar los niveles de pruebas (ciclos) que aplicaremos y la intensidad o profundidad a aplicar para cada nivel de pruebas definido. En este punto también es importante definir los criterios de entrada y salida para cada ciclo de pruebas a ejecutar.

- **Criterios de Salida:** entre las partes involucradas en el proceso, se define de manera formal, bajo qué condiciones se puede considerar que una actividad de pruebas fue finalizada. Los criterios de salida se deben definir para cada nivel de pruebas a ejecutar. Algunos ejemplos de criterios de salida que pueden ser utilizados son: porcentaje de funcionalidades de alto riesgo probadas con éxito, número defectos críticos y/o mayores aceptados, etc.
- **Otros aspectos:** tal y como se realiza en cualquier plan de proyecto, se debe incluir una estimación de tiempos, los roles y/o recursos que harán parte del proceso, la preparación del entorno de pruebas, cronograma base, etc.

Diseño de Pruebas: una vez elaborado y aprobado el plan de pruebas, el equipo de trabajo debe iniciar el análisis de toda la documentación existente con respecto al sistema, con el objeto de iniciar el diseño de los casos de prueba. Los entregables claves para iniciar este diseño pueden ser: casos de uso, historias de usuario, arquitectura del sistema, diseños, manuales de usuario (si existen), manuales técnicos (si existen). El diseño de los casos debe considerar la elaboración de casos positivos y negativos. Los casos de prueba negativos permiten validar cómo se comporta el sistema ante situaciones atípicas y permite verificar la robustez del sistema, atributo que constituye unos de los requerimientos no funcionales indispensable para cualquier software. Por último, es necesario definir cuáles son los datos de prueba necesarios para la ejecución de los casos de prueba diseñados.

Implementación y Ejecución de Pruebas: la ejecución de pruebas debe iniciar con la creación de los datos de prueba necesarios para ejecutar los casos de prueba diseñados. La ejecución de estos casos, puede realizarse de manera manual o automatizada; en cualquiera de los casos, cuando se detecte un fallo en el sistema, este debe ser documentado y registrado en una herramienta que permita gestionar los defectos (Bug Tracker). Una vez el defecto ha sido corregido por la firma desarrolladora en su respectivo proceso de depuración, es necesario realizar un re-test que permita confirmar que el defecto fue solucionado de manera exitosa. Por último, es indispensable ejecutar un ciclo de regresión que nos permita garantizar, que los defectos corregidos en el proceso de depuración de la firma, no hayan desencadenado otros tipos de defectos en el sistema.

Evaluación de Criterios de Salida: los criterios de salida son necesarios para determinar si es posible dar por finalizado un ciclo de pruebas. Para esto, es conveniente definir una serie de métricas que permitirán al finalizar un proceso de pruebas, comparar los resultados obtenidos contra las métricas definidas, sí los resultados obtenidos no superan la métricas definidas, no es posible continuar con el siguiente ciclo de pruebas.

Existen varios tipos de criterios de salida dentro de los cuales se pueden mencionar: cubrimiento de funcionalidades en general, cubrimiento de funcionalidades críticas para el sistema, Número de defectos críticos y mayores detectados, etc. También es importante aclarar que el proceso de

p ede ser suspendido y/o paralizado, debido entre otros, a aspectos relacionados con el p y la calidad mínima del sistema requerida para el inicio formal de pruebas.

Cierre del proceso: durante este periodo de cierre el cual históricamente se ha comprobado que se le destina muy poco tiempo en la planeación, se deben cerrar las incidencias reportadas, se debe verificar si los entregables planeados han sido entregados y aprobados, se deben finalizar y aprobar los documentos de soporte de prueba, analizar las lecciones aprendidas para aplicar en futuros proyectos, etc.

Principios Fundamentales del Proceso de Pruebas

En el proceso de Ingeniería de Software, existen, dependiendo la metodología de desarrollo utilizada (*SCRUM, RUP, etc*), una serie de etapas claramente identificadas en el proceso, tales como: análisis diseño, implementación, pruebas, etc. Sin embargo, por algunas razones que no vale la pena discutir en este apartado, es la etapa correspondiente al proceso de pruebas de un producto, a la que las firmas desarrolladoras menos tiempo de planeación asignan y a la que menos recursos comprometen en sus respectivos planes de trabajo. En muchos casos incluso, las fábricas de software, suelen acudir a una mala práctica en donde involucran personal de desarrollo, para ejecutar actividades de pruebas, sin tener en cuenta que estas personas, debido al rol que asumen dentro del proceso, de manera involuntaria protegerán el estado de su producto, muchas veces minimizando la criticidad de los fallos encontrados, sesgando de esta manera el grado real de calidad del producto. Por esta y muchas razones más, es altamente aconsejable, que esta etapa de pruebas sea ejecutada por un equipo ajeno a los procesos de desarrollo o en el mejor de los casos por un tercero especializado en el aseguramiento de la calidad del software. El presente artículo esta orientado para aquellas firmas o personas que se inician en la ingeniería del software y que estén reconociendo las necesidades inherentes de incorporar dentro de su proceso actividades de aseguramiento de calidad como la planeación y ejecución de pruebas sobre sus productos. A continuación quiero quiero dar a conocer 5 principios fundamentales, para los equipos o personas que hasta ahora, incursionan en metodologías de pruebas.

1. **Las pruebas exhaustivas no son viables:** para proyectos cuyo número de *casos de uso* o *historias de usuario* desarrolladas sea considerable, se requeriría de una inversión muy alta en cuanto a tiempo y recursos necesarios para cubrir pruebas sobre todas las funcionalidades del sistema; por esta razón, es conveniente realizar un análisis de riesgos de todas las funcionalidades del aplicativo y determinar en este punto cuales serán objeto de prueba y cuales no. Naturalmente, ninguna funcionalidad que haga parte del ciclo de negocio del aplicativo debe quedar por fuera de esta revisión. Por otra parte, es necesario evitar para el caso de funcionalidades complejas, escribir (n) casos de prueba, que cubran todas las posibles combinaciones de entrada y salida que puede llegar a tener las funcionalidades. Diseñar casos de prueba bajo estas condiciones, solo es justificable cuando la funcionalidad objeto de prueba tiene una complejidad trivial. Por las razones ya mencionadas, es altamente sugerible diseñar y ejecutar *pruebas de muestra*, las cuales sean elegidas bajo criterios de experiencia y/o aleatoriedad.
2. **El proceso no puede demostrar la ausencia de defectos:** independientemente de la

sidad con la que se haya planeado el proceso de pruebas de un producto, nunca será e garantizar al ejecutar este proceso, la ausencia total de defectos de un producto en su paso a producción, debido entre otras cosas, al principio no. 1, en el cual no se permite escribir y ejecutar casos de prueba de manera exhaustiva. Por lo anterior, un proceso de pruebas planeado, puede garantizar una reducción significativa de los posibles fallos y/o defectos del software, pero nunca podrá garantizar que el software no fallará en su ambiente de producción.

3. **Inicio temprano de pruebas:** es típico ver como algunas firmas de desarrollo, ven el proceso de pruebas como una serie de actividades que se dan de manera aislada y solo hasta el momento en el que se tiene una release de pruebas del producto, el equipo de pruebas se incorpora a ejecutar las respectivas actividades; aunque sea válido, lo recomendable es que las actividades de pruebas se ejecuten de manera paralela con cada una de las etapas del proceso. Las actividades de un proceso de pruebas, deben ser incorporadas incluso desde el mismo momento en el que se ejecutan las etapas de análisis y diseño, por esta razón, documentos de especificación y de diseño, deben ser sometidos a revisiones, lo que ayudará a detectar problemas en la lógica del negocio mucho antes de que se escriba una sola línea de código. En conclusión, cuanto mas temprano se detecte un defecto bien sea sobre los entregables de especificación y diseño o sobre el producto , menos costoso será para el equipo del proyecto dar solución a dichos incidentes.
4. **Las pruebas no garantizan la calidad del Software:** si bien las pruebas del Software ayudan a mejorar la calidad de un producto, esto no es totalmente garantizable, si estas actividades no son incorporadas desde etapas tempranas del proyecto. Este nivel de calidad no será garantizado, entre otros aspectos, porque existe la posibilidad de que algunas funcionalidades del software pueden no suplir las necesidades y expectativas de los usuarios finales a los cuales va dirigido el desarrollo, así el comportamiento del software sea correcto y responda fielmente a lo que fue especificado. Una buena práctica que ayuda a mitigar el riesgo de que el usuario final no este satisfecho con el producto, es involucrarlo desde instancias tempranas en el proceso y tener en cuenta sus apreciaciones para generar una retroalimentación a tiempo.
5. **Ejecución de pruebas bajo diferentes condiciones**: en un plan de pruebas, siempre existe un apartado relacionado con la estrategia a utilizar por parte del equipo de pruebas, en este item, se define entre otros aspectos, el número de ciclos de prueba que se ejecutarán sobre las funcionalidades del negocio. La idea consiste, en que por cada ciclo de prueba ejecutado, se generen diferentes tipos de condiciones, basados principalmente en la variabilidad de los datos de entrada y set de datos utilizados. No es conveniente, ejecutar en cada ciclo, los casos de prueba basados en los mismos datos del ciclo anterior, dado que con seguridad, se obtendrán los mismos resultados. En conclusión, ejecutar ciclos bajo diferentes tipos de condiciones, permitirá identificar posibles fallos en el sistema, que antes no eran fácilmente reproducibles.”

Créditos: Javier Zapata Sánchez

[/vc_column_text][/vc_column][vc_column width=”1/6?][/vc_column][/vc_row]

Fecha de creación
19 Nov 2017
Autor
flooming