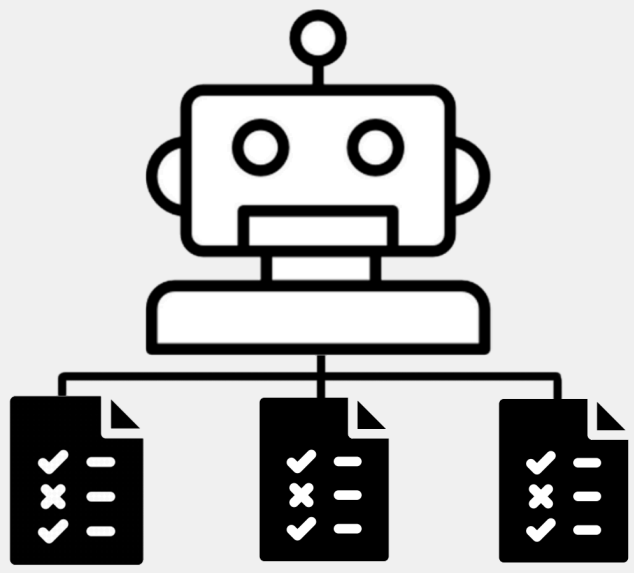
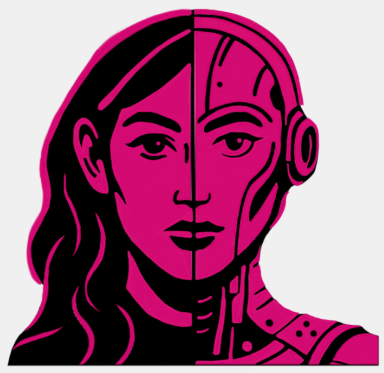
AUTOMATIZACIÓN DE PRUEBAS

Guía rápida basada en
ISTQB CTAL-TAE v2.0



Aprende lo esencial para iniciar y optimizar pruebas automatizadas

Mujeres Testing – Latam



Creado por Fransa J. Aravena

¿QUÉ ES LA AUTOMATIZACIÓN DE PRUEBAS?

Es el uso de herramientas y frameworks para controlar, configurar, ejecutar, validar y reportar pruebas sin intervención humana.

Esto permite aumentar la:

Velocidad

Repetibilidad

Cobertura de pruebas

Abarca una amplia gama de software cómo SUT:

- Con interfaz de usuario
- Sin interfaz de usuario
- Aplicaciones móviles
- Protocolos de red y conexiones

La automatización NO reemplaza las pruebas manuales:
Las complementa, especialmente en pruebas de regresión, pruebas de humo y repetitivas.

Creado por Fransa J. Aravena

CONCEPTOS CLAVE

SUT (*System Under Test*): **Sistema bajo prueba**. Puede ser una aplicación, módulo, componente.

TAA (*Test Automation Architecture*): **Arquitectura de Automatización de Pruebas**. Diseño técnico que define la estructura de alto nivel de una solución de automatización de pruebas, incluyendo sus componentes, capas, interfaces y la integración con otros sistemas.

TAF (*Test Automation Framework*): **Framework de Automatización de Pruebas**. Conjunto de herramientas, librerías, estándares, convenciones y buenas prácticas que soportan la implementación, ejecución y mantenimiento de pruebas automatizadas.

Flaky test: **Prueba inestable**, que a veces es exitosa y otras veces falla, sin que el código del sistema haya cambiado.
Posibles causas: problemas de sincronización, datos inconsistentes, dependencias externas, ambientes no controlados, etc.

CI/CD: **Integración e Implementación Continua**. Práctica que permite integrar y entregar software automáticamente y frecuentemente, con validación continua mediante pruebas automatizadas.

Paralelismo: Ejecución simultánea de pruebas automatizadas para reducir tiempos y optimizar la retroalimentación en CI/CD.

Creado por Fransa J. Aravena

VENTAJAS

La automatización de pruebas permite...

Ejecutar más pruebas por cada build en comparación con las pruebas manuales.

Crear y **ejecutar** pruebas que no se pueden ejecutar manualmente.

Realizar pruebas más **complejas** que las manuales.

Ejecutar pruebas más **rápido** que las pruebas manuales.

Minimizar riesgo de **fallos** causados por intervención **humana**.

Usar los **recursos** de prueba de forma más **eficaz** y **eficiente**.

Proporcionar retroalimentación más **rápida** sobre la **calidad** del SUT.

Mejorar la **fiabilidad** del sistema.

Creado por Fransa J. Aravena

DESVENTAJAS

La automatización de pruebas requiere...

Costos adicionales nuevos profesionales, recursos materiales, formación, etc.

Inversión inicial para configurar una solución de automatización de pruebas.

Tiempo para desarrollar y mantener la solución.

Objetivos claros de automatización para garantizar el éxito.

Rigidez de las pruebas, menor adaptabilidad a los cambios del SUT.

PASOS

Para implementar Automatización en tu proyecto

1. **Analizar** la viabilidad y objetivos
2. **Identificar** qué automatizar, definir criterios y métricas
3. **Diseñar** TAA (Arquitectura) y TAF (Framework)
4. **Desarrollar** scripts iniciales
5. **Integración** con CI/CD
6. **Ejecución** y generación de reportes
7. **Mantenimiento** y optimización continua

Creado por Fransa J. Aravena

BUENAS PRÁCTICAS

Selección estratégica de pruebas a automatizar
Priorizar casos estables, repetitivos y de alto valor.

Diseño modular y reutilizable (TAA y TAF)
Siguiendo patrones como Page Object Model (POM).

Documentar, versionar y mantener código
Revisar y refactorizar scripts regularmente. Detectar y eliminar flaky tests.

Integración temprana y continua (Shift-Left + CI/CD)
Ejecutar pruebas automatizadas en cada commit/build.

Gestionar datos de prueba
Usar datos controlados y separados de scripts (Data-Driven).

Reportes claros y accionables
Generar reportes con métricas clave.

Colaboración y comunicación
Trabajar en conjunto con Desarrolladores, DevOps, QA manuales.

Creado por Fransa J. Aravena

¿QUÉ AUTOMATIZAR?

Sí Automatizar

- Pruebas de regresión y repetitivas.
- Escenarios críticos y de alto riesgo.
- Pruebas de carga y performance.
- Casos con grandes volúmenes de datos.
- Pruebas de humo y sanidad.
- Validaciones de reglas de negocio estables.

NO Automatizar

- Pruebas exploratorias y de usabilidad.
- Pruebas que cambian frecuentemente.
- Verificaciones visuales complejas.
- Pruebas únicas o de una sola ejecución.
- Funcionalidades en desarrollo activo.

Creado por Fransa J. Aravena

PATRONES DE DISEÑO

Solución reutilizable, comprobada y documentada para resolver problemas comunes en automatización.

Patrón Fachada (Facade Pattern)

Oculta la implementación interna y expone únicamente lo necesario para que el tester cree los casos de prueba.

Beneficios

- **Simplifica** la **interacción** con el sistema bajo prueba (SUT).
- **Facilita** el **mantenimiento** al aislar los cambios internos del SUT.

Ejemplo: Una clase LoginFacade que expone solo el método login(user, pass), ocultando pasos internos como abrir la página, localizar campos y hacer click.

Modelo de Objetos de Página (POM)

Crea una clase para cada página o componente del SUT, centralizando los elementos (localizadores) y acciones relacionadas.

Beneficios

- **Mantenimiento** más **simple**: cambios en la UI solo requieren actualizar la clase correspondiente.
- **Código** más **legible** y **organizado**.

Ejemplo: LoginPage con métodos enterUsername(), enterPassword(), usados en múltiples casos de prueba.

Patrón Singleton

Garantiza que exista una única instancia de un objeto específico, asegurando un único punto de acceso.

Beneficios

- **Evita** instancias **duplicadas** del driver que interactúa con el SUT.
- **Facilita** la **gestión** de recursos y conexiones compartidas.

Ejemplo: Mantener una única instancia del driver que interactúa con el SUT durante toda la ejecución de las pruebas.

Patrón de modelo de flujo

Expansión del POM, agrega una capa adicional (facade) que almacena todas las acciones de usuario sobre los modelos de página.

Beneficios

- **Mayor abstracción**: encapsula flujos comunes de negocio.
- **Reutilización** de pasos en múltiples scripts.

Ejemplo: Combina los pasos de Login y Transferencia en un solo método para reutilizarlo en varias pruebas.

Creado por Fransa J. Aravena

MÉTRICAS Y KPIs

Cobertura de las pruebas

% de funcionalidades automatizadas

Cobertura del código

% del código cubierto por pruebas automatizadas

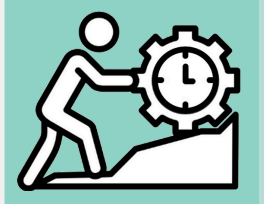
Tasa de prueba inestable

% de pruebas con resultados inconsistentes

Tasa de éxito de ejecución

% de pruebas que pasan sin errores

Esfuerzo de mantenimiento



Horas invertidas en mejorar o corregir las pruebas

Tasa de falsos positivos/negativos

% de fallos por problemas de automatización

Tiempo de ejecución

Tiempo total de ejecución de suite de pruebas

Retorno de la inversión (ROI)

Beneficios obtenidos por la automatización en relación con los costos totales de su implementación y mantenimiento

Creado por Fransa J. Aravena

HERRAMIENTAS Y FRAMEWORKS

UI (Interfaz de usuario)

Playwright	**JavaScript**, **TypeScript**, Python, Java, .NET (C#)
Selenium	**Java**, C#, **Python**, Ruby, JavaScript (Node.js), Kotlin
TestCafé	JavaScript**, TypeScript**
cypress	JavaScript, TypeScript

Mobile

appium	**Java**, Python, Ruby, **JavaScript**, C#, PHP
espresso	**Java**, Kotlin (SDK oficial de Android)
XCTest	Swift, **Objective-C**

API (Interfaz de programación de aplicaciones)

POSTMAN	**JavaScript**
REST ASSURED	**Java**
Karate Labs	**Java** - DSL propio basado en Gherkin

CI/CD

Jenkins	Configuración de Pipelines en **Groovy**
GitLab	Configuración de Pipelines en **YAML**
GitHub Actions	

Reporting

Allure Report

EXTENT REPORT

reportportal

Creado por Fransa J. Aravena

PARTICIPANTES Y TAREAS

Ingeniera de Automatización de Pruebas

- Diseña y mantiene frameworks y scripts.
- Selecciona herramientas y diseña la arquitectura.
- Aplica buenas prácticas y patrones.

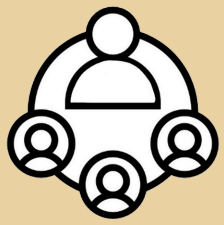
Líder de Pruebas

- Define estrategia y objetivos.
- Asigna recursos y gestiona riesgos.
- Evalúa ROI y efectividad.

Desarrollador/a

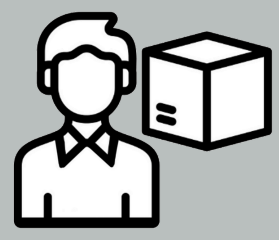
- Crean componentes reutilizables.
- Integran y soportan CI/CD.

DevOps



- Integra pruebas en pipelines.
- Gestiona entornos y despliegues.

Propietario del Producto



- Alinea la automatización con el negocio.
- Prioriza qué automatizar.
- Aprueba presupuesto y plazos.

Creado por Fransa J. Aravena

CONSEJOS FINALES

Confía

Confía en tu criterio y en la experiencia adquirida como Ingeniera de Calidad; tu conocimiento es clave para tomar decisiones acertadas.

Comunícate

Si las cosas no van como esperas, conversa, relaciónate y llega a acuerdos. Por ejemplo: Si las pruebas fallan por localizadores dinámicos, coordina con el equipo de desarrollo para definir IDs únicos y estables.

Adáptate

Sé flexible e integra nuevas herramientas cuando el contexto lo requiera; pruébalas y evalúa su impacto antes de recomendarlas.

Usa la Inteligencia Artificial a tu favor

Aprovecha la IA para acelerar la creación de casos automatizados, el mantenimiento y el debugging, pero siempre revisa y analiza el código críticamente para evitar inconsistencias o casos débiles.

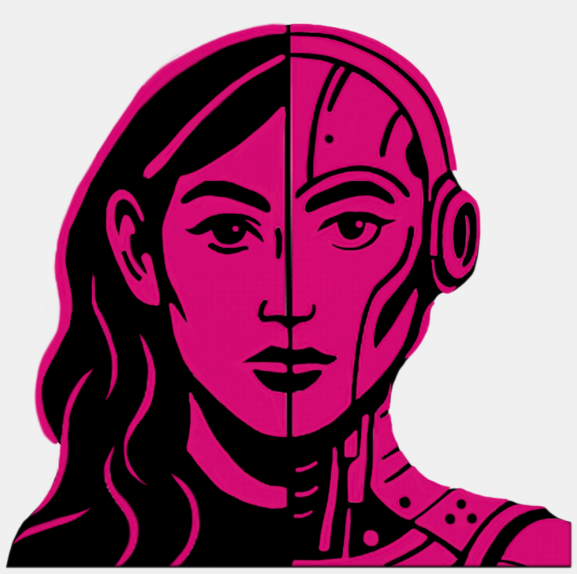
Todos en la misma página

Mantén alineado al equipo con un mismo contexto, prácticas y protocolos, para asegurar la consistencia en las pruebas y en la calidad del producto.

Creado por Fransa J. Aravena

Conectando y apoyando a Mujeres en LATAM para crecer juntas en el camino del Testing

Mujeres Testing - Latam



Creado por Fransa J. Aravena

Agosto, 2025