

La Diferencia entre el fracaso y el Éxito de la Automatización de Pruebas

Elisabeth Hendrickson

El Software de Árbol de calidad, Inc.

esh@qualitytree.com

21 de agosto de 1998

Traducido por Angel Hermoza Salas (Perú)

ahermoz@editoraperu.com.pe

Maestría en Ingeniería de Software (UNMSM – Lima – Perú)

Abstract

La comprobación automatizada es una actividad cara e intensiva en recursos. Puede ser difícil lograr el retorno de la inversión en la automatización de la prueba que la dirección superior espera. Este trabajo muestra dos proyectos de automatización de prueba en detalle: uno que tuvo éxito y uno que falló. Ambos proyectos fueron llevados por la misma persona y en la misma compañía y se pretendía que automatizara la prueba del mismo software. El propósito de este trabajo es ilustrar las diferencias entre los proyectos que llevaron al éxito o al fracaso.

INTRODUCCIÓN

En 1993 llegué a ser el líder de un esfuerzo de automatización de prueba para un vendedor de software independiente. Seis meses después, el equipo de la comprobación automatizado se disolvió habiendo hecho pocos progresos. La compañía intentó inicialmente integrar la automatización de la prueba con la comprobación manual, pero dejó toda la pretensión de automatización luego de dos meses. Fui asignada a otros proyectos donde trabajé automatizando mis propias pruebas esporádicamente.

En 1995 empecé un serio esfuerzo para automatizar mi proyecto actual, un puerto Mac del PC principal de la línea de producto. Mis esfuerzos tuvieron el éxito y se volvieron un programa modelo para un esfuerzo de automatización más grande. En 1996 me nombraron líder de un esfuerzo de automatización de prueba del PC principal de la línea de producto. El proyecto de automatización de prueba fue mejor y más exitoso que el segundo. Este trabajo examina las razones del fracaso del primer proyecto y el éxito del segundo proyecto.

Similitudes del proyecto

La comparación de estos dos proyectos es interesante debido a sus similitudes: la misma compañía, el mismo producto y el mismo líder.

La compañía es vendedora de software independiente de tamaño medio. Yo manejé ambos proyectos. El producto es una aplicación basada en la compleja interfase del cliente (GUI) cliente/servidor que consiste en muchos ejecutables para ejecutarse en tareas diferentes. El producto tiene una arquitectura abierta, soportando una variedad de las bases de datos (Oracle, Sybase, MS SQL) y sistemas operativos de red.

Diferencias del proyecto

Aunque los proyectos ocurrieron en ambientes similares, los proyectos eran inmensamente diferentes. Las diferencias entran en dos categorías: las diferencias a nivel de proyecto y las diferencias técnicas. Ninguna categoría de diferencias debe descontarse; los dos contribuyeron grandemente al fracaso del primer proyecto y al éxito del segundo.

Las diferencias a nivel de proyecto incluyen el nivel de experiencia de dirección, las metas del proyecto, y la prontitud de la organización para automatizar. Las diferencias técnicas incluyen la arquitectura de automatización, los métodos de la comprobación, reusabilidad y mantenimiento de código de la prueba, y las prácticas de programación.

DIFERENCIAS EN LOS NIVELES DEL PROYECTO

Esta sección examina las diferencias a nivel de proyecto:

- Nivel de experiencia de dirección o liderazgo.
- Las Metas de los proyectos
- Comunicación
- La Prontitud para automatizar
- El Papel del equipo de la comprobación de la automatización.

El Primer Proyecto: Un Caso de Estudio de Como no Automatizar

El proyecto que falló es un caso de estudio en como no automatizar. Desgraciadamente, quizás debido a la tendencia de los vendedores de herramienta de comprobación automatizados a vender con exceso la facilidad de uso de sus productos, los errores que cometimos son todos demasiado comunes.

Los problemas que llevaron al fracaso incluyeron la dirección inexperta, metas poco realistas, y el hecho que la organización no estaba lista para automatizar.

La Dirección inexperta

Cuando acepté el papel de líder del equipo, era inexperta tanto en la dirección y en la prueba de automatización. El gerente de QA también le faltó la experiencia con los proyectos de automatización de prueba. Esto se volvió una debilidad crítica como la presión para un retorno en la inversión de automatización montada. Cuando la mayor presión se hizo llegar, la mayor caída de moral se dejó caer en el grupo de automatización, y los menores logros que el grupo de la prueba automatizado.

Debido a mi inexperiencia, tenía dificultad para comunicar los problemas que estorbaban al equipo técnico al gerente de QA de modo que él pudiera entender. También tenía la dificultad de comunicar los objetivos comerciales de automatización a mi equipo. Finalmente, gasté demasiado tiempo creando mapas de Gantt de un cronograma ficticio y demasiado lejos para automatizarme

Las Metas poco realistas

La meta que fui dando cuando empecé era "Automatice Todo." Yo debí haber cuestionado esta meta. Cuando se puso evidente que todo no podía automatizarse en 3 meses (el tiempo original para el proyecto), la meta se volvió "el Golpe para el Ciervo." Nosotros no tuvimos éxito definiendo "el golpe eficaz para el ciervo" de las actividades. Así, las metas no eran claras y no teníamos los medios para medir nuestro progreso contra las metas.

La comunicación con la Dirección Superior

La dirección inexperta llevó a una falta de comunicación con la dirección superior sobre cómo su dinero estaba siendo gastado y qué beneficios estaban ganando de la pruebas de automatización.

Por consiguiente había presión mayor para reducir la inversión en la automatización porque el pago no era obvio.

No estar listo para Automatizar

Finalmente, aún cuando nosotros hubiésemos podido resolver todos los problemas anteriores, había todavía uno obstáculo mayor: la organización no entendió cómo probar el producto de una manera manual. Las pruebas necesitaron ser diseñadas de ser automatizadas.

El papel del Equipo de la Comprobación Automatizado

En el equipo de la comprobación automatizada no nos vimos como una organización de servicio. Nos vimos como constructores de la herramienta. Pensamos construir las herramientas para aquellos que vimos tenían la necesidad sin conferir con nuestros "clientes" (los verificadores manuales).

El equipo de automatización también padeció la falta de respeto para los verificadores manuales. Vimos la prueba manual como el trabajo pesado a no ser soportado.

Ahora, me avergüenzo profundamente cuando pienso sobre la analogía que diseñe entre las pruebas de automatización y la construcción. La dirección superior quiso mover a los miembros de equipo de

automatización fuera de la automatización para probar un parche manualmente. Luchando en este esfuerzo para no descarrilar la automatización de la prueba, use la analogía siguiente: "Hay un grupo de las personas que excavan una reguera con palas. Nosotros estamos construyendo una excavadora. Usted quiere que dejemos caer lo que estamos haciendo y recojamos las palas."

Estoy avergonzada profundamente porque:

1. Esta analogía lleva la idea de que pienso que los verificadores manuales son semejantes a las excavadoras de la reguera. En ese momento, era suficientemente ingenua que por haber pensado eso realmente. He retomado mis sentidos. Comprendo que era una declaración ofensiva.
2. He venido a comprender que a veces realmente se puede tomar mucho demasiado para construir una excavadora.

Imagine un fuego del bosque. Los luchadores de fuego están trabajando para construir un cortafuego excavando difícilmente las regueras. La excavadora ha llegado, algún ensamblaje requirió. Un grupo paró para ensamblar la excavadora. Ellos descubren que los tomará más tiempo que el anticipado para ensamblar la excavadora y que ellos no tienen las herramientas correctas para este trabajo. Ellos pueden luchar con la excavadora o pueda regresar a luchar contra el fuego.

Si el fuego es una muy malo y hay pocos bomberos, puede ser el mejor interés de la organización que los bomberos regresen a excavar del descanso y pierden el interés en la excavadora por ahora. Después de que el fuego esté bajo el mando, Luego los ensambladores de la excavadora pueden ver sobre encontrar las herramientas correctas completar el trabajo.

Si la organización padece un fuego uno tras de otro y el grupo de la prueba siempre está en la modo de lucha contra el fuego, la automatización no ayudará y podría doler. En las organizaciones manejadas en crisis, es importante mirar la causa en la raíz de los fuegos: el proceso, dirección, metas, y métodos. Después de que algunos de los factores de causa raíz sean ubicados las herramientas para la automatización pueden traerse.

La analogía también sugiere otro posible problema para ensamblar la excavadora: la excavadora no puede operar en el terreno. Semejantemente, no todas las herramientas de la prueba trabajarán bien con todos las aplicaciones.

La automatización no es una bala de plata por cualquier estiramiento de la imaginación; es una herramienta. Como cualquier otra herramienta, puede ser muy eficaz si se usa bien, pero también es muy fácil de causar daño severo a uno mismo, al proyecto, o a la organización usándolo imprudentemente.

El Segundo Proyecto: Los Factores Clave de Éxito

Unos de los factores más cruciales en el éxito del segundo se basó en los factores críticos del fracaso del primero proyecto. El segundo proyecto tenía dirección experimentada y metas realistas, y para el tiempo del segundo proyecto, la organización estaba lista para su automatización.

La Dirección experimentada

La cadena de dirección en su totalidad, toda la manera de pensar del presidente de la compañía, cambió entre el primer proyecto y el segundo. El nuevo gerente de QA había estado terminado varios proyectos de automatización y entendió los desafíos inherentes en la automatización de la prueba. Tenía dos años la dirección adicional y experiencia de automatización bajo mi mando. Finalmente, pude ver en mis experiencias en el primer proyecto para dirigir el segundo proyecto sin los errores anteriores.

Las Metas realistas

En el primer proyecto, me dieron las metas del proyecto. En el segundo proyecto, el equipo de comprobación automatizada definió las metas que creíamos apropiados y los presentamos a la dirección.

Las metas del segundo proyecto eran más estrechas y mejor medidas:

1. Ahorrar el tiempo de los verificadores manuales (medido en horas)
2. Mejore la cobertura de la prueba (medido en pruebas que no podrían correrse manualmente)

La agenda detrás de las metas del proyecto también cambiaron. En el primer proyecto, el enfoque era "Automatice Todo" en parte para reducir el número de verificadores manuales necesarios. En el segundo proyecto, el enfoque estaba en apoyar a los verificadores manuales y acortar el ciclo de la comprobación.

Es entendible que los verificadores manuales no apoyaron el primer proyecto mientras apoyaron el segundo proyecto. Se sentían amenazados por la primera iniciativa de automatización (desde que una de las metas era reducir la necesidad de verificadores manuales). El departamento interno asumió la crítica.

No todos los niveles de dirección estaban de acuerdo que éstas eran las metas correctas para el proyecto. Sin embargo, debido a que el gerente de QA estaba de acuerdo con las metas del equipo de prueba, pudimos proporcionar un frente unido a la dirección superior. Esto permitió al proyecto quedarse en el proyecto y mantener a los individuos en el proyecto de la distracción de "las demandas especiales."

La comunicación

Con las limitaciones del primer proyecto firmemente en mente, me aseguré de comunicar las metas, los éxitos, y los tropiezos claramente al equipo de comprobación automatizada, los verificadores manuales, y todos los niveles de dirección que mostraron poco interés.

El equipo de la prueba automatizado produjo un reporte de estado semanal que incluyó las medidas de nuestro progreso a través de las metas así como los proyectos actuales, logros, los problemas, y los planes futuros. Esto puede parecer como excesivo, pero me tomó aproximadamente 2 horas por semana juntar esta información de los informes de estado de los automatizadores individuales. Creando este informe me sirvió para tener que contestar las mismas preguntas repetidamente sobre nuestro estado.

Distribuyendo la misma información a todas las partes interesadas tenía el beneficio adicional de asegurar que todos tuvieramos la misma comprensión de las metas y el estado del proyecto; esto redujo el número de demandas contradictorias.

Prepare para Automatizar

Entre el primer proyecto de automatización y el segundo, la organización de QA maduró substancialmente. Habíamos creado casos de la prueba reales y planes de la prueba, los métodos establecidos de rastreo progresaron, y teníamos un mejor entendiendo de cómo probar el producto. Esta comprensión habilitó al equipo de la comprobación automatizado para enfocar nuestra comprobación automatizada con la importante funcionalidad. También nos permitió que apoyáramos bien a los verificadores manuales.

El papel del Equipo de la Comprobación Automatizado

En el segundo proyecto, el equipo de la comprobación automatizado llegó a ser una organización de servicio con el manual los verificadores como nuestros clientes. En lugar de intentar diseñar las pruebas automatizadas, trabajamos con los verificadores manuales para asegurarnos que los programas que nosotros creamos tenían valor por ellos.

Además, se animaron a verificadores fuera del grupo de comprobación automatizado para aprender las herramientas de automatización. Tuvimos clases y trabajamos con verificadores uno-en-uno para ayudarles a aprender a automatizar sus propias pruebas y activar las actividades. Hubieron tres ventajas para trabajar:

1. Los verificadores ganaron un mejor entendiendo de lo que nosotros estábamos intentando lograr.
2. Muchos de los verificadores ganaron la confianza en su habilidad de contribuir a la automatización.
3. Los verificadores entendieron que ellos eran una parte grande del proceso, no una molestia para ser reemplazados por la automatización.

La última visión del grupo de la comprobación automatizado era que habría un pequeño grupo del centro de automatizadores que crearía y mantendría la infraestructura de automatización (las guarniciones). Los verificadores manuales que tenían un interés en la automatización automatizarían sus propias pruebas. Los verificadores manuales que no tenían un interés en la automatización de la prueba no serían obligados a automatizar: reconocimos que no todo podría automatizarse y que habría siempre una necesidad de verificadores manuales.

LAS DIFERENCIAS DE APLICACIÓN TÉCNICAS

Las decisiones técnicas jugaron un papel importante en el fracaso del primer proyecto y el éxito del segundo proyecto. Algunas diferencias técnicas clave entre el primer proyecto y el segundo son :

- Arquitectura de sistema de prueba Automatizada
- El Método de creación de programas
- El Método de comprobación
- Prácticas de programación

El Primer Proyecto: No-reusable, No-Mantenible, Fracasos Falsos

La Arquitectura de Sistema de Prueba automatizada

Por "la arquitectura," quiero decir plan global e infraestructura. En el primer proyecto, nosotros éramos incapaces de crear una arquitectura real principalmente porque la herramienta no proporcionó una manera buena de organizar programas o para construir bibliotecas reusables de funciones.

El método de Creación de la Escritura

El método primario de creación de programas en el primer proyecto era el registro & reproducción. Modificamos los programas grabados para agregar declaraciones de flujo de control (if , while, for, etc.), pero no escribimos programas desde el principio.

El método de Comprobación

Quizás el error más grande que cometimos en el primer proyecto estaba enfocando en las comparaciones de bitmaps para la comprobación. La herramienta apoyó otros métodos de comprobación; pensamos eso equivocadamente, las comparaciones del bitmap nos permitirían verificar el UI más completamente. Hay muchos problemas con las comparaciones del bitmap: producen fallas falsas frecuentes, son difíciles de manejar, y ellos utilizan mucho espacio del disco.

Bitmap compara los frecuentes fracasos falsos producidos debido a que las diferencias simples, inocuas pueden ocasionar una falla de comparación del bitmap. Por ejemplo, ejecutando el programas en una máquina con una resolución diferente o con poco color y a estos cambios los UI cosméticos envían un mensaje con la causa de los fracasos.

Programando Prácticas

Porque usamos el registro & reproducción para crear los programas en el primer proyecto, nuestra programación se enfocaron más en las convenciones de denominación de archivo que las prácticas buenas de programación. Nunca hemos sostenido una revisión del código, no usamos el control de los programas fuente, y nosotros hicimos la programación básica como ardua codificando en nuestros programa. Además, la herramienta no soportó crear un una biblioteca de código reusable , de modo que después de un esfuerzo inicial por crear una biblioteca de funciones, nos rendimos. Gastamos mucho tiempo duplicando esfuerzos y manteniendo el código redundante.

El Segundo Proyecto: La Automatización de la prueba está Programando

La lección más importante que aprendí en el primer proyecto es que la automatización de la prueba es programación y debe manejarse de acuerdo con este concepto. La grabación & reproducción la cinta no produce las pruebas automatizadas robustas.

Es importante diseñar el sistema de la prueba de frente, escribir las especificaciones, documentar el trabajo, tomar el control de los fuentes, y para hacer todas esas otras cosas que nosotros la Aseguradores de Calidad sostenemos que el Desarrollo debe hacer. Los factores esenciales en el éxito del factor técnico del segundo proyecto ilustran esto.

La Arquitectura de Sistema de Prueba automatizada

En el segundo proyecto, pudimos definir una arquitectura global para el proyecto. Escogimos una herramienta que apoyó las funciones reusables, los datos manejados de la comprobación, y gráfica lógica entre los elementos del programa y los programas

Este documento no se enfoca en las herramientas usadas, sino en la manera en que usamos estas herramientas. Se dice, es importante notar que entre el tiempo del primer proyecto que empezó en 1993 y el segundo proyecto que empezó en 1996, la industria de herramienta de comprobación había madurado significativamente. Nosotros teníamos muchos más herramientas disponibles y las herramientas era más flexibles y poderosas.

LA CAPA ABSTRACTA

Esta capa abstracta, llamada un mapa de GUI o Marco de la Prueba que depende del sistema, nos hizo fácil adaptarnos a los cambios del producto.

Pudimos crear las declaraciones de los elementos de la aplicación para proporcionar una capa lógica entre el la aplicación bajo prueba y los programas de la prueba. Gastamos medio tiempo o más creando éstos declaraciones y las funciones reusables asociadas.

A algunos, esto podría haber parecido tiempo perdido debido a que no produjo ningún programa ejecutable. Sin embargo hicimos más del tiempo de inversión del mantenimiento. Cuando la interfaz cambió en el medio-release, era un simple problema de cambiar las declaraciones de elemento de aplicación en un lugar. En el primer proyecto de automatización donde no había ninguna capa lógica entre la aplicación y los programas, un cambio de la interfaz simple podría producir horas de reescribir programas y arreglarlos para hacer los programas trabajen.

LAS FUNCIONES REUSABLES

Donde el primer sistema no soportó las funciones reusables, el segundo si lo hizo. Pudimos usar las funciones reusables para reducir la cantidad de código requerida para cada programa. De hecho, una vez que la infraestructura estaba en el lugar, la creación de nuevos programas era trivial.

PRUEBA DE LOS DATOS

Con el segundo proyecto de automatización, Pudimos comprender el poder de datos manejando la comprobación. En lugar de crear cientos de programas, podíamos crear un programa y un archivo del datos con filas de datos.

El método de Creación de programas

En el segundo proyecto, usamos el Registro & Reproducción como una herramienta de aprendizaje. La mayoría del equipo de automatización estaba escribiendo una inmensa mayoría de sus programas a mano dentro de un par de meses después de la salida del proyecto.

Los programas escritos a mano tardan menos tiempo y con menores errores que el registro & reproducción, una vez usted conozca la herramienta de la prueba automatizada. Además, escribiendo programas a mano requiere que usted piense a través de cómo el programa trabajará en adelante. El diseño adelantado es importante en la automatización de las pruebas como programando.

El método de Comprobación

En el segundo proyecto, migramos hacia las comparaciones del bitmap y la existencia de la ventana lógica usada de las funciones. Esto redujo la proporción de fracaso falso y aceleró grandemente al desarrollo de los programas y el mantenimiento. En otros términos, en lugar de probar para ver si una ventana que se parecía exactamente en el cuadro aparecía, las escrituras verificaron para ver si una ventana con el nombre correcto aparecía. Esto la técnica nos dio información más útil que un fracaso de comparación de bitmap. Por ejemplo, nosotros creamos una prueba de humo que reconcilió los campos esperados a estar en el formulario con los campos que realmente aparecieron

Las Prácticas de programación

En el segundo proyecto, seguimos la clase de prácticas de programación que defendimos para los desarrolladores: los estándares de codificación, control de los fuentes y las revisiones del código.

Codificando estándares hizo posible para nosotros en el equipo de automatización compartir el código: todos podíamos entendernos viendo los programas y funciones. Nuestros estándares de codificación que cubrieron la función y las convenciones de la denominación de las variables, los valores duros codificados (no hacerlo), comentarios, buenas pruebas, diseño de los programas, etc. es importante notar que gastamos mucho menos tiempo definiendo los estándares en el segundo proyecto aun cuando el segundo proyecto eran más comprensivo y finalmente más útil.

El control de los fuentes nos salvó de trabajar nuevamente. Pusimos nuestras bibliotecas y programas en el control de los fuentes desde el principio del segundo proyecto. Esto demostró ser invaluable. Si perdiéramos los cambios copiando encima de un archivo, eramos capaces de volver los cambios inmediatamente: no teníamos que buscar el archivo en una cinta auxiliar o backup. Además, el control de los fuentes nos dio una lista de los cambios para rastrear la actividad: como gerente, era fácil para mí decir quién estaba trabajando en que y quienes tenían qué archivos revisando.

Las versiones del código nos permitieron compartir las técnicas, verificar la adhesión a las normas, y mejorar la fiabilidad. Durante mi tiempo en el proyecto, sostuvimos dos revisiones del código formales e innumerables revisiones informales. No intentamos las revisiones del código ni siquiera en el primer proyecto.

CONCLUSIÓN

La tabla siguiente resume las diferencias entre el proyecto exitoso y el proyecto: fallido:

	En el proyecto fallido	En el proyecto exitoso
Detalles gerenciales	La Alta Dirección tenía expectativas no realistas y fueron señaladas por el vendedor de la herramienta	La Alta Dirección tenía expectativas realistas señaladas por el líder del proyecto
Lider del proyecto	Ninguna experiencia	Experiencia
Metas del Proyecto	Las metas manejadas por el equipo del proyecto fueron : Automatizar todo	Las metas del proyecto que acepto el equipo fueron: Ahorrar el tiempo de la prueba manual, mejorar la cobertura de la prueba El Progreso de estas metas fue medido y los resultados reportados a la Gerencia tanto como las pruebas manuales
Detalles Técnicos	No se usó control de fuentes y ocasionalmente se perdió trabajo	Control de fuentes real, mejor estandar de codificación, las herramientas soportaron recuperación de errores y reuso de librerías de funciones
Tipo de Automatización	Registro & reproducción primaria	Manejador de datos primario y no Registro & reproducción

Las lecciones Aprendidas

Aprendí varias lecciones en la participación de ambos proyectos. Específicamente, es importante que :

1. Señalar metas realistas y encontrar maneras de medir el progreso hacia esas metas.

2. No sostenimiento de la automatización si la organización no está lista para él.
3. Señalar las expectativas de dirección apropiadamente.
4. Coordinar los esfuerzos de automatización con los verificadores manuales para asegurarse que no hay ninguna duplicación del esfuerzo.
5. Comunique abiertamente y a menudo sobre las metas del proyecto y el estado con todos los involucrados incluso el equipo de automatización de prueba, los verificadores manuales, y todos los niveles de dirección.
6. Busque programas simples que tengan un gran retorno
7. Tratar la automatización como programa de desarrollo: diseñe la arquitectura, use el control de los fuentes, y busque maneras de crear funciones reusables.
8. Enfoque en crear código robusto, reusable, mantenible
9. Use la herramienta correcta para el trabajo.