Control de Calidad

SDLC

¿QA sólo está en las pruebas?

1

SDLC

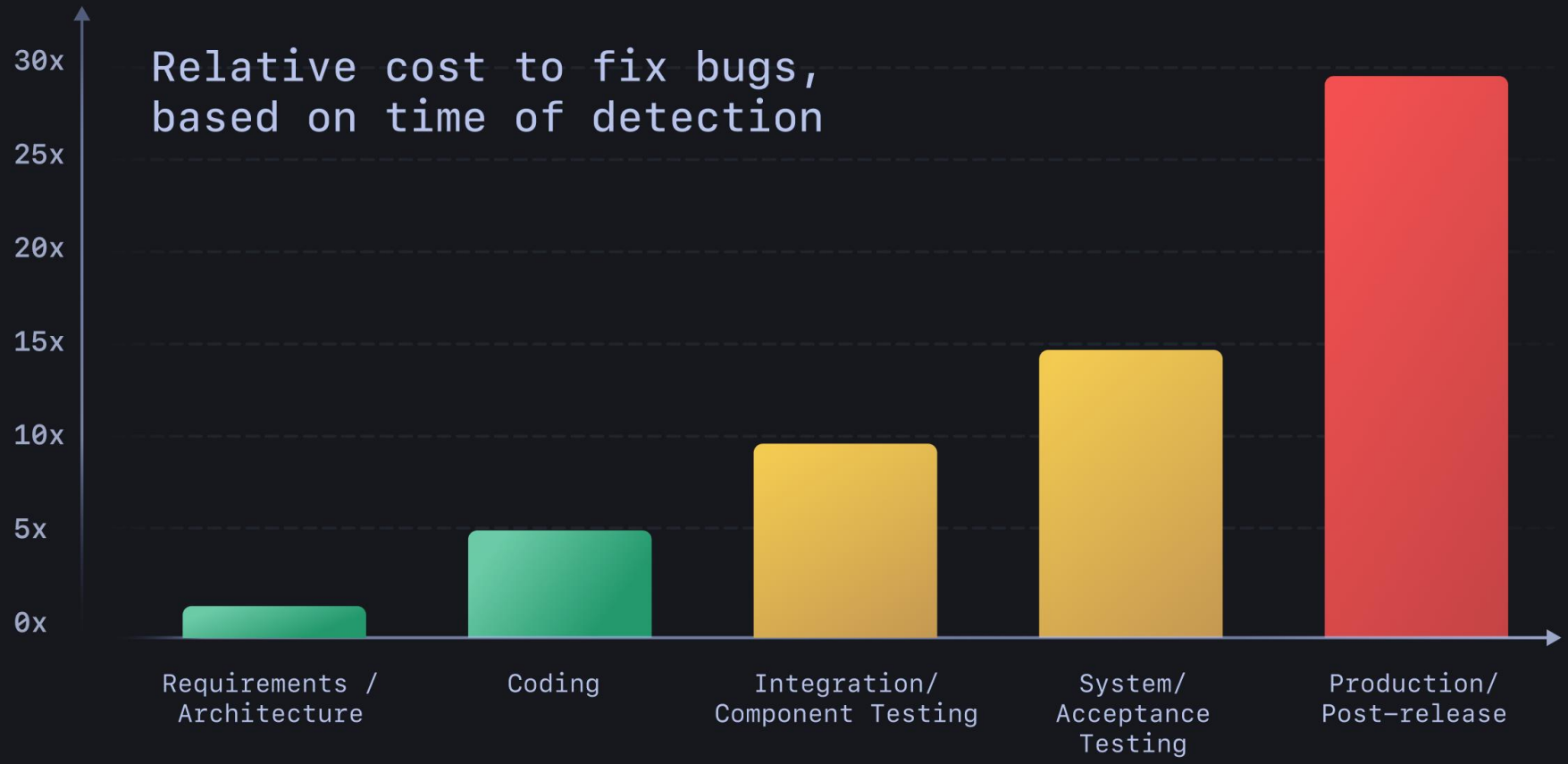
- Análisis de requerimientos
- Diseño
- Desarrollo
- Pruebas
- Implementación
- Mantenimiento

- **Comprensión de requerimientos**
- **Crear escenarios de pruebas**
- **Pruebas unitarias**
- **Funcionales, integración, seguridad, etc.**
- **Pruebas del cliente(UAT), pruebas en entorno de prod.**
- **Replicar posibles errores**

QA

¿Por qué es importante?

2


Relative cost to fix bugs,
based on time of detection
30x
25x
20x
15x
10x
5x
0x
Requirements / Architecture
Coding
Integration/ Component Testing
System/ Acceptance Testing
Production/ Post-release

Validación vs Verificación

- Verificación: que el sistema de información funcione correctamente como fue diseñado.
- Validación: que el sistema de información cumpla con los requerimientos funcionales.

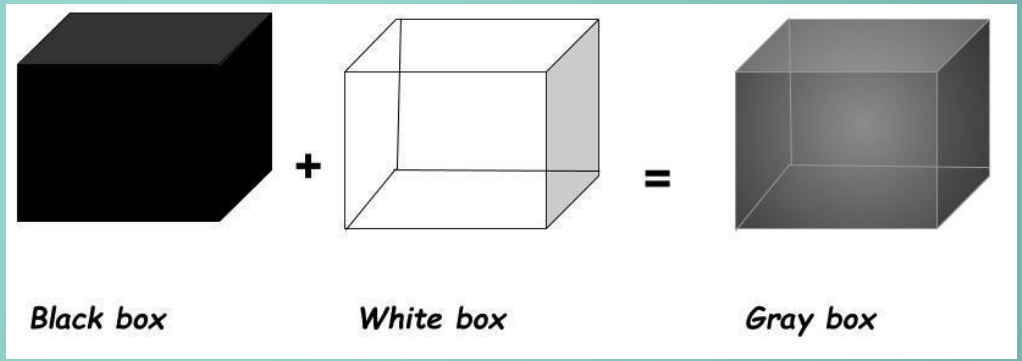
QA

Tipos de pruebas

3

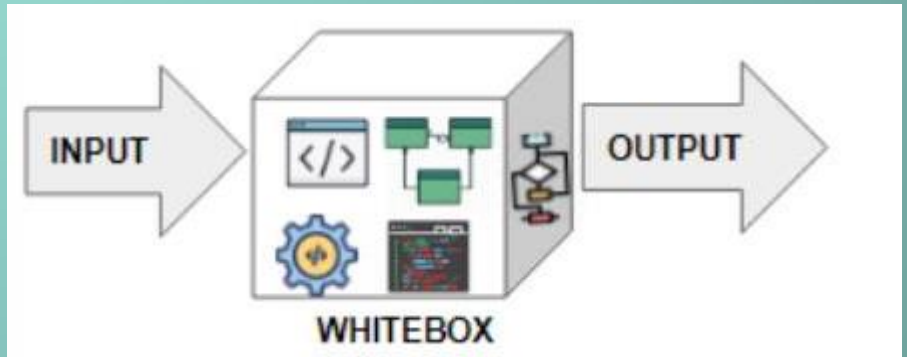
Pruebas Funcionales

- ◇ Caja Blanca
- ◇ Caja Negra
- ◇ Caja Gris



Pruebas Caja blanca

- Pruebas unitarias
- Algunas pruebas de seguridad
- Verificación de calidad de código con ciertas herramientas

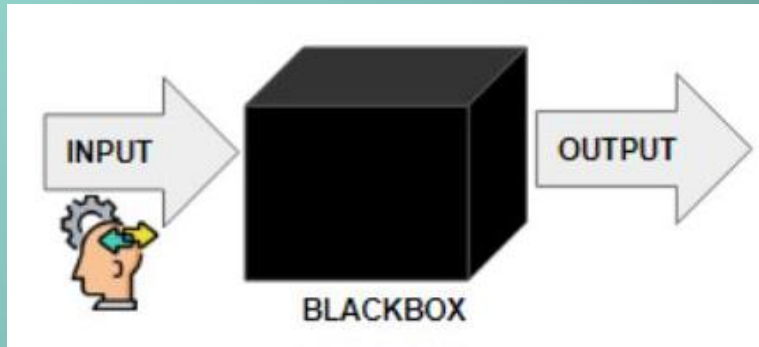


Pruebas Caja Gris

- Pruebas funcionales de backend
- Algunas pruebas funcionales de front end
- Pruebas de integración
- Pruebas de carga
- Pruebas de estrés
- Scanners de seguridad

Pruebas caja Negra

- Pruebas funcionales de front end(sin acceso a logs u otras herramientas)
- User Acceptance Testing
- Pruebas de experiencia de usuario
- Alfa
- Beta

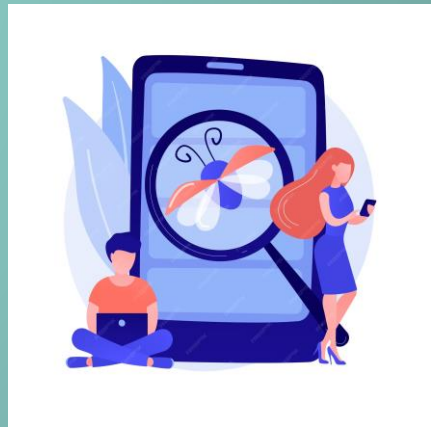


Pruebas Alfa

- Pruebas en presencia del equipo de desarrollo
- Pruebas por parte de algún representante del cliente
- Monitoreo constante
- El sistema no está del todo terminado

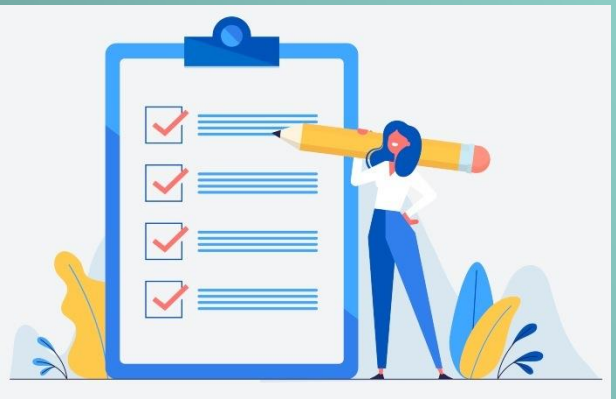
Pruebas Beta

- Grupo de los usuarios o clientes finales
- El desarrollador no está presente
- Se crean logs muy detallados



Pruebas de aceptación de usuario

- La aplicación está terminada pero antes de pasar a producción...
- El cliente inicia pruebas propias con data real
- Asegurar que información de producción funcione sea complatible

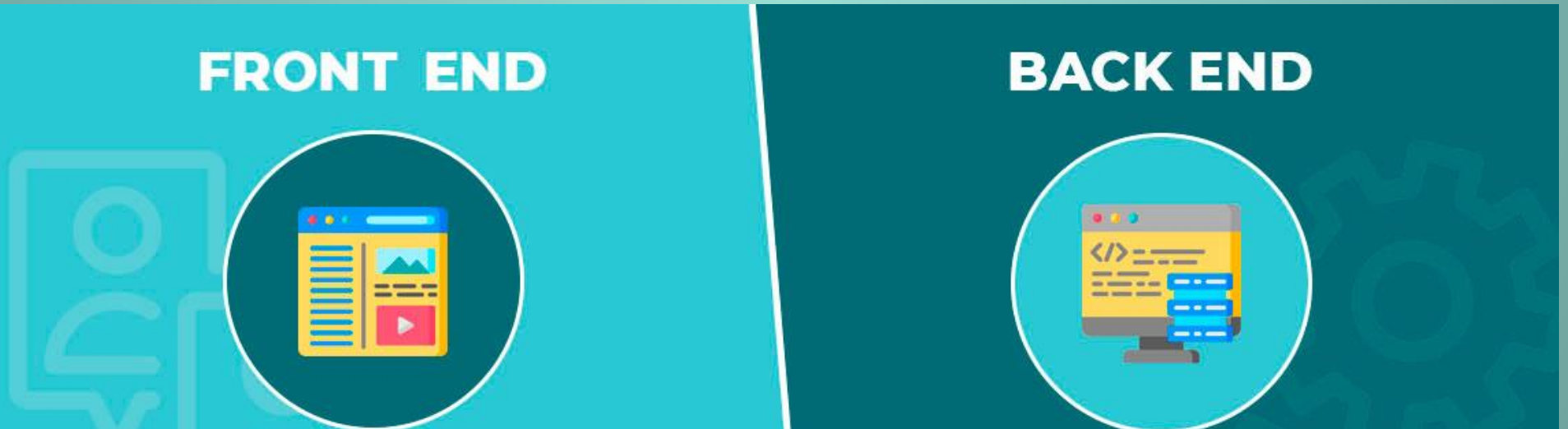


Pruebas Funcionales

- Pruebas que tienen como objetivo probar tanto los escenarios esperados como negativos basados en los casos de uso

Las pruebas funcionales pueden:

- ◇ Probar la interfaz gráfica
- ◇ Probar backend



Pruebas de integración

- Verificar que diferentes servicios o módulos se acoplen correctamente
- Los tipos de datos deben coincidir

Pruebas de experiencia de usuario

- Verificar qué tan intuitivo es el sistema
  - Controles fáciles de usar
  - Distribución lógica de elementos en pantalla
  - Instrucciones comprensibles

QA

Pruebas automáticas

4

Pruebas Unitarias

- Pruebas sobre métodos y funciones
- A nivel de código
- **Code Coverage**

Backend automático

- Simular llamadas del sistema
- Asserts pero a nivel de respuestas

UI Automática

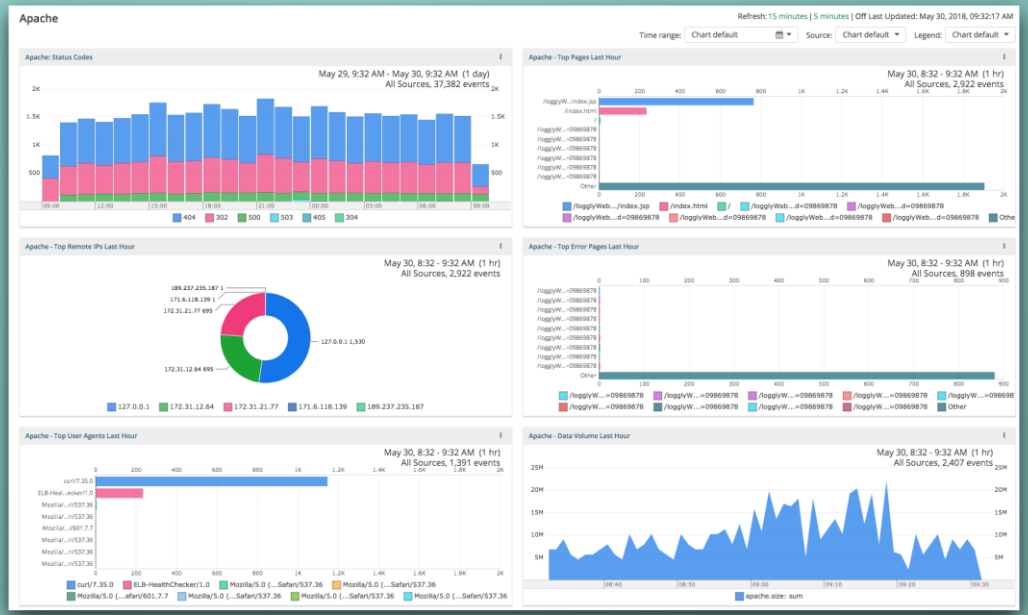
◇ Simular las acciones de un usuario

Pruebas de carga

- Asegurar la cantidad de transacciones soportadas por el SW.

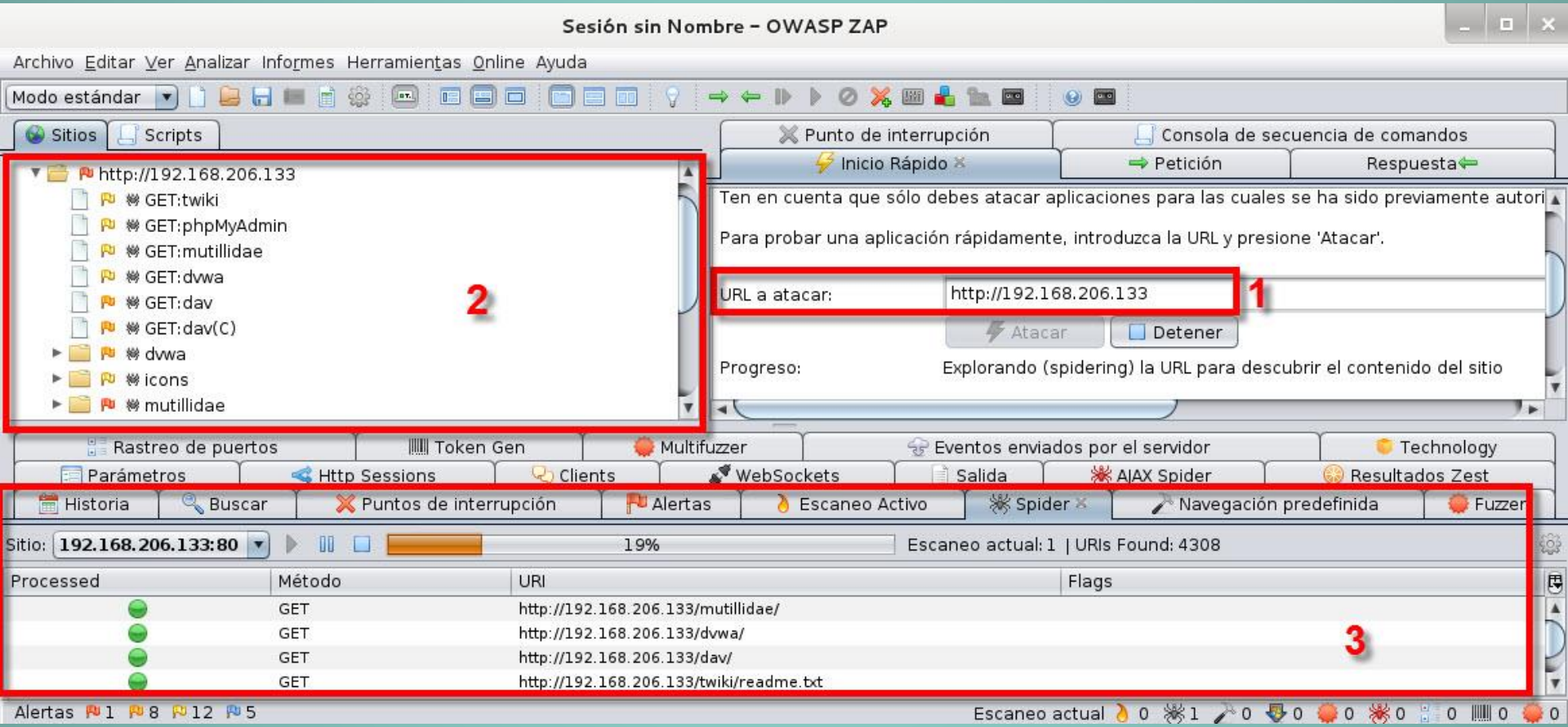
Pruebas de estrés

- Cómo se comporta el HW con cierta carga



Pruebas de seguridad

- Análisis estático de código
- Análisis dinámico de código
- Escáners de seguridad externos
- Auditores de seguridad
- Pruebas manuales


Sesión sin Nombre – OWASP ZAP
Archivo Editar Ver Analizar Informes Herramientas Online Ayuda
Modo estándar
Sitios
Scripts
http://192.168.206.133
GET:twiki
GET:phpMyAdmin
GET:mutillidae
GET:dvwa
GET:dav
GET:dav(C)
dvwa
icons
mutillidae
2
Punto de interrupción
Consola de secuencia de comandos
Inicio Rápido
Petición
Respuesta
Ten en cuenta que sólo debes atacar aplicaciones para las cuales se ha sido previamente autori
Para probar una aplicación rápidamente, introduzca la URL y presione 'Atacar'.
URL a atacar:
http://192.168.206.133
1
Atacar
Detener
Progreso:
Explorando (spidering) la URL para descubrir el contenido del sitio
Rastreo de puertos
Token Gen
Multifuzzer
Eventos enviados por el servidor
Technology
Parámetros
Http Sessions
Clients
WebSockets
Salida
AJAX Spider
Resultados Zest
Historia
Buscar
Puntos de interrupción
Alertas
Escaneo Activo
Spider
Navegación predefinida
Fuzzer
Sitio:
192.168.206.133:80
19%
Escaneo actual: 1 | URIs Found: 4308
Processed
Método
URI
Flags
GET
http://192.168.206.133/mutillidae/
GET
http://192.168.206.133/dvwa/
GET
http://192.168.206.133/dav/
GET
http://192.168.206.133/twiki/readme.txt
3
Alertas
1
8
12
5
Escaneo actual
0
1
0
0
0
0
0
0
0

QA

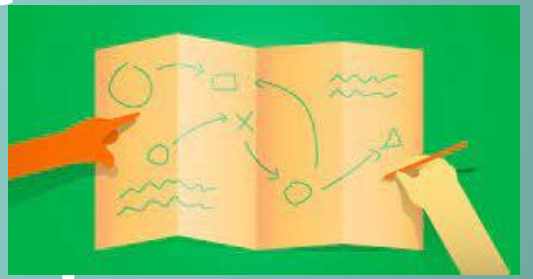
Diseño de pruebas

4

Las pruebas también necesitan planificación y diseño

Plan de pruebas

- Definición del tipo de pruebas que se harán
- Definición de módulos que se probarán
- **Alcance de las pruebas**
- **Tiempo de ejecución de pruebas**



¿Por qué es importante el plan y la estrategia?

- Muchas veces es un entregable al cliente
- Es algo que ayuda a todo el equipo
- Los programadores también pueden ver las pruebas
- Las pruebas se ejecutarán de forma ordenada y se sabrá el estado de las mismas en todo momento
- Se pueden realizar iteraciones de forma clara

Es importante que las pruebas sean flexibles y reutilizables

QA

Escenarios de prueba

5

Diseño

- Creación de casos de pruebas
  - Definir el camino feliz
  - Definir casos negativos
- Definición de herramientas necesarias
- Definición de accesos necesarios
- **Comprensión detallada de los requerimientos**

Casos de prueba

- **Título**
- **Pasos**
- **Resultados esperados**
- Precondiciones
- Adjuntos

QA

Ejecución de pruebas

6

Pruebas Funcionales

- Verificar todas las combinaciones que un usuario podría ejecutar
- Ejecución de los escenarios de pruebas previamente diseñados

Estados de las pruebas

- Passed
- Failed
- Skip/Not applicable
- **Blocked**

Configure Chart

Chart Type

- Pie
- Bar
- Column
- Stacked bar
- Pivot table

Name: Test Plan for Cycle 1 Overview

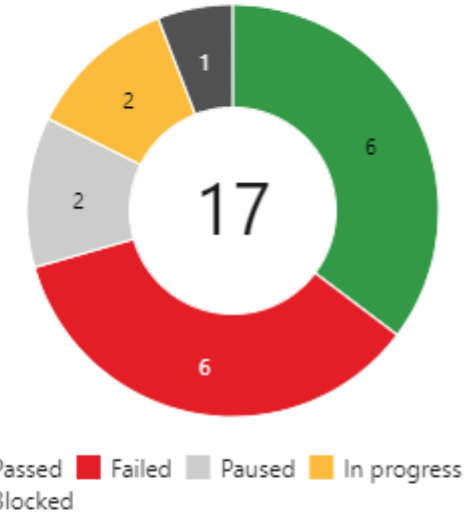
Group by*: Outcome

Aggregation: Count

of: Tests

Sort: Value, Descending

Series: Passed



OK Cancel

7

QA

Mejora contínua

Pruebas como parte del proceso de mejora continua

- Ayuda a mejorar el software en cada iteración
- Entre más tipos de control de calidad se efectúen, el software se hace menos vulnerable.
- Las métricas provenientes de este proceso ayudan en otras fases
- Ayudan a los desarrolladores a mejorar

Mejora continua en las pruebas

- Al llevar un registro de todo se puede:
  - Mejorar en la creación de escenarios de prueba.
  - Agilizar el tiempo de ejecución de pruebas.
  - Agilizar la búsqueda de errores
  - Abarcar más tipos de prueba

Casos de prueba como métricas

- Nos indican la calidad de cada entrega del equipo.
- Nos indican un progreso del proyecto.
- Indican la factibilidad de una entrega.
- Se identifican errores recurrentes y puntos débiles en el proceso de desarrollo.