

### Leyenda

Primera columna	Técnicas de Verificación
Segunda y tercera columna	Técnicas de Validación

### Walkthrough

Revisión informal de documentos o código con el equipo

**Fases** Diseño y codificación

**Fortalezas** Retroalimentación rápida, útil para compartir conocimiento

**Debilidades** Poco formal y depende de la experiencia del equipo

### Code inspections

Detectar errores y mejorar la calidad mediante revisiones detalladas del código

**Fases** Codificación

**Fortalezas** Alta precisión para encontrar defectos

**Debilidades** Requiere tiempo y personal capacitado

### Reviews

Evaluar la documentación y código en reuniones formales

**Fases** Diseño y codificación

**Fortalezas** Mejora la calidad y la comprensión del proyecto

**Debilidades** Puede ser costosa y depende de la disponibilidad del equipo

### Pruebas Formales

Verificar la corrección matemática del código

**Fases** Diseño y codificación

**Fortalezas** Mejora la calidad y la comprensión del proyecto

**Debilidades** Puede ser costosa y depende de la disponibilidad del equipo

### Black Box Testing

Evaluar la funcionalidad sin ver el código

**Fases** Fase de pruebas funcionales

**Fortalezas** No requiere conocimiento interno del sistema

**Debilidades** No detecta problemas estructurales

### White Box Testing

Evaluar la estructura interna del código

**Fases** Fase de pruebas de unidad y de integración

**Fortalezas** Permite identificar errores internos y optimizar el código

**Debilidades** Requiere acceso al código y conocimientos técnicos avanzados

### Heuristic Testing

Detectar fallos usando la experiencia y el conocimiento del sistema

**Fases** Fase de pruebas de sistema

**Fortalezas** Útil para identificar problemas difíciles de prever

**Debilidades** Depende de la habilidad del probador

### Interface Testing

Validar la interacción entre módulos o sistemas

**Fases** Fase de pruebas de integración

**Fortalezas** Asegura la correcta interacción de componentes

**Debilidades** Puede ser difícil simular todos los escenarios de interfaz

### Boundary Value Analysis

Prueba los valores en los límites de los rangos de entrada

**Fases** Pruebas de sistema

**Fortalezas** Detecta errores en límites de entrada

### Boundary Value Analysis (cont)

**Debilidades** No cubre casos intermedios

### Equivalence Class Partitioning

Divide los datos de entrada en clases que se consideran equivalentes

**Fases** Pruebas de sistema

**Fortalezas** Reduce casos de prueba

**Debilidades** Omite errores fuera de clases

### Decision Table-Based Testing

Usa tablas para representar combinaciones de condiciones y acciones

**Fases** Diseño de pruebas, pruebas de requisitos

**Fortalezas** Maneja lógica compleja

**Debilidades** Tablas grandes difíciles de gestionar

### Cause Effect Graphing

Crea un grafo para mostrar relaciones lógicas entre condiciones y efectos

**Fases** Análisis y diseño de pruebas

**Fortalezas** Visualiza condiciones y efectos

**Debilidades** Difícil con requisitos ambiguos

### References

G. R. Maquieira, "Qué es black box testing o pruebas de caja negra", Openwebinars.net, 06-ene-2023.

Edu.uy. [En línea]. Disponible en: <https://www.fing.edu.uy/tecnoinf/maldonado/cursos/ingsoft/materiales/teorico/is09-Verificacion-Validacion.pdf>. [Consultado: 06-nov-2024].

Not published yet.

Last updated 6th November, 2024.

Page 1 of 1.

Sponsored by [ApolloPad.com](https://www.apollopad.com)

Everyone has a novel in them. Finish Yours!

<https://apollopad.com>