



UNIVERSIDAD  
Popular del cesar

# Ingeniería de Sistemas



# Ingeniería de Software II

## **Unidad No. 1** **Pruebas de Software** **Diseño de pruebas**

# Flujo de actividades del proceso de prueba

- Sumario de Evaluación de Pruebas (Test Evaluation Summary)
- Requerimientos de Cambio (Change Request).

**EVALUAR LAS PRUEBAS**

**EJECUTAR LAS PRUEBAS (PRUEBAS DEL SISTEMA)**

• Resultado de Pruebas

• Plan de Pruebas

**PLANIFICAR LAS PRUEBAS**

**ETAPAS DE PROCESO DE PRUEBAS**

**EJECUTAR LAS PRUEBAS (INTEGRACIÓN DE PRUEBAS)**

• Resultado de Pruebas

- Modelo de Pruebas (Test Model)
- Casos de Prueba (Test Case)
- Procedimientos de Prueba (Test Procedures)
- Documento de Análisis de Carga de Trabajo (Workload Analysis Document)

**DISEÑAR LAS PRUEBAS**

**IMPLEMENTAR LAS PRUEBAS**

- Script de la Prueba
- Componente de la Prueba

# Flujo de actividades del proceso de prueba

## Planificar las Pruebas:

- En esta fase se describe la estrategia de prueba, se estiman los requisitos y se planifica el esfuerzo de la prueba
- El principal artefacto producido es el Plan de Pruebas.
- Se planifican pruebas personalizando los estándares específicamente para el proyecto.
- Se definen niveles de pruebas a aplicar.
- Se especifican las técnicas a utilizar.
- Se establece:
  - El tiempo para la ejecución de cada una de las pruebas.
  - El uso de herramientas.
  - Los criterios de aceptación.
  - Los recursos involucrados.

# Flujo de actividades del proceso de prueba

## Diseñar las Pruebas:

- Identificar casos de prueba y procedimientos de prueba
  - Diseñar casos de prueba de integración (para verificar que los componentes interaccionan correctamente)
  - Diseñar la prueba del sistema (para verificar que el sistema funciona correctamente como un todo)
  - Diseñar los casos de prueba de regresión
    - Al añadir un nuevo módulo puede haber problemas con módulos que antes iban bien.
    - Las pruebas de regresión son un conjunto de pruebas (ya realizadas antes) que aseguran que los cambios no han dado lugar a cambios colaterales
- Los principales artefactos producidos son:
  - El Modelo de Pruebas (Test Model)
  - Los Casos de Prueba (Test Case)
  - Los Procedimientos de Prueba (Test Procedures)
  - Documento de Análisis de Carga de Trabajo (Workload Analysis Document).

# Flujo de actividades del proceso de prueba

## Implementar las Pruebas:

- Automatizar los procedimientos de prueba, creando componentes de prueba.
- Los principales artefactos producidos son el Script de la Prueba y el Componente de la Prueba.

## Ejecutar las Pruebas en la etapa de Integración de Pruebas:

- Realizar las pruebas, comparar con los resultados esperados e informar de los defectos
- El principal artefacto producido es el documento Resultado de Pruebas.

## Ejecutar las Pruebas en la etapa de Pruebas del Sistema:

- Se comienzan después de las de integración y se realizan de manera análoga (realizar, comparar e informar)
- El principal artefacto producido es el documento Resultado de Pruebas.



# Flujo de actividades del proceso de prueba

## Evaluar las Pruebas:

- Se comparan los resultados de las pruebas con los objetivos esbozados en el plan de prueba. Hay que preparar métricas que permitan determinar el nivel de calidad del software y la cantidad de pruebas a realizar.
- Los principales artefactos producidos son el Sumario de Evaluación de Pruebas (Test Evaluation Summary) y los Requerimientos de Cambio (Change Request).

# DISEÑO DE CASOS DE PRUEBA

- Para el diseño de las pruebas, se tendrán en cuenta aspectos que permitirán encontrar defectos en el periodo de desarrollo del software, la realización de pruebas propias de verificación y validación de datos.

Pruebas de caja blanca

- **Encontrar casos de prueba “viendo” el código interno**

Pruebas de caja negra


- **Encontrar casos de prueba “viendo” los requisitos funcionales**





# DISEÑO DE CASOS DE PRUEBA

**Espacio de Prueba:** Conjunto de todos los posibles casos de Prueba.

- **Pruebas Funcionales:** Considerar solamente entradas válidas al sistema y condiciones normales de operación.
  - **Pruebas de Robustez:** Considerar datos de entrada inválidos, secuencias invalidas de comandos/acciones, etc.
  - **Pruebas de Frontera:** Considerar valores/tamaños mínimos y máximos para datos de entrada, carga del sistema mínima y máxima, etc.
- 

# DISEÑO DE CASOS DE PRUEBA

- **Pruebas de tolerancia a fallas:** Considerar condiciones anormales de operación, fallas hardware y software de la plataforma computacional sobre la que funciona el software en prueba



# DISEÑO DE CASOS DE PRUEBA

**Por cada propósito de prueba, hacer una lista de casos de prueba**

- Construir una versión preliminar de la lista de casos de prueba a partir de los escenarios típicos relacionados con el propósito de prueba
- Enriquecer la lista de casos de prueba, analizando las posibles variaciones o casos especiales de los escenarios considerados
- Complementar la lista de casos de prueba, analizando posibles casos que puedan revelar errores



Clases de Equivalencia				
Clases	Condición Entrada	Clase Equivalencia	Test	Id Test
1	Costo del proyecto	Valor positivo > 0.00	Prueba con entrada costo = 150000.00	1
2		Valor cero (0)	Prueba con entrada costo = 0.00	2
3		Valor < 0	Prueba con entrada costo = - 1000.00	3
Decisiones / Condiciones				
Condición	Lógica	Condición a probar	Test	
1	Costo >= 100000.00	true	Prueba con entrada costo = 150000.00	1
		false	Prueba con entrada costo = 60000.00	4
2	Costo < 100000.00	true	Prueba con entrada costo = 60000.00	4
		false	Prueba con entrada costo = 150000.00	1
3	Costo >= 50000.00	true	Prueba con entrada costo = 150000.00	1
		false	Prueba con entrada costo = 10000.00	5
Valores Límites				
Límite	Valor	Condición a probar	Test	
1	100000.00	=	Prueba con entrada costo = 100000.00	6
		>	Prueba con entrada costo = 100001.00	1
		<	Prueba con entrada costo = 99999.00	4
2	50000.00	=	Prueba con entrada costo = 50000.00	7
		>	Prueba con entrada costo = 50001.00	1
		<	Prueba con entrada	5



# Prueba de unidad de EsPrimo

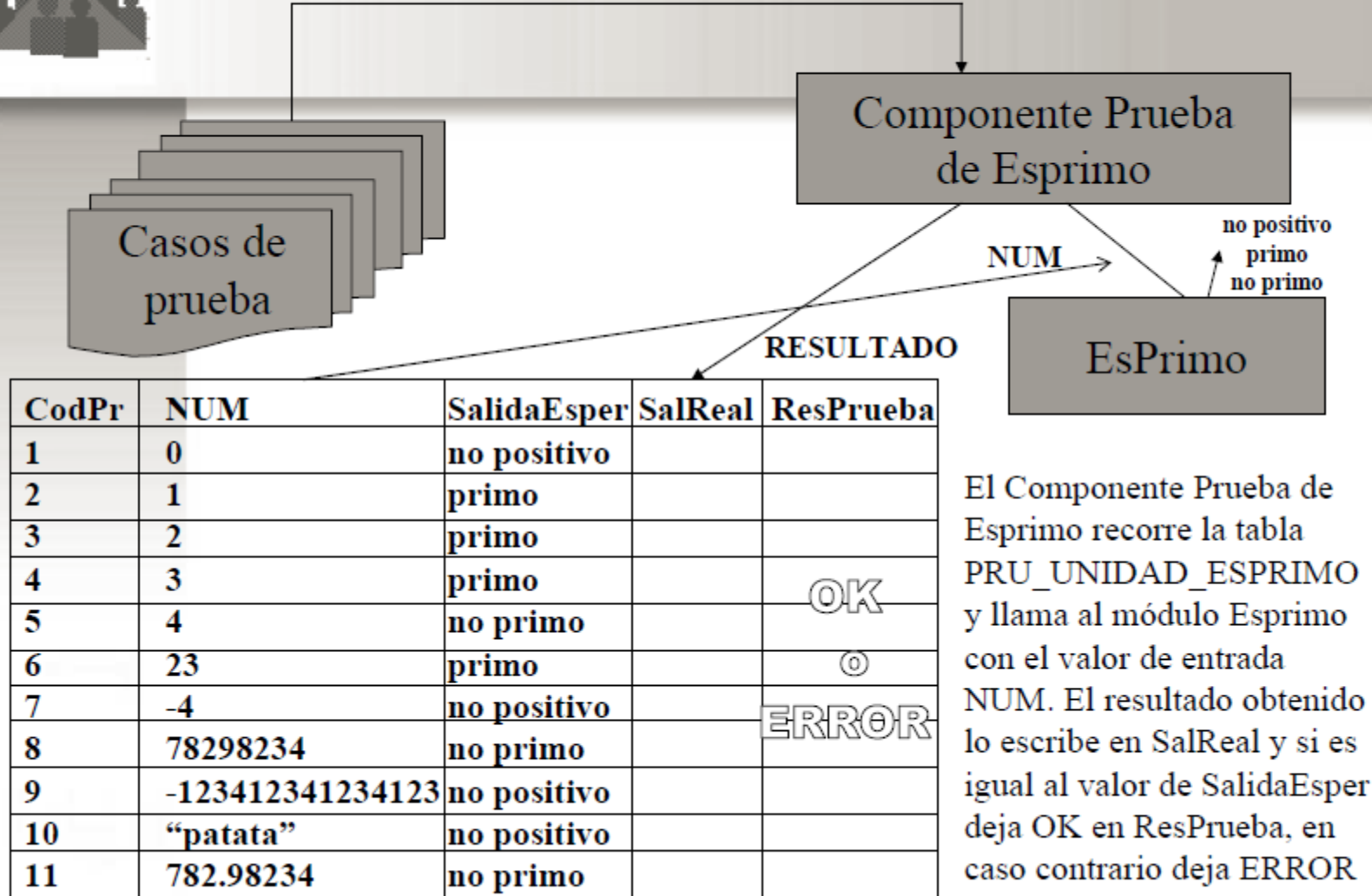


Tabla PRU UNIDAD ESPRIMO

El Componente Prueba de Esprimo recorre la tabla PRU\_UNIDAD\_ESPRIMO y llama al módulo Esprimo con el valor de entrada NUM. El resultado obtenido lo escribe en SalReal y si es igual al valor de SalidaEsper deja OK en ResPrueba, en caso contrario deja ERROR

A	B	C	D
Condición que se evalúa	Condición de entrada	Dato de entrada	Salida esperada
IF-1 (T)	args.length==0	args={}	Excepción ErrorFaltaparametro
IF-1(F) y IF-2(T)	args.length>1	args={4,12}	Excepción ErrorSolo1Parametro
IF-1 (F) y IF-2(F) y IF-3(T)	num<0	num=-4	Excepción ErrorNoNumeroPositivo
IF-1(F) y IF-2(F) y IF-3(F) y bucle 0 veces	num=2	num=2	primo
IF-1(F) y IF-2(F) y IF-3(F) y bucle 1 vez (IF4(F))	num=3	num=3	primo
IF-1(F) y IF-2(F) y IF-3(F) y bucle 2 veces (IF4(T))	num=4	num=4	no primo
IF-1(F) y IF-2(F) y IF-3(F) y bucleN veces	num>4	num=34	primo
Excepcion ErrorNoNumeroPositivo	tipo args[0] no número	args[0]="patata"	Excepción ErrorNoNumeroPositivo



Herramienta	Lenguaje
Coverage Pyton	Pyton
PHPUnit	PHP
JUnit	JAVA
CodeCover	Java, Cobol
JsCoverage	JavaScript
Ncover	Microsoft .Net

# Frameworks para pruebas unitarias



## Java

Para Java se cuenta con [Junit](#)

## PHP

Para PHP dispone de [PHPUnit](#)

## Javascript

Para Javascript, existe una mayor variedad de frameworks, entre los que podría destacar [MochaJS](#) o [JEST](#).

En el caso de MochaJS, trabaja tan bien con el front end como con el back end.

Mientras que JEST, destaca por ser sencillo.

## C#

[NUnit](#) es el framework Open source de pruebas unitarias para .NET más conocido. Las pruebas pueden realizarse tanto desde la consola, como con Visual Studio, o a través de terceros.



**UNIVERSIDAD**  
Popular del cesar