**Clase 2 – Segunda parte**

Bienvenidos y bienvenidas a la segunda parte de la clase 2 del curso de testing funcional y accesibilidad web.

**¿Qué veremos en esta clase?**

Esta clase la dedicaremos a profundizar las tareas del tester, conocer sobre los procesos de diseño y planificación, para finalizar viendo diferentes estrategias de desarrollo en los equipos de trabajo.

El material lo encontrarán en Word (“doc”) para facilitar la navegación como también la edición, subrayado o resaltado. Si lo prefieren podrán descargar la clase también en formato “pdf”.

1. **Introducción**

Las pruebas de software (software testing) son técnicas cuyo objetivo es proporcionar información objetiva e independiente sobre la calidad del producto a los actores involucrados. Es una actividad más en el proceso de control de calidad.

Las pruebas son básicamente un conjunto de actividades dentro del desarrollo de software. Dependiendo del tipo de pruebas, estas actividades podrán ser implementadas en cualquier momento de dicho proceso de desarrollo.

Existen distintos modelos de desarrollo de software, así como modelos de pruebas. A cada uno corresponde un nivel distinto de involucramiento en las actividades de desarrollo.

El objetivo de las pruebas es presentar información sobre la calidad del producto a las personas responsables de este. Las pruebas de calidad presentan los siguientes objetivos: encontrar defectos o bugs, aumentar la confianza en el nivel de calidad, facilitar información para la toma de decisiones, evitar la aparición de defectos.

Teniendo esta afirmación en mente, la información que puede ser requerida es de lo más variada. Esto hace que el proceso de testing sea completamente dependiente del contexto en el que se desarrolla.

El ambiente ideal de las pruebas es aquel que es independiente del desarrollo del software, de esta manera se logra objetividad en las pruebas.

1. **El proceso de testing**

La ejecución de pruebas es sólo una parte del proceso de las pruebas. Dependiendo del enfoque seleccionado el proceso de pruebas se realizará en diferentes puntos del proceso de desarrollo. Es decir, además del proceso de desarrollo, tenemos un proceso para el testing.

Si bien tenemos diferentes fases en este proceso, estas fases pueden superponerse y la fase de control se realiza en todas las fases para poder saber en cada momento en qué estado nos encontramos.

1. **Planificación y control**

El control de las pruebas es una actividad continua que influye en la planificación de las pruebas. El plan de pruebas maestro (master test plan) será modificado de acuerdo a la información adquirida a partir del control de prueba.

Para saber el estado del proceso de pruebas comparamos el progreso logrado con respecto al plan de pruebas. Se inician medidas correctivas y esto permite tomar decisiones.

Tareas del Control de Pruebas

- Medir y analizar los resultados de las pruebas: De esta manera tendremos conocimiento de la cantidad, el tipo y la importancia de los defectos que hemos encontrados.

- Monitorear y documentar el progreso: Esta tarea nos sirve para determinar cuántas pruebas se completaron, cuáles fueron los resultados, y qué riesgos se encontraron para evaluar.

- Brindar información de las pruebas a personas interesadas: Estos reportes les sirven a las personas interesadas para saber el estado de las pruebas y así poder tomar las acciones necesarias.

- Iniciar acciones correctivas dependiendo de lo que se necesite corregir: Dependiendo de lo que necesitemos corregir, es posible ajustar los criterios de salida para los defectos detectados, o priorizar defectos bloqueantes.

- Tomar decisiones que definimos si seguimos o paramos las pruebas.

Tareas de la Planificación de Pruebas

-Determinar el alcance y riesgos: Nos preguntamos si vamos a probar un software completo, un componente, o algún otro producto.

-Identificar los objetivos de las pruebas y los criterios de salida de pruebas: Qué es lo más importante en este caso? Prevenir defectos, verificar que el software cumple con los requerimientos, medir la calidad?

-Determinar el enfoque: Todas estas preguntas nos permiten definir el enfoque de nuestras pruebas. Cómo ejecutaremos las pruebas, qué técnicas usaremos, qué se probará y cuán extensamente, es decir la cobertura de pruebas, quienes van a participar en el equipo de pruebas y durante cuánto tiempo?

-Implementar la estrategia de pruebas, y planificación del período de tiempo para el desarrollo de las actividades a seguir.

-Adquirir recursos necesarios para las pruebas como personas, computadoras, software, entorno de pruebas, presupuesto de pruebas, fechas.

-Selección de condiciones de entrada y de salida: Una condición de entrada será, por ejemplo, que el entorno de pruebas está listo y estable, que las herramientas necesarias están instaladas, el equipo de pruebas está completo. Los criterios de salida pueden ser: que todos los casos de pruebas diseñados sean ejecutados, que se hayan ejecutado pruebas de regresión, que no existan defectos bloqueantes, que los defectos de criticidad mediana no superen un determinado número. Los criterios de salida son condiciones acordadas con la gente involucrada, para que el proceso sea considerado formalmente concluido.

Lo que se concluya en estas tareas se plasma en documentos que crearemos también en esta fase y que usaremos en todo el proceso. Los documentos son los siguientes:

-El documento de Plan de pruebas describe alcance, enfoque, recursos y calendario de las actividades de pruebas previstas.

-El documento de Estrategias de prueba describe a alto nivel los niveles de prueba a realizar.

-El documento de Enfoque de pruebas incluye análisis de riesgo, técnicas de diseño de pruebas a aplicar, criterios de salida y tipos de prueba a ejecutar.

1. **Análisis y diseño**

Las tareas más importantes:

- Revisión de elementos: Se revisan los elementos básicos para las pruebas (requerimientos, arquitectura, especificaciones de diseño, interfaces). Utilizamos estas bases para poder comenzar con el diseño de pruebas.

- Testeabilidad: Evaluación de si los elementos básicos del punto anterior pueden generar casos de prueba.

- Condiciones de Pruebas: Identificación y revisión de las condiciones de las pruebas basándonos en los puntos anteriores: Esto nos brinda una lista a alto nivel de lo que nos interesa probar.

- Diseño de Casos: tanto los positivos (que dan muestra de la funcionalidad en sí) como los negativos (comprueban situaciones en las que hay tratamiento de errores).

- Entorno: Esta actividad trata sobre la puesta a punto del entorno de pruebas: disponibilidad del entorno, administración de usuarios, carga de datos.

- Herramientas: Seleccionar, proveer e instalar las herramientas de pruebas, procesos, procedimientos y responsabilidades.

1. **Implementación y Ejecución**

En esta etapa realizamos la ejecución de las pruebas, manual o mediante el uso de alguna herramienta:

- Se finalizan, implementan y priorizan los casos de prueba y los procedimientos de prueba.

- Trazabilidad: Se verifica que se pueda realizar la trazabilidad entre los elementos básicos de prueba y los casos de prueba.

- Ejecución: Se ejecutan los casos de manera automática o manual.

- Registro de los resultados: Una vez que hemos ejecutado, ya sea manual o automáticamente, los casos de pruebas, entonces podemos registrar los resultados obtenidos, esto incluye: identidades y versiones del software, las herramientas de pruebas utilizados, los productos de soporte de pruebas y también incluye comparar los resultados reales con los resultados que eran esperados.

- Comparación: Se comparan los resultados obtenidos con los resultados esperados.

- Informe: Se informan discrepancias en un defecto para que sea arreglado.

- Repetición: Se repiten las pruebas para confirmar que el defecto ha sido corregido (re testing). También incluye realizar pruebas de regresión para asegurarnos que los cambios nos han introducido nuevos defectos (**testing de regresión**)

1. **Evaluación de criterio de salida y reportes**

Los resultados de las pruebas son evaluados contra los objetivos definidos.

Tareas:

- Comparación: Comparar los registros de prueba contra los criterios de salida especificados en la fase de planificación de prueba. Es decir, evaluamos la evidencia que se tiene de las pruebas que se ejecutaron, fallas reportadas, solucionadas o pendientes para así confirmar si se completaron los criterios de salida que dan fin a las pruebas.

- Evaluación: Con resultados de la tarea anterior, se evalúa si se necesitan ejecutar más pruebas o si el criterio de salida debe modificarse. Es decir, aquí definimos si ejecutamos más pruebas en caso de que no se hayan ejecutad todos los casos de prueba diseñados; definimos si continuamos con la ejecución de las pruebas si es que no se alcanzó la cobertura deseada; o definimos que necesitamos diseñar más casos de prueba en caso de que nuevos riesgos se han descubierto.

- Reporte: Lo que hacemos aquí es preparar un resumen de las pruebas ejecutadas para que las partes interesadas estén al tanto de nuestro trabajo y de nuestro avance.

1. **Actividades de Cierre - Test closure**

Tareas:

- Recolección: Recolectar información sobre las pruebas completadas.

- Verificación: Verificamos que las pruebas y la documentación acordadas hayan sido entregadas al cliente según lo definido en el plan de pruebas.

- Documentación: Se deben documentar todos los casos de pruebas corridos, todos los resultados, todos los bugs que se hayan encontrados, vamos a documentar hasta la aceptación del sistema.

- Analizar lecciones aprendidas para futuros proyectos. Entre ellos, mejoras a los procesos del ciclo de desarrollo de software. También verificaremos dónde hubo problemas y el número de fallas para realizar mejoras de diseño, ejecución y revisión de las pruebas.

1. **El proceso de desarrollo de software**

El proceso de desarrollo puede involucrar numerosas y variadas tareas, desde lo administrativo, pasando por lo técnico y hasta la gestión:

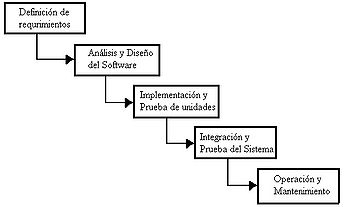
* Análisis de requisitos
* Diseño
* Codificación
* Pruebas (unitarias y de integración)
* Instalación y paso a producción
* Mantenimiento

Para cada una de las fases o etapas listadas, existen sub-etapas (o tareas). El modelo de proceso o **modelo de ciclo de vida** utilizado para el desarrollo, define el orden de las tareas o actividades involucradas,​ también define la coordinación entre ellas, y su enlace y realimentación. Entre los más conocidos se puede mencionar: modelo en cascada o secuencial, modelo espiral, modelo iterativo incremental.

Los diferentes modelos de **ciclo de vida** de desarrollo necesitan diferentes enfoques hacia la prueba. Los modelos de desarrollo software son utilizados para el desarrollo de software incluyendo las actividades del proceso de pruebas. Las pruebas no existen de manera aislada, están relacionadas con las actividades de desarrollo de software.

Desarrollo en cascada

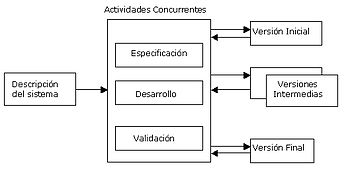
Se denomina de este modo, ya que a cada salida de una etapa cae en la siguiente, es decir, las etapas se llevan a cabo una a continuación de la otra. Una de las características de este proceso, es que no está previsto volver a una etapa anterior, es decir si se olvidó relevar algún requerimiento al comienzo, no tiene una alternativa para considerar este caso. Este proceso supone cada etapa independiente de las etapas anteriores.



Desarrollo iterativo y creciente

Se tiene las mismas etapas que en el proceso de desarrollo en cascada, sin embargo, en este proceso, la etapa de relevamiento se divide en distintos subconjuntos, y cada uno de estos subconjuntos se construye de la misma forma que con el ciclo de vida en cascada. Se van desarrollando por partes que luego se integran, una vez finalizadas las mismas.

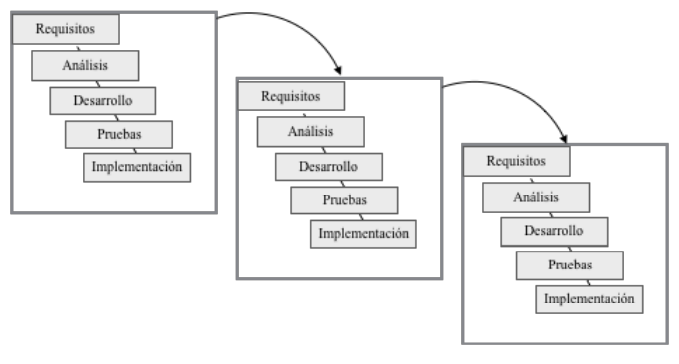
El desarrollo iterativo posee las mismas etapas de desarrollo que los procesos anteriores, pero trabajamos sobre el todo, no necesariamente conocemos al comienzo todos los detalles del producto que queremos construir.



Un modelo Iterativo busca reducir el riesgo que existe entre lo que el usuario necesita y el producto final.

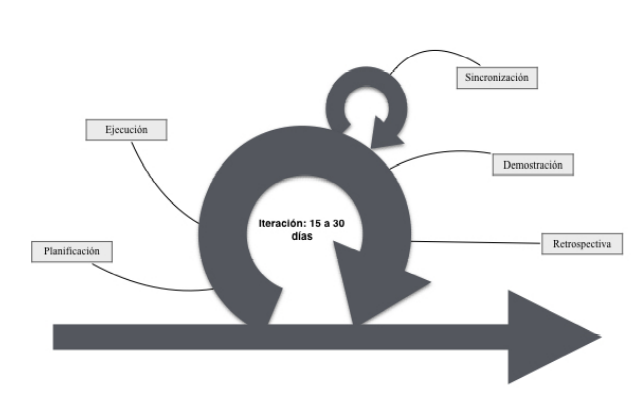
**ITERACION:** es una secuencia de actividades dentro de un plan, que se organiza con el objetivo de entregar parte de funcionalidad del producto. Es decir, en la primera iteración, le entrego al cliente parte de la funcionalidad, en la segunda iteración le entrego una mejora (de la misma funcionalidad o mayor funcionalidad).

Cada iteración contribuye con una característica adicional del sistema a desarrollar. Cada una de las iteraciones puede ser probada por separado. En cada iteración, la verificación (relación con el nivel precedente) y la validación (grado de corrección del producto dentro del nivel actual) se pueden efectuar por separado.



Por último, tenemos el desarrollo ágil de software, este es un proceso iterativo e incremental, se caracteriza por contar con iteraciones cortas y por no tener fases lineales, tipo cascada en cada iteración. Existen distintas metodologías Ágiles, que entre las más conocidas y utilizadas encontramos "Scrum" y "XP: Extreme Programming".

En esta metodología ágil se realizan entregas parciales del producto final. Está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos.



Un proyecto se ejecuta en iteraciones de aproximadamente de 15 días a 1 mes de duración. Cada una de ellas tiene que proporcionar un resultado completo, un incremento de producto final.

Empezamos con la lista de requerimientos priorizada o Product Backlog (el cliente es quien va definiendo la priorización de acuerdo con lo que aportarían)

**Sprint Planning:** Es la reunión que se realiza el primer día de la iteración, la cual cuenta con 2 partes importantes: la selección de requisitos y la planificación propiamente dicha.

**Sprint (ejecución):** A partir de este momento, ya se comienza a trabajar. Todos los días, todo el equipo se toma 15 minutos para realizar entre todos una reunión de sincronización. Esta reunión, o scrum meeting se realiza para que cada miembro del equipo sepa en que están trabajando los demás miembros. Cada uno, responderá a 3 preguntas: Qué he hecho desde la última reunión? Qué voy a hacer a partir de este momento? Qué impedimentos tengo o voy a tener?

**Sprint Review (Demostración):** El último día de la iteración, el equipo le presenta al cliente los requisitos completados. Aquí el cliente le puede exponer cambios o no.

**Sprint Retrospective (Retrospectiva):** El equipo analiza cómo fue su trabajo y cuáles son los problemas que podrían impedirle progresar, para así realizar los cambios necesarios.

|  |
| --- |
| **¿Qué es el ciclo de vida del lanzamiento de un software?**  El ciclo de vida del **lanzamiento** de un software es el conjunto de estados del progreso de los proyectos de creación de aplicaciones informáticas, para poder identificar cuánto se ha avanzado y cuánto queda hasta el final.  Cada versión importante de un producto pasa generalmente a través de una etapa en la que se agregan las nuevas características (etapa alfa), después una etapa donde se eliminan errores activamente (etapa beta), y finalmente una etapa en donde se han quitado todos los errores importantes (etapa estable). Las etapas intermedias pueden también ser reconocidas. |

**¡Felicitaciones! ¡Llegaste al final de la clase! ¡Nos vemos en el encuentro sincrónico!**