Pruebas del Software: Fundamentos

MSc. Juan José Quesada Sánchez
ajuanjo@gmail.com

Agenda

- El rol del testing
- Actividades de evaluación de la calidad del software
  - Pruebas Funcionales, Pruebas Estructurales, Análisis dinámico, Análisis estático
- Verificación y Validación
- Conceptos
  - Failure (falla), error (error), fault (defecto)
- Flujo de trabajo general de la disciplina
- Niveles de Pruebas: Modelo V
- Caso de Prueba (Artefacto)

El rol del testing

El rol del testing

- Esta disciplina actúa como un proveedor de servicios para otras disciplinas
- Se centra en la evaluación o la valoración de la calidad del producto por medio de las siguientes prácticas:
  - buscar y documentar defectos en la calidad del software
  - opinar sobre la calidad percibida del software
  - validar y demostrar las suposiciones efectuadas en las especificaciones de diseño y requisitos a través de demostraciones concretas

El rol del testing

- Prácticas de la disciplina(2)
  - validar que el producto de software funciona según el diseño
  - validar que los requisitos se han implementado de forma adecuada.

Actividades de la evaluación de la calidad del software

Actividades de la evaluación de la calidad del software

- **Prueba Funcional (caja negra)**
  - Prueba el software sin conocer su funcionamiento interno (cómo está construido o qué contiene)
  - El sistema bajo prueba es tratado como una caja negra
  - Los verificadores no requieren conocer la implementación del sistema
  - Los casos de prueba son creados a partir de los requerimientos

Actividades de la evaluación de la calidad del software

- **Prueba Estructural (caja blanca)**
  - Se enfoca en la estructura interna del sistema bajo prueba
  - Las rutas a través de la aplicación son identificadas y probradas
  - Se requiere conocer el lenguaje de programación en el que está implementado el sistema

Actividades de la evaluación de la calidad del software

- **Análisis estático**
  - Basado en la revisión de documentos
  - Requerimientos, modelos de software, documentos de diseño y código fuente
  - Incluye tareas como revisión del código, inspección, análisis algorítmico, ir-a-través-de y pruebas de exactitud

Actividades de la evaluación de la calidad del software

- **Análisis dinámico**
  - Involucra la ejecución de la aplicación de software para exponer posibles fallas
  - Las propiedades de ejecución y rendimiento también son observadas
  - Tanto los casos de prueba como los valores de entrada de la aplicación siendo probada son seleccionados con cuidado

Verificación y Validación

Verificación y Validación

- **Verificación**

¿Estamos construyendo correctamente el producto?

  - Permite evaluar si el producto de alguna fase de desarrollo satisface los requerimientos establecidos al inicio de esa fase
  - ¿El software está constituido de acuerdo con su especificación?
- Revisa productos intermedios: especificaciones de requerimientos, diseño, código, manuales de usuario

Verificación y Validación

- **Validación**

¿Estamos construyendo el producto correcto?

  - Las actividades de validación apuntan a confirmar que el producto cumple con las expectativas del cliente
  - Se enfoca en el *producto final (el sistema entero)* el cual es testeado extensivamente desde el punto de vista del cliente
- *Ejemplo:*
  - *En la metodología XP, el cliente interactúa con el equipo de desarrollo y conduce pruebas de aceptación durante cada iteración*

Verificación y Validación

- La verificación establece la correspondencia de una fase de implementación con su especificación
- La validación establece la correspondencia entre el sistema y las expectativas del usuario

Conceptos

Conceptos

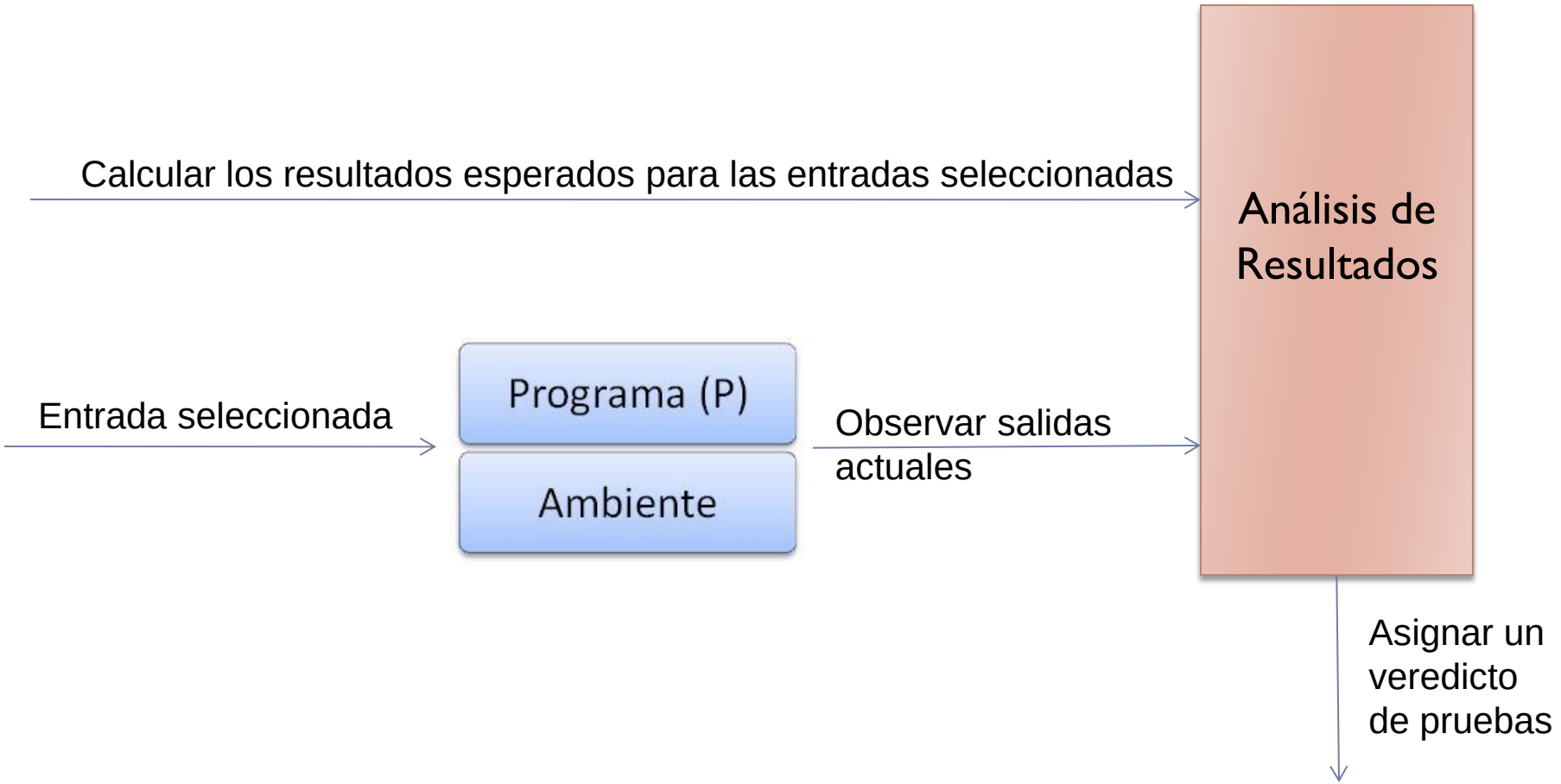
- **Failure** (falla): incapacidad de un sistema o componente para ejecutar una función requerida en acorde con sus especificaciones
  - La falla ocurre si el comportamiento del software es distinto del especificado
  - Puede ser que la falla no pueda ser observada o detectada
- **Error** (error): es un estado del sistema
  - En la ausencia de una acción correctiva del sistema, un estado de error puede conducir a una falla no atribuible a cualquier evento subsecuente al error

Conceptos

- **Fault** (defecto): la causa adjudicada de un error
  - En el argor de los ingenieros se le conoce como "una pulga" o "bug"
- El proceso de manifestación de una falla se puede representar mediante una cadena de comportamiento
  - defecto → error → falla
  - La cadena puede iterar en un ciclo

Flujo de trabajo general de la disciplina

Flujo de trabajo general de la disciplina



Flujo de trabajo general de la disciplina

- Secuencia de actividades ejecutadas por el ingeniero de pruebas o verificador
- **Identificar el objetivo de la prueba:**
  - Define la intención o propósito de generar uno o más casos de prueba para asegurar que el programa lo soporte
- **Seleccionar las entradas**
  - Pueden basarse en: especificación de requerimientos, código fuente o nuestras expectativas
  - Considerar el objetivo de la prueba

Flujo de trabajo general de la disciplina

- **Calcular los resultados esperados**
  - Deben estar basadas en las entradas seleccionadas
- **Configurar el ambiente de ejecución del programa**
  - Cubrir asunciones externas al programa. Ejemplos
  - Inicializar el sistema local: conexión a la red, acceso a la b.d. correcta
  - Inicialización remota: procesos socios remotos en un ambiente distribuido

Flujo de trabajo general de la disciplina

- **Ejecutar el programa**
  - El verificador ejecuta el programa con las entradas seleccionadas y observa las salidas generadas
  - Se debe valorar la ubicación de las entradas seleccionadas
- **Analizar el resultado de pruebas(1)**
  - Comparar los resultados obtenidos versus los resultados esperados
  - Asignar un veredicto: *falló, pasó, inconcluso*

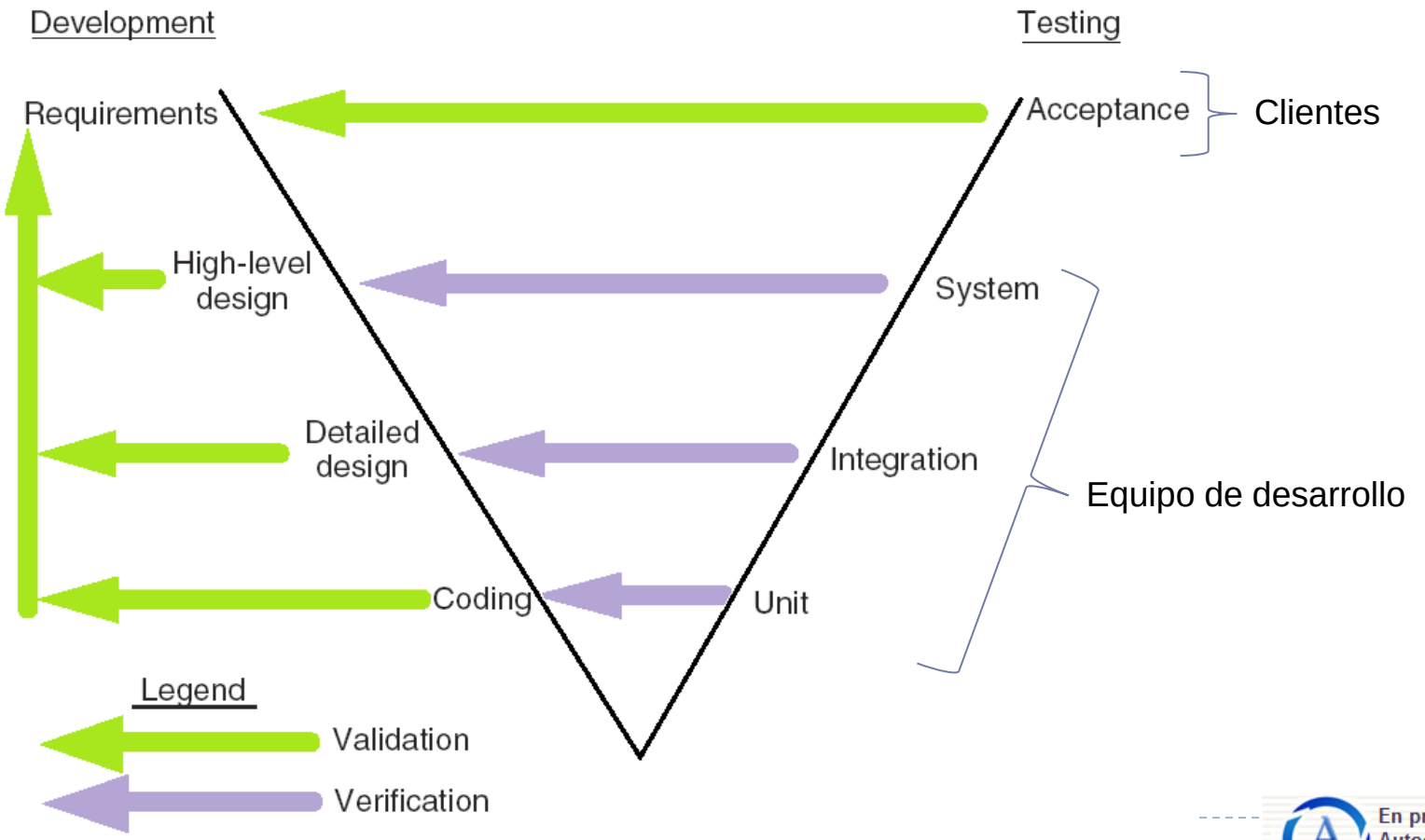
Flujo de trabajo general de la disciplina

- **Analizar el resultado de pruebas(2)**
  - Escribir un reporte para corregir el defecto si ocurrió una falla
    - ✓ explicar como reproducir la falla
    - ✓ describir la falla
    - ✓ referencia a los resultados obtenidos y al caso de prueba

Niveles de Pruebas: Modelo V

Niveles de Pruebas: Modelo V

- Las pruebas son ejecutadas en diferentes niveles del sistema completo o de sus partes en el ciclo de vida



Niveles de Pruebas: Modelo V

1.Pruebas de Unidad/Componente

Programadores prueban unidades de programa en aislamiento

(Procedimientos, funciones, métodos, clases)

Una vez realizadas, los subsistemas son ensamblados con técnicas de integración

2.Pruebas de Integración

Prueba interfaces entre unidades (componentes), interacciones con entidades externas(sistemas operativos, sistema de archivos, hw.) e interfaces con otros sistemas

Niveles de Pruebas: Modelo V

3.Pruebas de Sistema

Incluyen pruebas de: funcionalidad, seguridad, robustez, cargado, estabilidad, de estrés, performance y fiabilidad

Fase crítica cerca del final de la fecha de entrega

4.Pruebas de Aceptación

Ejecutadas por el cliente

Miden la calidad del software en lugar de buscar defectos

Caso de Prueba (Artefacto)

- Define un conjunto de entradas de prueba, condiciones de ejecución, y resultados esperados, identificados con el objetivo de evaluar algunos aspectos particulares de un elemento del destino de la prueba.
- Responsable: Analista de pruebas
- Especifica y comunica las condiciones específicas que deben validarse para habilitar una valoración de algunos aspectos concretos de los elementos del destino de la prueba.
- Habitualmente incluirán un subconjunto de requisitos como casos de uso, características de rendimiento y los riesgos que afectan al proyecto.

Caso de Prueba (Artefacto)

- Esquematización
  - **Descripción del caso de prueba:** Descripción de la finalidad u objetivo de la prueba, el ámbito y las condiciones previas de la prueba.
  - **Condición de ejecución:** Descripción de una condición que ejercerá durante esta prueba.
    - **Condiciones previas:** describa el estado necesario en el que debe encontrarse el sistema antes de que se pueda iniciar la prueba.
    - **Entradas de prueba:** enumere una lista de los estímulos específicos que deben aplicarse durante la prueba. Incluyen los objetos o campos con los que se interactúa y los valores de datos específicos introducidos al ejecutar este caso de prueba.
    - **Puntos de observación:** Durante la ejecución de la prueba, enumere las observaciones específicas que deben realizarse.
    - **Puntos de control:** Durante la ejecución de la prueba, identifique los puntos en los que el flujo de control puede alterarse o variar.
    - **Resultados esperados:** El estado resultante o las condiciones observables que se esperan como resultado de haber ejecutado la prueba. Tanto las respuestas positivas como las negativas.
    - **Condiciones posteriores**
      Para cada condición de ejecución, describa el estado necesario al que se debe devolver el sistema con el fin de que puedan llevar a cabo las pruebas subsiguientes.

Referencias

- Mena, A. (2011). Curso IF6100 Análisis y Diseño de Software. Universidad de Costa Rica
- [Jalote 2008] Jalote, Pankaj . **A Concise Introduction to Software Engineering**. Springer-Verlag London Limited, UK, 2008
- [Manassis 2003] Manassis, Enricos. **Practical Software Engineering: Analysis and Design for the .NET Platform** Addison Wesley, 2003 – *Cap. 9 Testing*
- [Naik & Priyadarshi 2008] Naik, Kshirasagar; Tripathy, Priyadarshi. ***Software testing and quality assurance: theory and practice***. *John Wiley & Sons, USA, 2008 - Cap. 1 Basic Concepts and Preliminaries*
- [RUP 2003] **Rational Unified Process® (RUP®) . Versión 7.0.1** IBM Corporation.