Probador Certificado del ISTQB®

Programa de Estudio de Nivel Avanzado

Analista de Prueba

Traducción realizada por el

Spanish Software Testing Qualifications Board

Versión ES 000.05

Basada en el Programa de Estudio

“Certified Tester - Advanced Level Syllabus, Test Analyst, Version 2019”

Spanish Software Testing Qualifications Board

International Software Testing Qualifications Board

Subgrupo de Trabajo de Analista de Prueba de Nivel Avanzado: Judy McKay (Chair), Graham Bath.

Por el programa de estudio de 2012: Judy McKay, Mike Smith, Erik van Veenendaal.

Por el programa de estudio de 2019: Graham Bath (vice-chair), Rex Black, Judy McKay, Kenji Onoshi, Mike Smith (chair), Erik van Veenendaal.

Historial de Revisiones

Versión	Fecha	Observaciones
V2012	19/10/2012	Entrega de ISTQB
V2019 V1.0	18/10/2019	Entrega de ISTQB
V2019 V1.1	19/12/2019	Lanzamiento de ISTQB

Índice General

Agradecimientos

Este documento ha sido elaborado por el grupo de trabajo de Analista de Prueba del International Software Testing Qualifications Board Advanced Level: Mette Bruhn-Pedersen (Chair); Matthias Hamburg (Product Owner); Wim Decoutere, István Forgács, Adam Roman, Jan Sabak, MarcFlorian Wendland (Authors).

El grupo de trabajo agradece a Paul Weymouth y Richard Green por su edición técnica, a Gary Mogyorodi por las comprobaciones de conformidad del Glosario, y a las Juntas Directivas por sus comentarios de revisión relacionados con la versión publicada de 2019 del programa de estudios y por los cambios propuestos.

Las siguientes personas participaron en la revisión y los comentarios de este plan de estudios:

Gery Ágnecz, Armin Born, Chenyifan, Klaudia Dussa-Zieger, Chen Geng (Kevin), Istvan Gercsák, Richard Green, Ole Chr. Hansen, Zsolt Hargitai, Andreas Hetz, Tobias Horn, Joan Killeen, Attila Kovacs, Rik Marselis, Marton Matyas, Blair Mo, Gary Mogyorodi, Ingvar Nordström, Tal Pe'er, Palma Polyak, Nishan Portoyan, Meile Posthuma, Stuart Reid, Murian Song, Péter Sótér, Lucjan Stapp, Benjamin Timmermans, Chris van Bael, Stephanie van Dijck y Paul Weymouth.

Este documento fue publicado por ISTQB® el 23 de febrero de 2021.

La versión 2019 de este documento fue producida por un equipo perteneciente al grupo de trabajo de Nivel Avanzado del International Software Testing Qualifications Board: Graham Bath, Judy McKay y Mike Smith

Las siguientes personas participaron en la revisión, comentarios y votación de la versión 2019 de este programa de estudios:

Laura Albert, Markus Beck, Henriett Braunné Bokor, Francisca Cano Ortiz, Guo Chaonian, Wim Decoutere, Milena Donato, Klaudia Dussa-Zieger, Melinda Eckrich-Brajer, Péter Földházi Jr, David Frei, Chen Geng, Matthias Hamburg, Zsolt Hargitai, Zhai Hongbao, Tobias Horn, Ágota Horváth, Beata Karpinska, Attila Kovács, József Kreisz, Dietrich Leimsner, Ren Liang, Claire Lohr, Ramit Manohar Kaul, Rik Marselis, Marton Matyas, Don Mills, Blair Mo, Gary Mogyorodi, Ingvar Nordström, Tal Peer, Pálma Polyák, Meile Posthuma, Lloyd Roden, Adam Roman, Abhishek Sharma, Péter Sótér, Lucjan Stapp, Andrea Szabó, Jan te Kock, Benjamin Timmermans, Chris Van Bael, Erik van Veenendaal, Jan Versmissen, Carsten Weise, Robert Werkhoven y Paul Weymouth.

La versión 2012 de este documento fue elaborada por un equipo perteneciente al subgrupo de trabajo de Nivel Avanzado – Analista de Pruebas Avanzado, del International Software Testing Qualifications Board: Judy McKay (Chair), Mike Smith y Erik van Veenendaal.

En el momento en que se completó el programa de estudios de nivel avanzado, el grupo de trabajo de nivel avanzado tenía los siguientes miembros (por orden alfabético):

Graham Bath, Rex Black, Maria Clara Choucair, Debra Friedenberg, Bernard Homès (Vice Chair), Paul Jorgensen, Judy McKay, Jamie Mitchell, Thomas Mueller, Klaus Olsen, Kenji Onishi, Meile Posthuma, Eric Riou du Cosquer, Jan Sabak, Hans Schaefer, Mike Smith (Chair), Geoff Thompson, Erik van Veenendaal y Tsuyoshi Yumoto.

Las siguientes personas participaron en la revisión, los comentarios y la votación de la versión 2012 del del programa de estudios:

Graham Bath, Arne Becher, Rex Black, Piet de Roo, Frans Dijkman, Mats Grindal, Kobi Halperin, Bernard Homès, Maria Jönsson, Junfei Ma, Eli Margolin, Rik Marselis, Don Mills, Gary Mogyorodi, Stefan Mohacsi, Reto Mueller, Thomas Mueller, Ingvar Nordstrom, Tal Pe'er, Raluca Madalina Popescu, Stuart Reid, Jan Sabak, Hans Schaefer, Marco Sogliani, Yaron Tsubery, Hans Weiberg, Paul Weymouth, Chris van Bael, Jurian van der Laar, Stephanie van Dijk, Erik van Veenendaal, Wenqiang Zheng y Debi Zylbermann.

Notas de la Versión en Idioma Español

El Spanish Software Testing Qualifications Board (SSTQB) ha llevado a cabo la traducción del Programa de Estudio de “ISTQB® Certified Tester - Advanced Level Syllabus, Test Analyst” versión de 2019. Este Programa de Estudio se denomina, en idioma español, “Probador Certificado del ISTQB® - Programa de Estudio de Nivel Avanzado, Analista de Prueba” versión 2019.

El equipo de traducción y revisión para este programa de estudio es el siguiente (por orden alfabético):

Responsable de la traducción: Gustavo Márquez Sosa (España)

Revisora: Luisa Morales Gómez Tejedor (España)

El Comité Ejecutivo del SSTQB agradece especialmente las aportaciones de los revisores.

En una siguiente versión se podrán incorporar aportaciones adicionales. El SSTQB considera conveniente mantener abierta la posibilidad de realizar cambios en el “Programa de Estudio”.

Traducciones Específicas, Definiciones, Acrónimos, Abreviaturas y Notas

En este capítulo se presentarán traducciones específicas, definiciones, acrónimos, abreviaturas y notas de términos y conceptos que no forman parte del Glosario de Términos del ISTQB y tampoco están incluidos en su traducción al idioma español o castellano. Se ha incorporado este apartado para facilitar la lectura y comprensión de este "Programa de Estudio".

Las "Traducciones Específicas, Definiciones, Acrónimos, Abreviaturas y Notas" de este apartado están identificados con el texto **Consultar **** en el pie de página. Sólo se identificará la primera ocurrencia del elemento.

Es conveniente observar que las traducciones de los distintos términos sean o no del glosario, se han realizado con el objetivo de evitar conflictos en sus traducciones y siguiendo las normas de traducción del SSTQB. Algunos conflictos surgen en el momento de traducir un nuevo término o programa de estudio.

La siguiente tabla está ordenada por orden alfabético de los términos de la primera columna etiquetada con **"español"**.

Traducciones Específicas		
Español	**Inglés**	**Observaciones**
asunto de interés	concern	No hay observaciones.
calendario	schedule	No hay observaciones.
ciclo de vida de desarrollo de software	software development lifecycle	No hay observaciones.
disponibilidad de asistencia	helpfulness	No hay observaciones.
interfaz visible para el usuario	user-visible interface	No hay observaciones.
iterativo embebido	embedded iterative	No hay observaciones.
prueba sin guion | prueba no especificada	unscripted testing	No hay observaciones.
solución provisional	workaround	No hay observaciones.
tiempo de respuesta	turnaround time	No hay observaciones.

Tabla 1 - Traducciones Específicas

Definiciones, Acrónimos y Abreviaturas			
Tipo de Término	**Idioma del Acrónimo**	**Término**	**Descripción**
ACRÓNIMO	Español	AVF	**A**nálisis de **V**alores **F**rontera

Definiciones, Acrónimos y Abreviaturas			
Tipo de Término	**Idioma del Acrónimo**	**Término**	**Descripción**
ACRÓNIMO	Inglés	BVA	**B**oundary **V**alue **A**nalysis
ACRÓNIMO	Español	CVDS	**C**iclo de **V**ida de **D**esarrollo de **S**oftware
ABREVIATURA	Español	N/A	**N**o **A**plica
ABREVIATURA	Inglés	SDLC	**S**oftware **D**evelopment **L**ife**C**ycle

Tabla 2 – Definiciones, Acrónimos y Abreviaturas

Notas
Modelo Ciclo de Vida de Desarrollo de Software (MCVDS)

Tabla 3 – Notas

0. Introducción a Este Programa de Estudio

0.1 Objetivo de este Documento

Este programa de estudio constituye la base para la Cualificación Internacional de Pruebas de Software de Nivel Avanzado para Analista de Prueba. El ISTQB® ofrece el presente programa de estudio:

1. a los Comités Nacionales, para que lo traduzcan a su idioma local y para acreditar a los proveedores de formación. Los Comités Nacionales podrán adaptar el programa de estudio a las necesidades específicas de su idioma y modificar las referencias para adaptarlas a sus publicaciones locales.
2. a los Comités de Exámenes, para que puedan crear preguntas de examen en su idioma local que se adapten a los objetivos de aprendizaje de cada programa de estudio.
3. a los proveedores de formación, para que elaboren los materiales didácticos y determinen las metodologías de enseñanza apropiadas.
4. a los candidatos a la certificación, para que se preparen para el examen (como parte de un curso de formación o de manera independiente).
5. a la comunidad internacional de ingeniería de sistemas y software, para que la actividad de pruebas de software y sistemas avance, y como referencia para la elaboración de libros y artículos.

El ISTQB® podrá autorizar que otras entidades utilicen este programa de estudio con otros fines, siempre y cuando soliciten y obtengan la correspondiente autorización previa por escrito.

0.2 El Probador Certificado Nivel Avanzado en Pruebas de Software

El Nivel Avanzado consta de tres programas de estudio independientes.

- Jefe de Prueba.
- Analista de Prueba.
- Analista de Prueba Técnicas.

El "ISTQB® Advanced Level Overview 2019" es un documento específico [ISTQB_AL_OVIEW] que incluye la siguiente información:

- Resultados de negocio para cada programa de estudio.
- Matriz que muestra la trazabilidad entre los resultados de negocio y los objetivos de aprendizaje.
- Resumen de cada programa de estudios.
- Las relaciones entre el programa de estudios.

0.3 Objetivos de Aprendizaje Sujetos a Examen y Niveles Cognitivos de Conocimiento

Los objetivos de aprendizaje respaldan los resultados de negocio y se utilizan para elaborar el examen para lograr la Certificación de Analista de Prueba de Nivel Avanzado.

Los niveles de conocimiento de los objetivos de aprendizaje específicos en los niveles K2, K3 y K4 se muestran al principio de cada capítulo y se clasifican de la siguiente manera:

- K2: Comprender.
- K3: Aplicar.
- K4: Analizar.

Se recordarán (K1) las definiciones de todos los términos enumerados como palabras clave inmediatamente a continuación de los títulos de los capítulos, aunque no se mencionen explícitamente en los objetivos de aprendizaje.

0.4 El Examen de Analista de Prueba de Nivel Avanzado

El examen de Analista de Prueba de Nivel Avanzado se basará en este programa de estudio. Las respuestas a las preguntas del examen pueden requerir el uso de materiales basados en más de una sección de este programa de estudio. Todas las secciones del programa de estudio pueden formar parte del examen, excepto la introducción y los apéndices. Los estándares, libros y otros programas de estudio ISTQB® se incluyen como referencias, pero su contenido no es evaluable más allá de lo que se resume en este programa de estudio en sí mismo a partir de dichos estándares, libros y otros programas de estudio ISTQB®.

El formato del examen es de selección múltiple. Tiene 40 preguntas. Para aprobar el examen, al menos el 65% de las preguntas deben ser respondidas correctamente.

Los exámenes pueden realizarse como parte de un curso de formación acreditado o de forma independiente (por ejemplo, en un centro de exámenes o en un examen público). La realización de un curso de formación acreditado no es un requisito previo para el examen.

0.5 Requisitos de Acceso al Examen

Es necesario haber obtenido el certificado de Probar Certificado de Nivel Básico antes de realizar el examen de certificación de Analista de Prueba de Nivel Avanzado.

0.6 Expectativas de Experiencia

Ninguno de los objetivos de aprendizaje del programa de estudio de Analista de Prueba Avanzado presupone que se dispone de experiencia específica.

0.7 Acreditación de Cursos

Un comité miembro de ISTQB® puede acreditar a los proveedores de formación cuyo material de curso siga este programa de estudio. Los proveedores de formación deben obtener las directrices de acreditación del Comité Miembro o del organismo que realiza la acreditación. Un curso acreditado es reconocido como conforme a este programa de estudio, y se le permite tener un examen ISTQB® como parte del curso.

0.8 Nivel de Detalle del Programa de Estudio

El nivel de detalle de este programa de estudio permite cursos y exámenes consistentes a nivel internacional. Para lograr este objetivo, el programa de estudio incluye:

- Objetivos de formación generales que describen el objetivo del Analista de Prueba de Nivel Avanzado.
- Una lista de elementos que los participantes deben ser capaces de recordar.
- Objetivos de aprendizaje de cada área de conocimiento, describiendo el resultado del aprendizaje cognitivo que se debe alcanzar.
- Una descripción de los conceptos clave, incluidas referencias a fuentes como la bibliografía o normas aceptadas

El contenido del programa de estudio no es una descripción de todo el área de conocimiento; refleja el nivel de detalle que se cubrirá en los cursos de formación de nivel avanzado. Se centra en material que puede aplicarse a todos los proyectos de software, incluidos los proyectos ágiles. El plan de estudio no contiene ningún objetivo de aprendizaje específico relacionado con ningún ciclo de vida de desarrollo

de software (CVDS) en particular, pero sí analiza cómo se aplican estos conceptos en proyectos ágiles, otros tipos de ciclos de vida iterativos e incrementales, y en ciclos de vida secuenciales.

0.9 Cómo Está Organizado este Programa de Estudio

Hay seis capítulos con contenido examinable. En el encabezamiento de cada capítulo se especifica el tiempo mínimo lectivo y de ejercicios para el capítulo; no se indican los tiempos por debajo del nivel del capítulo. Para los cursos de formación acreditados, el programa de estudios requiere un mínimo de 20 horas y 30 minutos lectivas, distribuidas en los seis capítulos de la siguiente manera:

- Capítulo 01: Tareas del Analista de Prueba en el Proceso de Prueba (150 minutos)
- Capítulo 02: Tareas del Analista de Prueba en la Prueba Basada en el Riesgo (60 minutos)
- Capítulo 03: Técnicas de Prueba (630 minutos)
- Capítulo 04: Prueba de las Características de Calidad del Software (180 minutos)
- Capítulo 05: Revisiones (120 minutos)
- Capítulo 06: Herramientas de Prueba y Automatización (90 minutos)

1. Tareas del Analista de Prueba en el Proceso de Prueba - 150 minutos

Palabras Clave

Español	Inglés
análisis de la prueba	test analysis
base de prueba	test basis
calendario de ejecución de prueba	test execution schedule
caso de prueba de alto nivel	high-level test case
caso de prueba de bajo nivel	low-level test case
condición de prueba	test condition
criterios de salida	exit criteria
datos de prueba	test data
diseño de la prueba	test design
ejecución de la prueba	test execution
implementación de la prueba	test implementation
juego de prueba	test suite
procedimiento de prueba	test procedure
prueba	test

Objetivos de Aprendizaje para "Tareas del Analista de Prueba en el Proceso de Prueba"

1.1 Introducción

No hay objetivos de aprendizaje

1.2 Prueba en el Ciclo de Vida del Desarrollo de Software

AP-1.2.1	**(K2)**	Explicar cómo y por qué el tiempo y el nivel de participación del Analista de Prueba varía cuando se trabaja con diferentes modelos de ciclo de vida de desarrollo de software.

1.3 Análisis de Prueba

AP-1.3.1	**(K2)**	Resumir las tareas correspondientes al Analista de Prueba cuando se llevan a cabo actividades de análisis.

1.4 Diseño de Prueba

AP-1.4.1	**(K2)**	Explicar por qué las condiciones de prueba deben ser comprendidas por los implicados.
AP-1.4.2	**(K4)**	Seleccionar el nivel de diseño adecuado para los casos de prueba (de alto o bajo nivel) para un escenario de proyecto determinado.
AP-1.4.3	**(K2)**	Explicar las cuestiones a considerar en el diseño de los casos de prueba.

1.5 Implementación de Prueba

AP-1.5.1	**(K2)**	Resumir las tareas correspondientes al Analista de Prueba cuando se llevan a cabo actividades de implementación de prueba.

1.6 Ejecución de Prueba

AP-1.6.1	**(K2)**	Resumir las tareas correspondientes al Analista de Prueba cuando se llevan a cabo la ejecución de prueba.

1.1 Introducción

En el Programa de Estudio de Nivel Básico del ISTQB®, se describe el proceso de prueba incluyendo las siguientes actividades:

- Planificación de la prueba.
- Monitorización y control de la prueba (también seguimiento y control de la prueba).
- Análisis de la prueba.
- Diseño de la prueba.
- Implementación de la prueba.
- Ejecución de la prueba.
- Compleción de la prueba.

En este programa de estudio de Analista de Prueba de Nivel Avanzado se consideran más a fondo las actividades que tienen particular relevancia para el Analista de Prueba. Esto proporciona un refinamiento adicional del proceso de prueba para ajustarse mejor a los diferentes modelos de ciclo de vida de desarrollo de software (SDLC por sus siglas en inglés).

Determinar las pruebas adecuadas, diseñarlas, implementarlas y luego ejecutarlas son las principales áreas de atención del Analista de Prueba. Si bien es importante comprender los demás pasos del proceso de prueba, la mayor parte de la labor del Analista de Prueba suele centrarse en las siguientes actividades:

- Análisis de la prueba.
- Diseño de la prueba.
- Implementación de la prueba.
- Ejecución de la prueba.

Las demás actividades del proceso de prueba están descritas adecuadamente en el Nivel Básico y no necesitan mayor elaboración en este nivel.

1.2 La Prueba en el Ciclo de Vida de Desarrollo de Software

Se debe tener en cuenta el Ciclo de Vida de Desarrollo de Software (CVDS, o SDLC por sus siglas en inglés) general al definir una estrategia de prueba. El momento de participación del Analista de Prueba es diferente para los diversos CVDS; la intensidad de la participación, el tiempo requerido, la información disponible y las expectativas también pueden ser muy variadas. El Analista de Prueba debe ser consciente de los tipos de información que debe suministrar a otros roles de la organización relacionados, tales como:

- Ingeniería y gestión de requisitos - revisiones de requisitos.
- Gestión del proyecto - entrada del calendario.
- Gestión de la configuración y del cambio - prueba de verificación de la construcción, control de versiones.
- Desarrollo de software - notificaciones de los defectos detectados.
- Mantenimiento de software - gestión de defectos, tiempo de respuesta (es decir, tiempo para detectar e informar anomalías, y luego tiempo para realizar e informar pruebas de confirmación).
- Soporte técnico - documentación precisa para las soluciones provisionales y los problemas conocidos.
- Elaboración de documentación técnica (por ejemplo, especificaciones de diseño de la base de datos, documentación del entorno de prueba) - aportación a estos documentos, así como revisión técnica de los documentos.

Las actividades de prueba deben estar alineadas con el Ciclo de Vida de Desarrollo de Software elegido cuya naturaleza puede ser secuencial, iterativa, incremental o una mezcla híbrida de estas. Por ejemplo, en el modelo V secuencial, el proceso de prueba aplicado al nivel de prueba del sistema podría alinearse de la siguiente manera:

- La planificación de la prueba de sistema se produce simultáneamente con la planificación del proyecto, la monitorización y el control de la prueba continúa hasta la compleción de la misma. Esto influirá en las aportaciones de calendario proporcionadas por el Analista de Prueba a efectos de gestión de proyectos.
- El análisis y diseño de la prueba de sistema se corresponde con documentos como la especificación de los requisitos del sistema, la especificación del diseño del sistema y la arquitectura (de alto nivel) y la especificación del diseño de los componentes (de bajo nivel).
- La implementación del entorno de prueba de sistema podría comenzar durante el diseño del sistema, aunque el grueso de éste normalmente se produciría simultáneamente con la codificación y la prueba de componente, y el trabajo en las actividades de implementación de las pruebas del sistema se extendería a menudo hasta pocos días antes del comienzo de la ejecución de la prueba de sistema.
- La ejecución de la prueba de sistema comienza cuando se cumplen todos los criterios de entrada de la prueba de sistema (o se renuncia a algunos de ellos), lo que normalmente significa que al menos la prueba de componente y a menudo también la prueba de integración de componentes ha cumplido sus criterios de salida o definiciones de hecho. La ejecución de la prueba de sistema continúa hasta que se cumplen los criterios de salida de la prueba de sistema.
- Las actividades de compleción de la prueba de sistema ocurren después de que se cumplen los criterios de salida de la prueba de sistema.
- Es posible que los modelos iterativos e incrementales no sigan el mismo orden de actividades y que excluyan algunas actividades. Por ejemplo, un modelo iterativo puede utilizar un conjunto reducido de actividades de prueba para cada iteración. El análisis, el diseño, la implementación y la ejecución pueden llevarse a cabo para cada iteración, mientras que la planificación de alto nivel se realiza al principio del proyecto y las tareas de cierre se realizan al final.
- En un proyecto ágil, es común utilizar un proceso menos formalizado y una relación de trabajo mucho más estrecha con los implicados del proyecto que permite que los cambios se produzcan más fácilmente dentro del proyecto. Es posible que no exista una función bien definida de Analista de Prueba. La documentación de las pruebas es menos exhaustiva y la comunicación es más breve y frecuente.
- Los proyectos ágiles implican la realización de pruebas desde el principio. Esto comienza desde el inicio del proyecto a medida que los desarrolladores realizan su trabajo inicial de arquitectura y diseño. Las revisiones pueden no estar formalizadas, pero son continuas a medida que el software evoluciona. Se espera que la participación sea a lo largo de todo el proyecto y se espera que las tareas de los analistas de prueba sean realizadas por el equipo.
- Los modelos iterativos e incrementales van desde el enfoque ágil, en el que se espera un cambio a medida que evoluciona el software, hasta los modelos de desarrollo iterativo/incremental que existen dentro de un modelo V (a veces llamado iterativo embebido). En un modelo iterativo embebido, los analistas de prueba deberían esperar participar en los aspectos de planificación y diseño, pero luego pasarían a desempeñar un papel más interactivo a medida que el software se diseña, desarrolla y prueba.

Cualquiera que sea el CVDS que se utilice, es importante que los analistas de prueba comprendan las expectativas de participación, así como el momento de dicha participación. Hay muchos modelos híbridos en uso, como el modelo iterativo embebido mencionado anteriormente. Los Analistas de Prueba deben determinar su papel más efectivo. Deben trabajar para lograr el momento adecuado de participación, en lugar de depender de la definición de un modelo establecido.

1.3 Análisis de la Prueba

Durante la planificación de la prueba, se define el alcance del proyecto de prueba. Los analistas de prueba utilizan esta definición de alcance para:

- Analizar la base de la prueba
- Identificar los defectos de diversos tipos en la base de la prueba
- Identificar y priorizar las condiciones de prueba y las prestaciones que se van a probar
- Capturar la trazabilidad bidireccional entre cada elemento de la base de prueba y las condiciones de prueba asociadas
- Realizar tareas asociadas a las pruebas basadas en el riesgo (véase el capítulo 2)

Para que los analistas de prueba puedan proceder de forma eficaz con el análisis de la prueba, deberían cumplirse los siguientes criterios de entrada:

- Se dispone de un cuerpo de conocimientos (por ejemplo, requisitos, historias de usuario) que describe el objeto de prueba y que puede servir como base de la prueba (véase [ISTQB_FL_SYL] Sección 1.4.2 para una lista de otras posibles fuentes de base de la prueba).
- Esta base de prueba ha pasado la revisión con resultados razonables y se ha actualizado según ha sido necesario después de la revisión. Obsérvese que si hay que definir casos de prueba de alto nivel (véase la sección 1.4.1), puede que la base de pruebas no necesite todavía definirse completamente. En un proyecto ágil, este ciclo de revisión será iterativo, ya que las historias de usuario se perfeccionan al comienzo de cada iteración.
- Se dispone de un presupuesto y un calendario aprobados para llevar a cabo las tareas de prueba restantes para este objeto de prueba.

En general, las condiciones de la prueba se identifican mediante el análisis de la base de prueba en conjunción con los objetivos de la prueba (tal como se definen en la planificación de la prueba). En algunas situaciones, en las que la documentación puede ser antigua o inexistente, las condiciones de la prueba pueden identificarse mediante el debate con los implicados correspondientes (por ejemplo, en talleres o durante la planificación de la iteración). En un proyecto Agile, los criterios de aceptación que se definen como parte de las historias de usuarios se utilizan a menudo como base para el diseño de la prueba.

Si bien las condiciones de la prueba suelen ser específicas del elemento que se está probando, existen algunas consideraciones estándar para el Analista de Prueba.

- Suele ser aconsejable definir las condiciones de prueba con diferentes niveles de detalle. Inicialmente, se identifican condiciones de alto nivel para definir objetivos generales de la prueba, como la "funcionalidad de la pantalla x". Posteriormente, se identifican condiciones más detalladas como base de casos de prueba específicos, como "la pantalla x rechaza un número de cuenta que le falta un dígito para la longitud correcta". La utilización de este tipo de enfoque jerárquico para definir las condiciones de prueba puede ayudar a garantizar que la cobertura sea suficiente para los elementos de alto nivel. Este enfoque también permite que un Analista de Prueba comience a trabajar en la definición de condiciones de prueba de alto nivel para las historias de usuarios que aún no han sido refinadas.
- Si se han definido los riesgos de producto, entonces las condiciones de prueba que serán necesarias para abordar cada riesgo del producto deben identificarse y trazarse hasta ese elemento de riesgo.

El uso de técnicas de prueba (identificadas en la estrategia y/o el plan de prueba) puede ser útil en el proceso de análisis de la prueba y se puede utilizar para apoyar los siguientes objetivos:

- Identificar condiciones de prueba.
- Reducir la probabilidad de omitir condiciones de prueba importantes.
- Definir condiciones de prueba más precisas y exactas.
- Una vez que se hayan identificado y refinado las condiciones de prueba, se puede llevar a cabo una revisión de esas condiciones con los implicados para asegurar que los requisitos se han comprendido claramente y que las pruebas se ajustan a los objetivos del proyecto.

Al concluir las actividades de análisis de prueba para un área determinada (por ejemplo, una función específica), el Analista de Prueba debe saber qué pruebas específicas deben diseñarse para ese área.

1.4 Diseño de la Prueba

Siguiendo con el alcance determinado durante la planificación de la prueba, el proceso de la prueba continúa a medida que el Analista de Prueba diseña las pruebas que serán implementadas y ejecutadas. El diseño de la prueba incluye las siguientes actividades:

- Determinar en qué áreas de prueba son adecuados los casos de prueba de bajo o alto nivel.
- Determinar la(s) técnica(s) de prueba que permitirá(n) lograr la cobertura de prueba necesaria. Las técnicas que pueden utilizarse se establecen durante la planificación de la prueba.
- Utilizar las técnicas de prueba para diseñar casos de prueba y conjuntos de casos de prueba que cubran las condiciones de prueba identificadas.
- Identificar los datos de prueba necesarios para dar soporte a las condiciones de prueba y los casos de prueba.
- Diseñar el entorno de prueba e identificar cualquier infraestructura necesaria, incluyendo herramientas.
- Capturar la trazabilidad bidireccional (por ejemplo, entre la base de la prueba, las condiciones de la prueba y los casos de prueba).

Se deberían aplicar los criterios de priorización identificados durante el análisis de riesgos y la planificación de la prueba a lo largo de todo el proceso, desde el análisis y el diseño hasta la implementación y la ejecución.

Dependiendo de los tipos de pruebas que se diseñen, uno de los criterios de entrada para el diseño de pruebas puede ser la disponibilidad de herramientas que se utilizarán durante el trabajo de diseño.

Durante el diseño de la prueba, el Analista de Prueba debe tener en cuenta, al menos, lo siguiente:

- Algunos elementos de prueba se abordan mejor definiendo sólo las condiciones de la prueba en lugar de profundizar en la definición de los guiones de prueba, que dan la secuencia de instrucciones necesarias para ejecutar una prueba. En este caso, deben definirse las condiciones de prueba para que sirvan de guía para las pruebas sin guion.
- Los criterios de paso/fallo deben estar claramente identificados.
- Las pruebas deben diseñarse de manera que sean comprensibles para otros probadores, no sólo para el autor. Si el autor no es la persona que ejecuta la prueba, otros probadores tendrán que leer y comprender pruebas previamente especificadas para entender los objetivos de la prueba y la importancia relativa de la misma.
- Las pruebas también deben ser comprensibles para otros implicados, como los desarrolladores, que pueden revisar las pruebas, y los auditores, que pueden tener que aprobar las pruebas.
- Las pruebas deben abarcar todos los tipos de interacción con el objeto de prueba y no deben limitarse a las interacciones de las personas a través de la interfaz visible para el usuario. También pueden incluir, por ejemplo, la interacción con otros sistemas y eventos técnicos o físicos. (véase [IREB_CPRE] para más detalles).
- Las pruebas deben diseñarse para comprobar las interfaces entre los diversos objetos de prueba, así como los comportamientos de los propios objetos.
- El esfuerzo de diseño de las pruebas debe ser priorizado y equilibrado para alinearse con los niveles de riesgo y el valor de negocio.

1.4.1 Casos de Prueba de Bajo y Alto Nivel

Uno de los trabajos del Analista de Prueba es determinar el mejor nivel de diseño de los casos de prueba para una situación determinada. Se cubren los casos de prueba de bajo y alto nivel [ISTQB_FL_SYL]. Algunas de las ventajas y desventajas de su utilización se describen en las siguientes listas:

Los casos de prueba de bajo nivel presentan las siguientes ventajas:

- El personal de prueba inexperto se puede apoyar en la información detallada que se proporciona en el proyecto. Los casos de prueba de bajo nivel proporcionan toda la información y los procedimientos específicos necesarios para que el probador ejecute el caso de prueba (incluidos los requisitos de datos) y verifique los resultados.
- Las pruebas pueden ser repetidas por diferentes probadores y deberían obtener los mismos resultados.
- Se pueden detectar defectos no evidentes en la base de prueba.
- El nivel de detalle permite una verificación independiente de las pruebas, como las auditorías, si fuera necesario.
- Se puede reducir el tiempo dedicado a la implementación de los casos de prueba automatizados.

Los casos de prueba de bajo nivel presentan las siguientes desventajas:

- Pueden requerir un esfuerzo considerable, tanto para su creación como para su mantenimiento.
- Tienden a limitar el ingenio de los probadores durante la ejecución.
- Requieren que la base de la prueba esté bien definida.
- Su trazabilidad para probar las condiciones puede requerir más esfuerzo que con los casos de pruebas de alto nivel.

Los casos de prueba de alto nivel presentan las siguientes ventajas:

- Ofrecen directrices sobre lo que debe probarse y permiten al Analista de Prueba variar los datos reales o incluso el procedimiento que se sigue al ejecutar la prueba.
- Pueden proporcionar una mejor cobertura de riesgos que los casos de prueba de bajo nivel porque varían un poco cada vez que se ejecutan.
- Pueden definirse en una etapa temprana del proceso de requisitos.
- Aprovechan la experiencia del Analista de Prueba tanto con la prueba como con el objeto de la prueba cuando ésta se ejecuta.
- Se pueden definir cuando no se requiere documentación detallada y formal.
- Son más adecuados para su reutilización en diferentes ciclos de prueba cuando se pueden utilizar diferentes datos de prueba.

Los casos de pruebas de alto nivel presentan las siguientes desventajas:

- Son menos reproducibles, lo que dificulta la verificación. Esto se debe a que carecen de la descripción detallada que se encuentra en los casos de prueba de bajo nivel.
- Puede ser necesario un personal de prueba más experimentado para ejecutarlas.
- Cuando se automatiza sobre la base de casos de prueba de alto nivel, la falta de detalles puede dar lugar a la validación de resultados erróneos o a la falta de elementos que deberían ser validados.
- Los casos de prueba de alto nivel pueden utilizarse para desarrollar casos de prueba de bajo nivel cuando los requisitos se vuelven más definidos y estables. En este caso, la creación del caso de prueba se hace de forma secuencial, pasando de nivel alto a nivel bajo y utilizando únicamente los casos de prueba de bajo nivel para su ejecución.

1.4.2 Diseño de Casos de Prueba

Los casos de prueba se diseñan mediante la elaboración y el refinamiento por etapas de las condiciones de prueba identificadas utilizando técnicas de prueba (véase el capítulo 3). Los casos de prueba deben ser repetibles, verificables y trazables hasta la base de la prueba (por ejemplo, los requisitos).

El diseño de la prueba incluye la identificación de lo siguiente:

- Objetivo (es decir, el objetivo observable y medible de la ejecución de la prueba).
- Precondiciones, como los requisitos del proyecto o del entorno de prueba localizado y los planes para su entrega, el estado del sistema antes de la ejecución de la prueba, etc.
- Requisitos de los datos de prueba (tanto los datos de entrada para el caso de prueba como los datos que deben existir en el sistema para que el caso de prueba se ejecute).
- Resultados esperados con criterios explícitos de paso/fallo.
- Postcondiciones, como datos afectados, estado del sistema después de la ejecución de la prueba, disparadores para un procesamiento subsiguiente, etc.

Un desafío especial puede ser la definición del resultado esperado de una prueba. El cálculo manual suele ser tedioso y propenso a los errores; de ser posible, sería preferible encontrar o crear un oráculo de prueba automatizado. En la identificación del resultado esperado, los probadores se preocupan no sólo de los resultados en la pantalla, sino también de los datos y las postcondiciones de entorno. Si la base de prueba está claramente definida, la identificación del resultado correcto, en teoría, debería ser sencilla. Sin embargo, la documentación de la base de prueba puede ser vaga, contradictoria, carente de cobertura de las áreas clave o totalmente inexistente. En esos casos, un Analista de Prueba debe tener conocimientos especializados en la materia o tener acceso a ellos. Además, incluso cuando la base de la prueba está bien especificada, las complejas interacciones de estímulos y respuestas pueden dificultar la definición de los resultados esperados; por consiguiente, es esencial un oráculo de prueba. En un proyecto ágil, el oráculo de prueba puede ser el propietario del producto. La ejecución del caso de prueba sin ninguna forma de determinar la corrección de los resultados podría tener un valor añadido o beneficio muy bajo, generando a menudo informes de fallos inválidos o falsa confianza en el sistema.

Las actividades descritas anteriormente pueden aplicarse a todos los niveles de prueba, aunque la base de prueba variará. Al analizar y diseñar las pruebas, es importante recordar el nivel objetivo de la prueba, así como el objetivo de la misma. Esto ayuda a determinar el nivel de detalle necesario, así como cualquier herramienta que pueda ser necesaria (por ejemplo, controladores y stubs en el nivel de prueba de componente).

Durante el desarrollo de las condiciones de prueba y de los casos de prueba, se suele crear cierta cantidad de documentación, lo que da lugar a productos de trabajo de prueba. En la práctica, el grado en que se documentan los productos de trabajo de prueba varía considerablemente. Esto puede verse afectado por cualquiera de los siguientes factores:

- Riesgos de proyecto (qué debe/no debe ser documentado).
- El valor añadido que la documentación aporta al proyecto.
- Estándares que se deben seguir y/o normativa que se debe cumplir.
- CVDS o el enfoque utilizado (por ejemplo, un enfoque ágil apunta a una documentación "lo suficiente").
- El requisito de trazabilidad desde la base de prueba hasta el análisis y el diseño de prueba.

Dependiendo del alcance de la prueba, el análisis y el diseño de la misma abordan las características de calidad del objeto o los objetos de prueba. La norma ISO 25010 [ISO25010] proporciona una referencia útil. Cuando se prueban sistemas hardware/software, pueden ser necesarias características adicionales.

Los procesos de análisis y diseño de pruebas pueden mejorarse entrelazándolos con revisiones y análisis estáticos. De hecho, la realización del análisis y el diseño de prueba suelen ser una forma de prueba estática porque durante este proceso pueden encontrarse problemas en los documentos de la base de prueba. El análisis y diseño de prueba basado en la especificación de requisitos es una forma excelente de prepararse para una reunión de revisión de requisitos. La lectura de los requisitos para utilizarlos en la creación de pruebas requiere comprender el requisito y poder determinar una forma de

evaluar el cumplimiento del mismo. Esta actividad a menudo descubre requisitos que no son claros, no tienen capacidad de ser probados o no tienen criterios de aceptación definidos. Del mismo modo, los productos de trabajo de prueba, como los casos de prueba, los análisis de riesgo y los planes de prueba, pueden ser objeto de revisiones.

Durante el diseño de la prueba pueden definirse los requisitos detallados de infraestructura de la prueba, aunque en la práctica es posible que no se finalicen hasta la implementación de la prueba. Hay que recordar que la infraestructura de prueba incluye algo más que objetos y programas de prueba. Por ejemplo, los requisitos de infraestructura pueden incluir salas, equipos, personal, software, herramientas, periféricos, equipo de comunicaciones, autorizaciones de usuarios y todos los demás elementos necesarios para ejecutar las pruebas.

Los criterios de salida para el análisis y el diseño de la prueba variarán en función de los parámetros del proyecto, pero todos los puntos tratados en estas dos secciones deben tenerse en cuenta para su inclusión en los criterios de salida definidos. Es importante que los criterios de salida sean mensurables y que se haya facilitado toda la información y preparación necesarias para las etapas subsiguientes.

1.5 Implementación de la Prueba

La implementación de la prueba prepara los productos de prueba necesario para la ejecución de la prueba basada en el análisis y el diseño de la prueba. Incluye las siguientes actividades:

- Crear un calendario de ejecución de pruebas, incluida la asignación de recursos, para permitir el inicio de la ejecución de los casos de prueba (véase [ISTQB_FL_SYL] Sección 5.2.4).
- Organizar las pruebas (tanto manuales como automatizadas) en conjuntos de pruebas y definir el orden de ejecución de estas.
- Crear pruebas automatizadas (o identificar los casos de prueba que serán automatizados por un desarrollador o un ingeniero de automatización).
- Desarrollar procedimientos de prueba.
- Finalizar los datos y los entornos de prueba.
- Actualizar la trazabilidad entre la base de prueba y los productos de prueba, como las condiciones de prueba, los casos de prueba y los juegos de prueba.

Durante la implementación de la prueba, los analistas de pruebas identifican los casos de prueba que pueden agruparse (por ejemplo, todos ellos se relacionan con la prueba de un determinado proceso de negocio de alto nivel), y los organizan en juegos de prueba. Esto permite que los casos de prueba relacionados se ejecuten juntos.

Se crean procedimientos de prueba que finalizan y confirman el orden en que deben ejecutarse las pruebas manuales y automatizadas y los juegos de prueba. Para definir los procedimientos de prueba es necesario identificar cuidadosamente las limitaciones y dependencias que pueden influir en la secuencia de ejecución de la prueba. Los procedimientos de prueba documentan cualquier precondición inicial (por ejemplo, la carga de datos de prueba de un repositorio de datos) y cualquier otra actividad después de la ejecución (por ejemplo, restableciendo el estado del sistema). Si se utiliza una estrategia de prueba basada en el riesgo, el nivel de riesgo puede dictar la orden de ejecución para los casos de prueba. Puede haber otros factores que determinen el orden de ejecución de los casos de prueba, como la disponibilidad de las personas, el equipo y los datos adecuados y la funcionalidad que se ha de probar.

No es raro que el código se entregue por secciones y el esfuerzo de prueba debe coordinarse con la secuencia en la que el software se pone a disposición para la prueba. Particularmente en los modelos de ciclo de vida iterativo e incremental, es importante que el Analista de Prueba se coordine con el equipo de desarrollo para asegurar que el software se entregue para su prueba en un orden que pueda ser objeto de prueba.

Los factores anteriores se tienen en cuenta al crear un calendario de ejecución de prueba.

El nivel de detalle y la complejidad asociada para el trabajo realizado durante la implementación de la prueba pueden verse influidos por el detalle de los casos de prueba y las condiciones de la prueba. En algunos casos se aplican las reglas normativas, y los productos del trabajo de prueba deben

proporcionar evidencia del cumplimiento de las normas aplicables, como la norma DO-178C de los Estados Unidos (en Europa, ED 12C). RTCA DO-178C/ED-12C].

Como se ha indicado anteriormente, se necesitan datos de prueba para la mayoría de las pruebas, y en algunos casos estos conjuntos de datos pueden ser bastante grandes. Durante la implementación, los analistas de prueba crean datos de entrada y de entorno para cargarlos en bases de datos y otros repositorios similares. Estos datos deben ser " adecuados al propósito" para permitir la detección de defectos. Los analistas de prueba también pueden crear datos para ser utilizados con pruebas de automatización basadas en datos y palabras clave (véase la sección 6.2), así como para pruebas manuales.

La implementación de la prueba también se refiere al entorno o entornos de prueba. Durante esta actividad, el o los entornos deben estar completamente configurados y verificados antes de la ejecución de la prueba. Es esencial un entorno de prueba "adecuado al propósito", es decir, el entorno de prueba debe ser capaz de permitir la exposición de los defectos presentes durante la prueba controlada, funcionar normalmente cuando no se producen fallos y reproducir adecuadamente, si es necesario, el entorno de producción o de usuario final para niveles de prueba superiores. Puede ser necesario introducir cambios en el entorno de prueba durante la ejecución de ésta, en función de los cambios imprevistos, los resultados de la prueba u otras consideraciones. Si se producen cambios en el entorno durante la ejecución, es importante evaluar el impacto de los cambios en las pruebas que ya se han realizado.

Durante la ejecución de la prueba, los analistas de prueba deben comprobar que los responsables de la creación y el mantenimiento del entorno de prueba son conocidos y están disponibles, y que todos los productos de prueba, las herramientas de apoyo a la prueba y los procesos asociados están preparados para su uso. Esto incluye la gestión de la configuración, la gestión de defectos y el registro y la gestión de la prueba. Además, los analistas de prueba deben verificar los procedimientos que recopilan datos para evaluar el estado actual en relación con los criterios de salida y el informe de los resultados de prueba.

Es conveniente utilizar un enfoque equilibrado para la implementación de la prueba, tal como se determinó durante la planificación de la misma. Por ejemplo, las estrategias de pruebas analíticas basadas en el riesgo se combinan a menudo con estrategias de pruebas reactivas. En este caso, un porcentaje del esfuerzo de ejecución de la prueba se asigna a pruebas que no siguen guiones predeterminados (sin guion).

Las pruebas sin guion no deben ser aleatorias o sin objetivo, ya que pueden ser impredecibles en cuanto a su duración y cobertura, y dar lugar a un bajo rendimiento en términos de defectos. Más bien, deben realizarse en sesiones de tiempo limitado, cada una de las cuales debe recibir una dirección inicial por un contrato de prueba, pero con la libertad de apartarse de las prescripciones del contrato si en el curso de la sesión se descubren oportunidades de prueba potencialmente más productivas. A lo largo de los años, los probadores han desarrollado una variedad de técnicas de prueba basadas en la experiencia, como los ataques [Whittaker03], la predicción de errores [Myers11] y las pruebas exploratorias [Whittaker09]. El análisis, el diseño y la ejecución de las pruebas siguen teniendo lugar, pero principalmente durante la ejecución de la prueba.

Cuando se siguen esas estrategias de pruebas reactivas, los resultados de cada prueba influyen en el análisis, diseño e implementación de las pruebas subsiguientes. Aunque estas estrategias son ligeras y a menudo eficaces para encontrar defectos, existen algunos inconvenientes, entre los que se incluyen los siguientes:

- Es necesaria pericia por parte del Analista de Prueba.
- Puede ser difícil predecir la duración.
- Puede ser difícil hacer un seguimiento de la cobertura.
- Se puede perder la repetibilidad sin una buena documentación o apoyo de herramientas.

1.6 Ejecución de la Prueba

La ejecución de la prueba se realiza de acuerdo con el calendario de ejecución de la prueba e incluye las siguientes tareas: (véase [ISTQB_FL_SYL])

- Ejecutar pruebas manuales, incluyendo pruebas exploratorias.
- Ejecutar pruebas automatizadas.
- Comparar los resultados reales con los resultados esperados.
- Analizar las anomalías para establecer sus posibles causas.
- Informar defectos en base a los fallos observados.
- Registrar el resultado de la ejecución de la prueba.
- Actualizar la trazabilidad entre la base de prueba y el producto de prueba para tener en cuenta los resultados de la prueba.
- Ejecutar pruebas de regresión.

Las tareas de ejecución de la prueba enumeradas anteriormente pueden ser realizadas por el probador o por el Analista de Prueba.

Las siguientes son las tareas adicionales habituales que puede realizar el Analista de Prueba:

- Reconocer grupos de defectos que puedan indicar la necesidad de realizar más pruebas de una parte determinada del objeto de prueba.
- Hacer sugerencias para futuras sesiones de pruebas exploratorias basadas en los hallazgos de las pruebas exploratorias.
- Identificar nuevos riesgos a partir de la información obtenida al realizar las tareas de ejecución de la prueba.
- Hacer sugerencias para mejorar cualquiera de los productos de trabajo de la actividad de implementación de la prueba (por ejemplo, mejoras en los procedimientos de prueba).

2. Tareas del Analista de Prueba en la Prueba Basada en el Riesgo - 60 minutos

Palabras Clave

Español	Inglés
riesgo de producto	product risk
identificación del riesgo	risk identification
nivel de riesgo	risk level
mitigación del riesgo	risk mitigation
prueba basada en el riesgo	risk-based testing

Objetivos de Aprendizaje para "Tareas del Analista de Prueba en la Prueba Basada en el Riesgo"

Tareas del Analista de Prueba en la Prueba Basada en el Riesgo

AP-2.1.1	**(K3)**	Para una situación específica, participar en la identificación de riesgos, realizar una evaluación de riesgos y proponer una mitigación de riesgos adecuada.

2.1 Introducción

Los Jefes de Prueba, a menudo, tienen la responsabilidad general de establecer y gestionar una estrategia de prueba basada en el riesgo. Por lo general, solicitarán la participación de un Analista de Prueba para garantizar que el enfoque basado en el riesgo se aplique correctamente.

Los Analistas de Prueba deben involucrarse de forma activa en las siguientes tareas de prueba basada en el riesgo:

- Identificación del riesgo.
- Evaluación del riesgo.
- Mitigación del riesgo.

Estas tareas se realizan de forma iterativa en todo el CVDS para hacer frente a los riesgos emergentes, prioridades cambiantes y para evaluar y comunicar regularmente el estado de los riesgos (véase [vanVeenendaal12] y [Black02] para más detalles). En un proyecto ágil, las tres tareas se combinan, a menudo, en una llamada sesión de riesgo con enfoque en una iteración o una entrega.

Los analistas de prueba deben trabajar dentro del marco de la prueba basada en el riesgo establecido para el proyecto por el Jefe de Prueba. Deberían aportar sus conocimientos sobre los riesgos del dominio del negocio que son inherentes al proyecto, tales como los riesgos relacionados con la seguridad, los asuntos de interés del negocio y económicos, y los factores políticos, entre otros.

2.2 Identificación del Riesgo

Si se recurre a la muestra más amplia posible de implicados, es muy probable que el proceso de identificación de riesgos detecte el mayor número posible de riesgos importantes.

Los analistas de prueba suelen poseer conocimientos únicos sobre el dominio del negocio particular del sistema que se está probando. Esto significa que están particularmente bien preparados para las siguientes tareas:

- Realizar entrevistas de expertos con los expertos y usuarios del dominio.
- Realizar evaluaciones independientes.
- Utilizar plantillas de riesgo.
- Participar en talleres de riesgo.
- Participar en sesiones de tormenta de ideas con usuarios potenciales y reales.
- Definir las listas de comprobación de prueba.
- Recurrir a la experiencia pasada con sistemas o proyectos similares.

En particular, los analistas de pruebas deben trabajar en estrecha colaboración con los usuarios y otros expertos del dominio (por ejemplo, ingenieros de requisitos, analistas de negocio) para determinar las

áreas de riesgo de negocio que deben abordarse durante la prueba. En los proyectos ágiles, esta estrecha relación con los implicados permite que la identificación del riesgo se realice de forma regular, por ejemplo, durante las reuniones de planificación de la iteración.

Entre los ejemplos de riesgos que podrían identificarse en un proyecto se incluyen:

- Problemas con la corrección funcional, por ejemplo, cálculos incorrectos.
- Problemas de usabilidad, por ejemplo, escasez de métodos abreviados de teclado.
- Problemas de portabilidad, por ejemplo, la incapacidad de instalar una aplicación en determinadas plataformas.

2.3 Evaluación del Riesgo

Mientras que la identificación del riesgo consiste en identificar el mayor número posible de riesgos relevantes, la evaluación del riesgo es el estudio de esos riesgos identificados. En concreto, consiste en clasificar cada riesgo y determinar la probabilidad y el impacto asociados a cada uno de ellos.

Determinar el nivel de riesgo, normalmente, implica evaluar para cada elemento de riesgo, la probabilidad de que se produzca y el impacto en el momento de producirse. La probabilidad de que ocurra se suele interpretar como la probabilidad de que el posible problema pueda existir en el sistema sometido a prueba y se observe cuando el sistema esté en producción. Los Analistas de Pruebas Técnicas deben contribuir a encontrar y comprender la probabilidad potencial para cada elemento de riesgo, mientras que los Analistas de Pruebas contribuyen a comprender el posible impacto sobre el negocio del problema en caso de que ocurra (en los proyectos Agile esta distinción basada en la función puede ser menos fuerte).

El impacto en el momento de ocurrir se interpreta a menudo como la gravedad del efecto en los usuarios, clientes u otros implicados. En otras palabras, surge del riesgo de negocio. Los analistas de prueba deberían contribuir a identificar y evaluar el posible dominio de negocio o el impacto en el usuario para cada elemento de riesgo. Entre los factores que influyen en el riesgo de negocio se encuentran los siguientes:

- Frecuencia de uso de la prestación afectada.
- Pérdida de negocio.
- Daños financieros.
- Pérdidas o responsabilidad ecológica o social.
- Sanciones legales civiles o penales.
- Asuntos de interés relativos a la seguridad.
- Multas, pérdida de la licencia.
- Falta de soluciones provisionales razonables si la gente no puede trabajar más.
- Visibilidad de la prestación.
- La visibilidad del fracaso que conduce a la publicidad negativa y al posible daño de la imagen.
- Pérdida de clientes.

Dada la información sobre riesgos disponible, los analistas de prueba deben establecer los niveles de riesgo de negocio según las directrices proporcionadas por un Jefe de Prueba. Éstos podrían clasificarse con términos (por ejemplo, bajo, medio, alto) o utilizar números y/o colores.

A menos que exista una forma de medir el riesgo en una escala definida, no puede ser una verdadera medida cuantitativa. Se pueden asignar números al valor cualitativo, pero eso no lo convierte en una verdadera medida cuantitativa. Por ejemplo, los encargados de las pruebas pueden determinar que el riesgo de negocio debe clasificarse en una escala numérica determinada (por ejemplo, del 1 al 5 para la probabilidad y el impacto). Una vez que se han asignado la probabilidad y el impacto, estos valores pueden utilizarse para determinar la clasificación general del riesgo para cada elemento de riesgo. Esa calificación general se utiliza luego para priorizar las actividades de mitigación de riesgos [vanVeenendaal12].

2.4 Mitigación del Riesgo

Durante el proyecto, los analistas de prueba deben tratar de hacer lo siguiente:

- Reducir el riesgo de producto diseñando casos de prueba eficaces que demuestren sin ambigüedades si las pruebas pasan o fallan, y participando en revisiones de productos de trabajo de software como requisitos, diseños y documentación de usuario.
- Implementar actividades adecuadas de mitigación del riesgo identificadas en la estrategia y el plan de prueba (por ejemplo, probar un proceso de negocio de alto riesgo utilizando técnicas de prueba específicas).
- Reevaluar los riesgos conocidos sobre la base de la información adicional reunida a medida que se desarrolla el proyecto, ajustando la probabilidad, el impacto o ambos, según corresponda.
- Identificar nuevos riesgos a partir de la información obtenida durante la prueba.

Cuando se habla de un riesgo de producto, las pruebas contribuyen de manera esencial a mitigar esos riesgos. Al encontrar los defectos, los probadores reducen el riesgo al dar a conocer los defectos y las oportunidades de tratar los defectos antes de su entrega. Si los probadores no encuentran defectos, las pruebas reducen el riesgo al proporcionar pruebas de que, en determinadas condiciones (es decir, las condiciones probadas), el sistema funciona correctamente. Los analistas de prueba ayudan a determinar las opciones de mitigación de riesgos investigando oportunidades para reunir datos de prueba precisos, crear y probar escenarios de usuario realistas y realizar o supervisar estudios de usabilidad, entre otros.

2.5 Priorización de las Pruebas

El nivel de riesgo también se utiliza para priorizar las pruebas. Un Analista de Prueba podría determinar que existe un alto riesgo en el área de la exactitud de las transacciones en un sistema de contabilidad. En consecuencia, para mitigar el riesgo, el probador puede trabajar con otros expertos en el dominio del negocio para reunir un sólido conjunto de datos de muestra que puedan ser procesados y verificados en cuanto a su exactitud. Del mismo modo, un Analista de Prueba puede determinar que los problemas de utilización constituyen un riesgo importante para un nuevo objeto de prueba. En lugar de esperar a que una prueba de aceptación de usuario descubra cualquier problema, el Analista de Prueba podría dar prioridad a una prueba de usabilidad temprana basada en un prototipo para ayudar a identificar y resolver los problemas de diseño de usabilidad antes de la prueba de aceptación del usuario. Esta priorización debe considerarse lo antes posible en las etapas de planificación, de modo que el calendario pueda contemplar las pruebas necesarias en el momento oportuno.

En algunos casos, todas las pruebas de mayor riesgo se realizan antes que las de menor riesgo, y las pruebas se ejecutan en estricto orden de riesgo (llamado "profundidad-primero"); en otros casos, se utiliza un enfoque de muestreo para seleccionar una muestra de pruebas de todos los riesgos identificados utilizando el nivel de riesgo para ponderar la selección y, al mismo tiempo, asegurar la cobertura de cada riesgo por lo menos una vez (llamado "amplitud-primero").

Independientemente de que las pruebas basadas en el riesgo se realicen primero en profundidad o primero en anchura, es posible que el tiempo asignado a las pruebas se consuma sin que se realicen todas las pruebas. La prueba basada en el riesgo permite a los probadores informar a la dirección sobre el nivel de riesgo restante en este momento, y permite a la dirección decidir si ampliar la prueba o transferir el riesgo restante a los usuarios, clientes, servicio de asistencia/soporte técnico y/o personal operativo.

2.5.1 Ajuste de la Prueba para Futuros Ciclos de Prueba

La evaluación del riesgo no es una actividad que se realiza una sola vez antes del comienzo de la implementación de la prueba; es un proceso continuo. Cada ciclo de prueba previsto en el futuro debe someterse a un nuevo análisis de riesgos para tener en cuenta factores como:

- Todo riesgo de producto nuevo o que haya cambiado de forma significativa.
- Las áreas inestables o propensas a fallar descubiertas durante la prueba.
- Riesgos provenientes de defectos corregidos.

- Defectos típicos encontrados durante la prueba.
- Áreas que han sido sometidas a prueba (baja cobertura de pruebas).

3. Técnicas de Prueba - 630 minutos

Palabras Clave

Español	Inglés
técnica de prueba de caja negra	black-box test technique
análisis de valores frontera	boundary value analysis
prueba basada en listas de comprobación	checklist-based testing
técnica del árbol de clasificación	classification tree technique
prueba de tabla de decisión	decision table testing
taxonomía de defectos	defect taxonomy
técnica de prueba basada en defectos	defect-based test technique
partición de equivalencia	equivalence partitioning
predicción de errores	error guessing
prueba basada en la experiencia	experience-based testing
técnica de prueba basada en la experiencia	experience-based test technique
prueba exploratoria	exploratory testing
pruebas de a pares de elementos	pairwise testing
prueba de transición de estado	state transition testing
contrato de prueba	test charter
prueba de caso de uso	use case testing

Objetivos de Aprendizaje para “Técnicas de Prueba”

3.1 Introducción		
		No hay objetivos de aprendizaje
3.2 Técnicas de Prueba de Caja Negra		
AP-3.2.1	**(K4)**	Analizar uno o más elementos concretos de una especificación y diseñar casos de prueba aplicando la técnica de prueba de partición de equivalencia.
AP-3.2.2	**(K4)**	Analizar uno o más elementos concretos de una especificación y diseñar casos de prueba aplicando la técnica de prueba de análisis de valores frontera.
AP-3.2.3	**(K4)**	Analizar uno o más elementos concretos de una especificación y diseñar casos de prueba aplicando la técnica de prueba de tabla de decisión.
AP-3.2.4	**(K4)**	Analizar uno o más elementos concretos de una especificación y diseñar casos de prueba aplicando la técnica de prueba de transición de estado.
AP-3.2.5	**(K2)**	Explicar cómo los diagramas de árbol de clasificación dan soporte a las técnicas de prueba.
AP-3.2.6	**(K4)**	Analizar uno o más elementos concretos de una especificación y diseñar casos de prueba aplicando la técnica de prueba de a pares de elementos.
AP-3.2.7	**(K4)**	Analizar uno o más elementos concretos de una especificación y diseñar casos de prueba aplicando la técnica de prueba de casos de uso.
AP-3.2.8	**(K4)**	Analizar un sistema, o su especificación de requisitos, para determinar los tipos de defectos que probablemente se puedan detectar y seleccionar la(s) técnica(s) de prueba de caja negra adecuada(s)
3.3 Técnicas de Prueba Basadas en la Experiencia		
AP-3.3.1	**(K2)**	Explicar los principios de las técnicas de prueba basadas en la experiencia, los beneficios y desventajas comparadas con las técnicas de prueba de caja negra y basadas en defectos.

AP-3.3.2	**(K3)**	Identificar las pruebas exploratorias a partir de un escenario concreto.
AP-3.3.3	**(K2)**	Describir la aplicación de las técnicas de prueba basadas en defectos y diferenciar su uso de las técnicas de prueba de caja negra.
3.4 Aplicación de las Técnicas de Prueba más Adecuadas		
AP-3.4.1	**(K4)**	Determinar, para una situación de proyecto determinada, qué técnicas de prueba de caja negra o basadas en la experiencia deben aplicarse para lograr objetivos específicos.

3.1 Introducción

Las técnicas de prueba consideradas en este capítulo se dividen en las siguientes categorías:

- Caja negra.
- Basada en la experiencia.

Estas técnicas son complementarias y pueden utilizarse según convenga para cualquier actividad de prueba determinada, independientemente del nivel de prueba que se realice.

Obsérvese que ambas categorías de técnicas pueden utilizarse para probar características de calidad funcionales y no funcionales. La comprobación de las características del software se aborda en el capítulo siguiente.

Las técnicas de prueba que se tratan en estas secciones pueden centrarse, principalmente, en determinar los datos de prueba óptimos (p. ej., de las particiones de equivalencia) o en obtener procedimientos de prueba (p. ej., de los modelos de estado). Es común combinar técnicas para crear casos de prueba completos.

3.2 Técnicas de Prueba de Caja Negra

En el Programa de Estudio del Nivel Básico de ISTQB® [ISTQB_FL_SYL] se presenta una introducción a las técnicas de prueba de caja negra.

Entre las características comunes de las técnicas de prueba de caja negra se incluyen:

- Los modelos, por ejemplo, los diagramas de estado y las tablas de decisión, se crean durante el diseño de la prueba de acuerdo con la técnica de la prueba.
- Las condiciones de prueba se obtienen sistemáticamente de estos modelos.

Las técnicas de prueba generalmente proporcionan criterios de cobertura, que pueden utilizarse para medir el diseño de la prueba y las actividades de ejecución de la prueba. El cumplimiento completo de los criterios de cobertura no significa que todo el conjunto de pruebas esté completo, sino más bien que el modelo ya no sugiere ninguna prueba adicional para aumentar la cobertura basada en esa técnica.

La prueba de caja negra suele basarse en algún tipo de documentación de especificación, como una especificación de requisitos del sistema o historias de usuarios. Dado que la documentación de especificación debe describir el comportamiento del sistema, particularmente en el área de la funcionalidad, la obtención de pruebas a partir de los requisitos suele formar parte de las pruebas del comportamiento del sistema. En algunos casos puede no haber documentación de especificación, pero hay requisitos implícitos, como la sustitución de la funcionalidad de un sistema legado.

Existen varias técnicas de prueba de caja negra. Estas técnicas se dirigen a diferentes tipos de software y escenarios. En las siguientes secciones se expone la aplicabilidad de cada técnica, algunas limitaciones y dificultades que puede experimentar el Analista de Prueba, el método de medición de la cobertura y los tipos de defectos a los que se dirige.

Por favor, para más detalles, consultar [ISO29119-4], [Bath14], [Beizer95], [Black07], [Black09], [Copeland04], [Craig02], [Koomen06], y [Myers11].

3.2.1 Partición de Equivalencia

La partición de equivalencia (PE) es una técnica utilizada para reducir el número de casos de prueba que se requiere para comprobar eficazmente el tratamiento de las entradas, las salidas, los valores internos y los valores relacionados con el tiempo. La partición se utiliza para crear clases de equivalencia (a menudo llamadas particiones de equivalencia) que se crean a partir de conjuntos de valores que deben ser procesados de la misma manera. Al seleccionar un valor representativo de una partición, se asume la cobertura de todos los elementos de la misma partición.

Aplicabilidad

Esta técnica se puede aplicar en cualquier nivel de prueba y es adecuada cuando se espera que todos los miembros de un conjunto de valores que se van a probar sean tratados de la misma forma y cuando los conjuntos de valores utilizados por la aplicación no interactúan. La selección de conjuntos de valores es aplicable a las particiones válidas e inválidas (es decir, las particiones que contienen valores que deberían considerarse inválidos para el software que se está probando).

Estos valores se deben considerar en las siguientes categorías:

- Valores en un rango continuo. Por ejemplo, la expresión $0 < x < 5000$ representa el rango continuo para la variable "x" cuya partición válida está entre 1 y 4999. Un valor representativo válido en este rango continuo puede ser, por ejemplo, 3249.
- Valores discretos. Por ejemplo, los colores "rojo, azul, verde, amarillo" son todos particiones válidas y sólo hay un valor posible por partición.

La PE es más fuerte cuando se utiliza en combinación con el análisis de valores frontera que expande los valores de prueba para incluir los valores situados en los bordes de las particiones. PE, utilizando valores de las particiones válidas, es una técnica comúnmente utilizada para pruebas de humo de una nueva construcción o versión, ya que determina rápidamente si la funcionalidad básica está funcionando.

Limitaciones/Dificultades

Si la suposición es incorrecta y los valores en la partición no son tratados exactamente de la misma manera, esta técnica puede omitir defectos. También es importante seleccionar cuidadosamente las particiones. Por ejemplo, un campo de entrada que acepte números positivos y negativos se probaría mejor como dos particiones válidas, una para los números positivos y otra para los negativos, debido a la probabilidad de un tratamiento diferente. Dependiendo de si se permite o no el cero, ésta podría convertirse también en otra partición (teniendo en cuenta que el cero no es ni positivo ni negativo). Es importante que un Analista de Prueba comprenda el procesamiento subyacente para determinar la mejor partición de los valores. Esto puede requerir apoyo para comprender el diseño del código.

Cobertura

La cobertura se determina tomando el número de particiones para las que se ha probado un valor y dividiendo ese número por el número de particiones que se han identificado. La cobertura de la PE se expresa entonces como un porcentaje. El uso de valores múltiples para una sola partición no aumenta el porcentaje de cobertura.

Tipos de Defectos

Un Analista de Prueba utiliza esta técnica para encontrar defectos en el tratamiento de diversos valores de los datos.

3.2.2 Análisis de Valores Frontera

El análisis de valores frontera (AVF) se utiliza para probar el tratamiento adecuado de los valores que existen en las fronteras de las particiones de equivalencia. Hay dos formas de aproximarse al BVA: prueba de dos o tres valores. Con la prueba de dos valores, se utiliza el valor frontera (en la frontera) y el valor que está justo fuera de la frontera (por el menor incremento posible). Por ejemplo, si la partición incluyera los valores 1 a 10 en incrementos de 0,5, los dos valores de prueba para la frontera superior serían 10 y 10,5. Los valores de prueba para la frontera inferior serían 1 y 0,5. Las fronteras están definidas por los valores máximo y mínimo en la partición de equivalencia definida.

Para la prueba de la frontera con tres valores, se utilizan los valores antes, sobre y después de la frontera. En el ejemplo anterior, la prueba de la frontera superior incluiría los valores 9,5, 10 y 10,5. La prueba de la frontera inferior incluiría los valores 1,5, 1 y 0,5. La decisión de utilizar dos o tres valores límite debería basarse en el riesgo asociado al elemento que se está probando, utilizándose el criterio de los tres valores frontera para los elementos de mayor riesgo.

Aplicabilidad

Esta técnica es aplicable a cualquier nivel de prueba y es adecuada cuando existen particiones de equivalencia ordenadas. Por esta razón, la técnica BVA se realiza a menudo junto con la técnica EP. Las particiones de equivalencia ordenadas son necesarias debido al concepto de estar dentro y fuera de la frontera. Por ejemplo, un rango de números es una partición ordenada. Una partición que consiste en todos los objetos rectangulares no es una partición ordenada y no tiene valores frontera. Además de los rangos de números, el análisis de valores límite puede aplicarse a lo siguiente:

- Atributos numéricos de variables no numéricas (por ejemplo, longitud).
- Los bucles, incluidos los bucles de los diagramas de estado, de los casos de uso y de las estructuras de datos almacenados, como las matrices.
- Objetos físicos (incluyendo la memoria).
- Actividades determinadas por el tiempo.

Limitaciones/Dificultades

Dado que la precisión de esta técnica depende de la identificación exacta de las particiones de equivalencia para identificar correctamente las fronteras, está sujeta a las mismas limitaciones y dificultades. El Analista de Prueba también debe ser consciente de los incrementos de los valores válidos e inválidos para poder determinar con precisión los valores que se van a probar. Sólo se pueden utilizar particiones ordenadas para el análisis de los valores frontera, pero esto no se limita al rango de entradas válidas. Por ejemplo, cuando se prueba el número de celdas que admite una hoja de cálculo, hay una partición que contiene el número de celdas hasta e incluyendo el máximo permitido de celdas (la frontera) y otra partición que comienza con una celda por encima del máximo (por encima de la frontera).

Cobertura

La cobertura se determina tomando el número de condiciones frontera que se prueban y dividiéndolo por el número de condiciones frontera identificadas (ya sea utilizando el método de dos o tres valores). La cobertura se expresa en forma de porcentaje.

Tipos de Defectos

El análisis de valores frontera detecta de manera fiable el desplazamiento u omisión de fronteras, y puede encontrar casos de fronteras adicionales. Esta técnica encuentra defectos en el tratamiento de los valores frontera, en particular errores con una lógica menor o mayor que (es decir, desplazamiento). También se puede utilizar para detectar defectos no funcionales; por ejemplo, un sistema admite 10.000 usuarios concurrentes, pero no 10.001.

3.2.3 Prueba de Tabla de Decisión

Las tablas de decisión permiten capturar las condiciones que trata el software sometido a prueba y evaluar los resultados de la aplicación de diferentes valores verdaderos o falsos a esas condiciones. Los analistas de prueba pueden utilizar las tablas de decisión para capturar las reglas de decisión que se aplican al software sometido a prueba.

Para utilizar esta técnica, el Analista de Prueba configura la tabla de decisión inicial como una tabla completa en la que el número de columnas es 2 elevado a la potencia correspondiente al número de condiciones (2^n donde n es el número de condiciones). Por ejemplo, la tabla de condiciones iniciales para tres condiciones tendría ocho columnas (2^3).

Cada columna de la tabla representa una regla de decisión. Una regla de decisión típica (genérica) podría ser "Si la condición A es verdadera y la condición B es verdadera y la condición C es falsa, entonces se espera la acción X".

En la prueba de la tabla de decisión se pueden aplicar dos enfoques, dependiendo de lo que se deba cubrir:

- Cobertura de las combinaciones de condiciones que componen las reglas.
- Cobertura de las condiciones individuales.

Cobertura de Combinaciones de Condiciones.

El uso "clásico" de las tablas de decisión es para probar las reglas de decisión que consisten en combinaciones de condiciones.

Cuando se intenta probar todas las combinaciones posibles, las tablas de decisión pueden llegar a ser muy extensas. Un método para reducir sistemáticamente el número de combinaciones de todas las posibles a las que se desvían significativamente de otras combinaciones se denomina prueba de tabla de decisión colapsada [Copeland04]. Cuando se utiliza esta técnica, las combinaciones se reducen a aquellas que producirán resultados diferentes, eliminando de la tabla de decisión los conjuntos de condiciones que no son relevantes para el resultado. Estas reglas se consideran redundantes y se marcan en la tabla de decisiones como "indiferentes". Además, se eliminan las reglas de decisión que contienen combinaciones de condiciones que no son viables. Por ejemplo, si una máquina de billetes aplica diferentes reglas de decisión en función de las condiciones "antes de las 9 de la mañana" y "después de las 5 de la tarde", entonces no es factible definir una regla de decisión en la que ambas condiciones sean verdaderas (en el mismo día). Las reglas de decisión inviables se eliminan de la tabla. La eliminación de las redundancias o inviabilidades da como resultado una tabla de decisión parcialmente colapsada. Una tabla totalmente colapsada ha eliminado ambas.

Cuando se cubren combinaciones de condiciones, un Analista de Prueba intenta crear una tabla de decisión colapsada. Si un Analista de Prueba considera que el número de casos de prueba derivados de la tabla colapsada sigue siendo demasiado grande, se realiza una selección basada en el riesgo.

Cobertura de los Resultados de las Condiciones Individuales

El objetivo de este enfoque es asegurar que el resultado verdadero y falso de cada condición sea probado individualmente. La primera regla de decisión que se debe tener en cuenta define el valor para que todas las condiciones individuales sean iguales (por ejemplo, verdadero). Esta regla de decisión se introduce en la tabla de decisiones junto con la acción esperada. La definición de la segunda regla de decisión implica cambiar el valor de la primera condición al estado opuesto (es decir, falso). Esta segunda regla de decisión también se introduce en la tabla de decisión junto con la acción esperada. La definición de la tercera regla de decisión requiere que el valor de la primera condición se conmute de nuevo al utilizado en la primera regla de decisión, y que el valor de la segunda condición se conmute. De nuevo, la regla de decisión se introduce en la tabla de decisión y se define la acción esperada. El procedimiento se repite hasta que se definen las reglas de decisión en las que cada condición ha tomado el valor "verdadero" y "falso". Esto da como resultado una tabla de decisión con el número de reglas de decisión igual al número de condiciones + 1. En común con el enfoque de "todas las combinaciones de condiciones" descrito anteriormente, las reglas que no son factibles o redundantes se eliminan de la tabla.

Este enfoque sistemático suele dar lugar a un número menor de reglas de decisión que deben probarse en comparación con el enfoque de "todas las combinaciones de condiciones". Cada condición se prueba por lo menos una vez para el valor "verdadero" y por lo menos una vez para el valor "falso". Dado que cada regla de decisión se centra en una condición específica, se simplifica la localización de los defectos revelados. Con este enfoque, los defectos que se producen sólo cuando existen determinadas combinaciones de condiciones pueden pasar desapercibidos, ya que sólo se consideran las combinaciones simples.

Aplicabilidad

Ambos enfoques de esta técnica se aplican comúnmente para los niveles de integración, sistema y prueba de aceptación. Dependiendo del código, también puede ser aplicable para la prueba de componentes cuando un componente es responsable de un conjunto de lógica de decisión. Esta técnica es particularmente útil cuando los requisitos se presentan en forma de diagramas de flujo o tablas de reglas de negocio.

Las tablas de decisión son también una técnica de definición de requisitos y algunas especificaciones de requisitos pueden estar ya definidas en este formato. Incluso cuando los requisitos no se presentan en forma de tabla o de diagrama de flujo, las condiciones y las posibles combinaciones de condiciones se encuentran a menudo en la parte descriptiva.

Al diseñar las tablas de decisión, es importante tener en cuenta las reglas de decisión definidas, así como las que no están expresamente definidas, pero existirán. En general, un Analista de Prueba debe ser capaz de deducir las acciones esperadas para todas las reglas de decisión de acuerdo con la especificación o el oráculo de prueba.

Limitaciones/Dificultades

Al considerar las combinaciones de condiciones, encontrar todas las condiciones que interactúan puede ser difícil, en particular cuando los requisitos no están bien definidos o no existen. Hay que tener cuidado al seleccionar el número de condiciones consideradas en una tabla de decisión, de modo que el número de combinaciones de esas condiciones siga siendo controlable. Cuando el número de condiciones da lugar a un número inmanejable de combinaciones, un Analista de Prueba puede considerar la posibilidad de definir una jerarquía de tablas de decisión, en la que las condiciones de una tabla de decisión determinada puedan resultar de la invocación de otra tabla de decisión.

Cobertura

Habitualmente se consideran tres niveles de cobertura:

- Cobertura de todos los resultados de la condición: se aplican suficientes reglas de decisión para obtener los resultados verdaderos/falsos de cada condición. Esto da una cobertura máxima basada en el número de columnas de la tabla de condiciones iniciales (es decir, 2 a la potencia del número de condiciones). Esta es la cobertura más eficaz en función de los costes, pero no necesariamente se llevan a cabo todas las acciones esperadas posibles.
- Cubrir todas las acciones: se aplican suficientes reglas de decisión para generar todas las acciones esperadas posibles.
- Cubrir todas las reglas de decisión no redundantes y factibles: se cubren todas las reglas de decisión de una tabla de decisión totalmente colapsada. Este es un nivel de cobertura más alto, pero puede generar un gran número de casos de prueba.

Cuando se determinan las pruebas a partir de una tabla de decisión, también es importante tener en cuenta las condiciones frontera que deben ser probadas. Estas condiciones frontera pueden dar lugar a un mayor número de casos de prueba necesarios para probar adecuadamente el software. El análisis de los valores frontera y la partición de equivalencia son complementarias respecto de la técnica de la tabla de decisión.

Tipos de Defectos

Entre los defectos frecuentes figuran el procesamiento incorrecto basado en combinaciones particulares de condiciones que dan lugar a resultados inesperados. Durante la creación de las tablas de decisión, los defectos pueden encontrarse en el documento de especificación. No es inusual preparar un conjunto de condiciones y determinar que el resultado esperado no está especificado para una o más reglas de decisión. Los tipos de defectos más comunes son las omisiones (no hay información sobre lo que debería ocurrir realmente en una determinada situación) y las contradicciones. Las pruebas también pueden encontrar problemas con combinaciones de condiciones que no se tratan o que no se tratan bien.

3.2.4 Prueba de Transición de Estado

La prueba de transición de estado se utiliza para comprobar la capacidad del objeto de prueba para entrar y salir de los estados definidos mediante transiciones válidas, así como para entrar a estados inválidos o practicar transiciones inválidas. Los eventos hacen que el objeto de prueba pase de un estado a otro y realice acciones. Los eventos pueden estar cualificados por condiciones (a veces llamadas condiciones de guardia o guardias de transición) que influyen en el camino de la transición a tomar. Por ejemplo, un evento de inicio de sesión con una combinación válida de nombre de usuario/contraseña dará lugar a una transición diferente que un evento de inicio de sesión con una contraseña inválida. Esta información se representa en un diagrama de transición de estado o en una

tabla de transición de estado (que también puede incluir posibles transiciones inválidas entre estados) [Black09].

Aplicabilidad

La prueba de transición de estado es aplicable a cualquier software que tenga estados definidos y que tenga eventos que causen las transiciones entre esos estados (por ejemplo, cambio de pantalla). La prueba de transición de estado puede utilizarse en cualquier nivel de prueba. El software embebido, el software web y cualquier tipo de software transaccional son buenos candidatos para este tipo de prueba. Los sistemas de control, por ejemplo, los controladores de los semáforos, también son buenos candidatos para este tipo de prueba.

Limitaciones/Dificultades

La identificación de los estados suele ser la parte más difícil de la definición del diagrama de transición de estado o de la tabla de transición de estado. Cuando el objeto de prueba tiene una interfaz de usuario, las diversas pantallas que se muestran para el usuario se utilizan a menudo para definir los estados. En el caso de software embebido, los estados pueden depender de los estados del hardware.

Además de los propios estados, la unidad básica de las pruebas de transición de estado es la transición individual, también conocida como conmutación-0. Simplemente probando todas las transiciones individuales se encontrarán algunos tipos de defectos en las transiciones de estado, pero se pueden encontrar más probando secuencias de transiciones. Una secuencia de dos transiciones sucesivas se denomina conmutación-1; una secuencia de tres transiciones sucesivas es una conmutación-2, y así sucesivamente. (Estas conmutaciones se designan a veces alternativamente como conmutaciones N-1, donde N representa el número de transiciones que se atravesarán. Una sola transición, por ejemplo (una conmutación 0), sería una conmutación 1-1. (Bath14)).

Cobertura

Al igual que con otros tipos de técnicas de prueba, existe una jerarquía de niveles de cobertura de la prueba. El grado mínimo aceptable de cobertura es haber visitado todos los estados y atravesado cada transición por lo menos una vez. Una cobertura de transición del 100% (también conocida como cobertura del 100% de las conmutaciones-0 o cobertura del 100% de las ramas lógicas) garantizará que se visite cada estado y se atraviese cada transición, a menos que el diseño del sistema o el modelo de transición del estado (diagrama o tabla) sean defectuosos. Dependiendo de las relaciones entre los estados y las transiciones, puede ser necesario atravesar algunas transiciones más de una vez para ejecutar otras transiciones una sola vez.

El término "cobertura de conmutación-n" se refiere al número de conmutadores de longitud N+1 cubiertos, como porcentaje del número total de conmutaciones de esa longitud. Por ejemplo, para lograr el 100% de cobertura de conmutación-1 se requiere que cada secuencia válida de dos transiciones sucesivas haya sido probada por lo menos una vez. Esta prueba puede desencadenar algunos tipos de fallos que la cobertura del 100% de conmutación-0 omitiría.

La "cobertura de ida y vuelta" se aplica a situaciones en las que las secuencias de transiciones forman bucles. Se obtiene una cobertura de ida y vuelta del 100% cuando todos los bucles de cualquier estado que regresan al mismo estado han sido probados para todos los estados en los que los bucles comienzan y terminan. Este bucle no puede contener más de una ocurrencia de cualquier estado en particular (excepto el inicial/final) [Offutt16].

Para cualquiera de estos enfoques, un grado de cobertura aún mayor intentará incluir todas las transiciones inválidas identificadas en una tabla de estados. Los requisitos de cobertura y los conjuntos de cobertura para las pruebas de transición de estados deben identificar si se incluyen las transiciones inválidas.

El diseño de los casos de prueba para lograr la cobertura deseada se apoya en el diagrama de transición de estados o en la tabla de transición de estado para el objeto de prueba en particular. Esta información también puede representarse en una tabla que muestre las transiciones de las conmutaciones n para un valor particular de "n" [Black09].

Se puede aplicar un procedimiento manual para identificar los elementos que deben cubrirse (por ejemplo, transiciones, estados o conmutaciones-n). Un método sugerido consiste en imprimir el diagrama de transición de estado y la tabla de transición de estado y utilizar un bolígrafo o lápiz para

marcar los elementos cubiertos hasta que se muestre la cobertura requerida [Black09]. Este método llevaría demasiado tiempo en el caso de los diagramas de transición de estado y las tablas de transición de estado más complejos. Por consiguiente, debería utilizarse una herramienta que sirviera de apoyo a la prueba de transición de estado.

Tipos de Defectos

Entre los defectos habituales [Beizer95] se encuentran los siguientes:

- Tipos o valores de eventos incorrectos.
- Tipos o valores de acción incorrectos.
- Estado inicial incorrecto.
- Incapacidad de alcanzar algún estado de salida.
- Incapacidad de entrar en los estados requeridos.
- Estados adicionales (innecesarios).
- Capacidad de ejecutar transiciones inválidas.

Durante la creación del modelo de transición de estado, los defectos pueden encontrarse en el documento de especificación. Los tipos de defectos más comunes son las omisiones (no hay información sobre lo que realmente debería suceder en una determinada situación) y las contradicciones.

3.2.5 Técnica del Árbol de Clasificación

Los árboles de clasificación sirven de apoyo a ciertas técnicas de prueba de caja negra al permitir una representación gráfica del espacio de datos que se creará y que se aplica al objeto de prueba.

Los datos se organizan en clasificaciones y clases de la siguiente manera:

- Clasificaciones: Las clasificaciones representan parámetros dentro del espacio de datos para el objeto de prueba. Pueden ser parámetros de entrada, que pueden contener además estados y precondiciones del entorno, así como parámetros de salida. Por ejemplo, si una aplicación puede ser configurada de muchas maneras diferentes, las clasificaciones pueden incluir el cliente, el navegador, el idioma y el sistema operativo.
- Clases: Cada clasificación puede tener cualquier número de clases y subclases que describan la ocurrencia del parámetro. Cada clase, o partición de equivalencia, es un valor específico dentro de una clasificación. En el ejemplo anterior, la clasificación de idiomas podría incluir clases de equivalencia para el inglés, el francés y el español.

Los árboles de clasificación permiten al Analista de Prueba introducir combinaciones según se considere oportuno. Esto incluye, por ejemplo, combinaciones de a pares (véase la sección 3.2.6), combinaciones de a tríos y simples.

En los documentos [Bath14] y [Black09] se proporciona información adicional sobre la utilización de la técnica del árbol de clasificación.

Aplicabilidad

La creación de un árbol de clasificación ayuda al Analista de Prueba a identificar las particiones de equivalencia (clasificaciones) y los valores (clases) de una partición que sean de interés.

Un análisis más detallado del diagrama del árbol de clasificación permite identificar los posibles valores límite y determinar ciertas combinaciones de entradas que son de particular interés o que pueden descartarse (por ejemplo, porque son incompatibles). El árbol de clasificación resultante puede utilizarse entonces para apoyar la prueba de partición equivalente, el análisis de valores frontera o la prueba de a pares (véase la sección 3.2.6).

Limitaciones/Dificultades

A medida que aumenta la cantidad de clasificaciones y/o clases, el diagrama se hace más grande y menos fácil de utilizar.

Cobertura

Los casos de prueba pueden diseñarse para lograr, por ejemplo, una cobertura de clase mínima (es decir, todos los valores de una clasificación probada al menos una vez). El Analista de Prueba también puede decidir cubrir combinaciones de pares o incluso de tríos.

Tipos de Defectos

Los tipos de defectos encontrados dependen de la(s) técnica(s) a las que da soporte el árbol de clasificación (es decir, partición de equivalencias, análisis de valores frontera o prueba de a pares).

3.2.6 Prueba de a Pares

La prueba de a pares se utiliza cuando se prueba un software en el que varios parámetros de entrada, cada uno con varios valores posibles, deben probarse en combinación, dando lugar a más combinaciones de las que son factibles probar en el tiempo permitido. Los parámetros de entrada deben ser compatibles en el sentido de que cualquier opción para cualquier factor (es decir, cualquier valor seleccionado para cualquier parámetro de entrada) puede combinarse con cualquier opción para cualquier otro factor. La combinación de un parámetro específico (variable o factor) con un valor específico de ese parámetro se denomina un par parámetro-valor (por ejemplo, si "color" es un parámetro con (digamos) siete valores permitidos, entonces "color = rojo" sería un par parámetro-valor).

La prueba de a pares utiliza la aritmética combinatoria para asegurar que cada par par parámetro-valor se pruebe una vez contra cada par parámetro-valor de cada uno de los otros parámetros (es decir, se prueban "todos los pares" de pares parámetro-valor), a la vez que se evita probar todas las combinaciones de pares parámetro-valor. Si el Analista de Prueba utilizara un enfoque manual (no recomendado), se construiría una tabla con el caso de prueba representado por filas y una columna para cada parámetro. El Analista de Prueba rellenaría entonces la tabla con valores tales que todos los pares de valores puedan ser identificados en la tabla (véase [Black09]). Cualquier entrada en la tabla que esté en blanco puede ser rellenada con valores por el Analista de Prueba utilizando su propio conocimiento del dominio.

Hay varias herramientas disponibles para ayudar al Analista de Prueba en esta tarea (véase www.pairwise.org para ejemplos). Éstas requieren como entrada una lista de los parámetros y sus valores y calculan un conjunto mínimo de pruebas que cubre todos los pares de parámetros-valor. La salida de la herramienta puede utilizarse como entrada para los casos de prueba. Tenga en cuenta que el Analista de Prueba debe suministrar los resultados esperados para cada combinación que se cree con las herramientas.

Con frecuencia se utilizan árboles de clasificación (véase la sección 3.2.5) junto con la prueba de a pares [Bath14]. El diseño de los árboles de clasificación se apoya en herramientas y permite visualizar las combinaciones de parámetros y sus valores (algunas herramientas ofrecen una mejora para la prueba de a pares). Esto ayuda a identificar la siguiente información:

- Entradas para ser utilizadas por la técnica de prueba de a pares.
- Combinaciones de interés específicas (por ejemplo, de uso frecuente o una fuente común de defectos).
- Combinaciones particulares que son incompatibles. Esto no supone que los factores combinados no se afecten entre sí; muy bien podrían, pero deberían afectarse entre sí de manera aceptable.
- Relaciones lógicas entre variables. Por ejemplo, "si la variable 1 = x, entonces necesariamente la variable 2 no puede ser y". Los árboles de clasificación que captan estas relaciones se denominan "modelos de prestaciones".

Aplicabilidad

El problema de tener demasiadas combinaciones de valores de parámetros se manifiesta en al menos dos situaciones diferentes relacionadas con la prueba. Algunas situaciones de prueba implican varios parámetros, cada uno con varios valores posibles, por ejemplo, una pantalla con varios campos de entrada. En este caso, las combinaciones de valores de parámetros constituyen los datos de entrada para los casos de prueba. Además, algunos sistemas pueden ser configurables en varias dimensiones, lo que da lugar a un espacio de configuración potencialmente grande. En ambas situaciones, la prueba de a pares puede utilizarse para identificar un subconjunto de combinaciones, factibles en tamaño.

En el caso de los parámetros con muchos valores, puede aplicarse primero una partición de equivalencia o algún otro mecanismo de selección a cada parámetro, de forma individual, para reducir el número de valores de cada uno de ellos, antes de aplicar una prueba por pares para reducir el conjunto de combinaciones resultantes. La captura de los parámetros y sus valores en un árbol de clasificación apoya esta actividad.

Estas técnicas suelen aplicarse a los niveles de prueba de integración, sistema e integración de sistemas.

Limitaciones/Dificultades

La principal limitación de estas técnicas es la suposición de que los resultados de unas pocas pruebas son representativos de todas las pruebas y que esas pocas pruebas representan el uso previsto. Si hay una interacción inesperada entre ciertas variables, puede pasar desapercibida con este tipo de pruebas si no se prueba esa combinación particular. Estas técnicas pueden ser difíciles de explicar a un público no técnico, ya que puede no entender la reducción lógica de los ensayos. Toda explicación de este tipo debe equilibrarse mencionando los resultados de estudios empíricos [Kuhn16], que demostraron que en el ámbito de los dispositivos médicos objeto de estudio, el 66% de los fallos fueron provocados por una sola variable y el 97% por la interacción de una o dos variables. Existe un riesgo residual de que las pruebas de a pares no detecten los fallos de los sistemas en los que interactúan tres o más variables.

La identificación de los parámetros y sus respectivos valores es a veces difícil de lograr. Por consiguiente, esta tarea debe realizarse con el apoyo de árboles de clasificación cuando sea posible (véase la sección 3.2.5). Encontrar un conjunto mínimo de combinaciones para satisfacer un cierto nivel de cobertura es difícil de hacer manualmente. Las herramientas generalmente encuentran el conjunto de combinaciones más pequeño posible. Algunas herramientas apoyan la capacidad de obligar a que algunas combinaciones se incluyan o se excluyan de la selección final de combinaciones. Esta capacidad puede ser utilizada por un Analista de Prueba para destacar o restar importancia a los factores basados en el conocimiento del dominio o la información sobre el uso del producto.

Cobertura

La cobertura del 100% de los pares requiere que cada par de valores de cualquier par de parámetros se incluya en, al menos, una combinación.

Tipos de Defectos

El tipo más común de defectos detectados con este tipo de prueba son los defectos relacionados con los valores combinados de dos parámetros.

3.2.7 Prueba de Casos de Uso

Las pruebas de casos de uso proporcionan pruebas transaccionales basadas en escenarios que deberían emular el uso previsto del componente o sistema especificado por el caso de uso. Los casos de uso se definen en términos de las interacciones entre los actores y un componente o sistema que logra algún objetivo. Los actores pueden ser usuarios humanos, hardware externo u otros componentes o sistemas.

Aplicabilidad

Las pruebas de casos de uso suelen aplicarse a nivel de prueba de sistema y de aceptación. Puede utilizarse para pruebas de integración en las que el comportamiento de los componentes o sistemas se especifica mediante casos de uso e incluso para pruebas de componentes en las que el comportamiento de los mismos se especifica mediante casos de uso. Los casos de uso también son, a menudo, la base para la prueba de rendimiento, ya que representan un uso realista del sistema. Los escenarios descritos en los casos de uso pueden asignarse a usuarios virtuales para crear una carga realista en el sistema (siempre que los requisitos de carga y rendimiento se especifiquen en ellos o para ellos).

Limitaciones/Dificultades

Para que sean válidos, los casos de uso deben transmitir transacciones de usuario realistas. Las especificaciones de los casos de uso son una forma de diseño del sistema. Los requisitos de lo que

los usuarios deben lograr deben provenir de los usuarios o de sus representantes, y deben verificarse con los requisitos de la organización antes de diseñar los casos de uso correspondientes. El valor de un caso de uso se reduce si no refleja los requisitos reales del usuario y de la organización, o si dificulta, en lugar de ayudar, la realización de las tareas del usuario.

Una definición precisa de los comportamientos relativos al tratamiento de excepciones, alternativas y errores es importante para que la cobertura de la prueba sea completa. Los casos de uso deben tomarse como guía, pero no como una definición completa de lo que debe probarse, ya que pueden no proporcionar una definición clara de todo el conjunto de requisitos. También puede ser beneficioso crear otros modelos, como diagramas de flujo y/o tablas de decisión, a partir de la narración de los casos de uso para mejorar la precisión de la prueba y verificar el propio caso de uso. Al igual que con otras formas de especificación, es probable que esto revele anomalías lógicas en la especificación del caso de uso, si es que existen.

Cobertura

El mínimo aceptable de cobertura de un caso de uso es tener un caso de prueba para el comportamiento básico y suficientes casos de prueba adicionales para cubrir cada comportamiento alternativo y de tratamiento de errores. Si se requiere un conjunto mínimo de pruebas, se pueden incorporar múltiples comportamientos alternativos en un caso de prueba siempre que sean mutuamente compatibles. Si se requiere una mejor capacidad de diagnóstico (por ejemplo, para ayudar a aislar los defectos), puede diseñarse un caso de prueba adicional por cada comportamiento alternativo, aunque los comportamientos alternativos anidados seguirán requiriendo que algunos se amalgamen en casos de prueba únicos (por ejemplo, comportamientos alternativos de terminación versus no terminación dentro de un comportamiento de excepción de "reintento").

Tipos de Defectos

Los defectos incluyen el tratamiento incorrecto de comportamientos definidos, comportamientos alternativos omitidos, el procesamiento incorrecto de las condiciones presentadas e implementadas o una información de error incorrecta.

3.2.8 Combinación de Técnicas

A veces se combinan técnicas para crear casos de prueba. Por ejemplo, las condiciones identificadas en una tabla de decisión pueden someterse a una partición de equivalencia para descubrir las múltiples formas en que puede satisfacerse una condición. Los casos de prueba abarcarían entonces no sólo todas las combinaciones de condiciones, sino que también, para las condiciones que se partieran, se generarían casos de prueba adicionales para abarcar las particiones de equivalencia. Al seleccionar la técnica concreta que se aplicará, el Analista de Prueba deberá considerar la aplicabilidad de la técnica, las limitaciones, dificultades y los objetivos de la prueba en cuanto a la cobertura y los defectos que se han de detectar. Estos aspectos se describen para las distintas técnicas que se tratan en el presente capítulo. Es posible que no haya una sola técnica "mejor" para una situación. Las técnicas combinadas ofrecerán a menudo la cobertura más completa, suponiendo que se disponga de tiempo y competencia suficientes para aplicar correctamente las técnicas.

3.3 Técnicas de Prueba Basadas en la Experiencia

Las pruebas basadas en la experiencia utilizan las aptitudes y la intuición de los probadores, junto con su experiencia en aplicaciones o tecnologías similares para realizar pruebas con el fin de aumentar la detección de defectos. Estas técnicas de prueba van desde "pruebas rápidas" en las que el probador no tiene actividades formalmente planificadas para realizar, pasando por sesiones previamente planificadas hasta sesiones con un guion. Casi siempre son útiles, pero tienen un valor concreto cuando se pueden lograr los aspectos incluidos en la siguiente lista de ventajas.

La prueba basada en la experiencia tiene las siguientes ventajas:

- Es eficaz para detectar defectos.
- Puede ser una buena alternativa a los enfoques más estructurados en los casos en que la documentación del sistema es deficiente.
- Puede aplicarse cuando el tiempo de prueba está severamente restringido.
- Permite que se apliquen en la prueba los conocimientos especializados disponibles en el dominio y la tecnología. Esto puede incluir a quienes no participan en la prueba, por ejemplo, de analistas de negocio, clientes o clientes.
- Puede proporcionar una retroalimentación temprana a los desarrolladores.
- Ayuda al equipo a familiarizarse con el software a medida que se produce.
- Es eficaz cuando se analizan los fallos operativos.
- Permite aplicar una diversidad de técnicas de prueba.

La prueba basada en la experiencia tiene las siguientes desventajas:

- Puede ser inapropiado en sistemas que requieren una documentación de prueba detallada.
- Es difícil lograr altos niveles de repetibilidad.
- La capacidad de evaluar con precisión la cobertura de las pruebas es limitada.
- Las pruebas son menos adecuadas para una automatización posterior.

Cuando se utilizan enfoques reactivos y heurísticos, los probadores normalmente utilizan pruebas basadas en la experiencia, que son más reactivas a los acontecimientos que los enfoques de pruebas planificadas previamente. Además, la ejecución y la evaluación son tareas concurrentes. Algunos enfoques estructurados de las pruebas basadas en la experiencia no son totalmente dinámicos, es decir, las pruebas no se crean totalmente al mismo tiempo que el probador ejecuta la prueba. Este podría ser el caso, por ejemplo, cuando se utiliza la predicción de errores para centrarse en aspectos concretos del objeto de la prueba antes de su ejecución.

Obsérvese que, aunque se presentan algunas ideas sobre la cobertura de las técnicas que se examinan aquí, las técnicas de prueba basadas en la experiencia no tienen criterios formales de cobertura.

3.3.1 Predicción de Errores (o Conjetura de Errores)

Al utilizar la técnica de conjetura de errores, un Analista de Prueba utiliza la experiencia para suponer los posibles errores que podrían haberse cometido cuando se diseñó y desarrolló el código. Una vez identificados los errores esperados, un Analista de Prueba determina entonces los mejores métodos a utilizar para descubrir los defectos resultantes. Por ejemplo, si un Analista de Prueba espera que el software manifieste fallos cuando se introduce una contraseña inválida, se ejecutarán pruebas para introducir una variedad de valores diferentes en el campo de la contraseña para verificar si el error se cometió efectivamente y ha dado lugar a un defecto que puede considerarse como un fallo cuando se ejecuten las pruebas.

Además de utilizarse como técnica de prueba, la conjetura de errores también es útil durante el análisis de riesgos para identificar posibles modos de fallo. [Myers11]

Aplicabilidad

La conjetura de errores se realiza principalmente durante las pruebas de integración y de sistema, pero puede utilizarse en cualquier nivel de prueba. Esta técnica se utiliza a menudo con otras técnicas y contribuye a ampliar el alcance de los casos de prueba existentes. La conjetura de errores también puede utilizarse eficazmente cuando se prueba una nueva versión del software para comprobar los defectos comunes antes de iniciar pruebas más rigurosas y con guiones.

Limitaciones/Dificultades

Las siguientes limitaciones y dificultades corresponden a la conjetura de errores:

- La cobertura es difícil de evaluar y varía ampliamente con la capacidad y la experiencia del Analista de Prueba.
- Es mejor que la utilice un probador experimentado que esté familiarizado con los tipos de defectos que se introducen comúnmente en el tipo de código que se está probando.
- Se utiliza de forma habitual, pero con frecuencia no está documentado y, por tanto, puede ser menos reproducible que otras formas de prueba.
- Los casos de prueba pueden ser documentados, pero de una manera que sólo el autor entiende y puede reproducir.

Cobertura

Cuando se utiliza una taxonomía, la cobertura se determina tomando el número de elementos de la taxonomía probados dividido por el número total de elementos de la taxonomía y expresando la cobertura como un porcentaje. Sin una taxonomía, la cobertura está limitada por la experiencia y el conocimiento del probador y el tiempo disponible. La cantidad de defectos encontrados con esta técnica variará en función de la eficacia con que el probador pueda detectar las zonas problemáticas.

Tipos de Defectos

Los defectos típicos suelen ser los definidos en la taxonomía particular o "supuestos" por el Analista de Prueba, que podrían no haber sido encontrados en la prueba de la caja negra.

3.3.2 Prueba Basada en Lista de Comprobación

Al aplicar la técnica de prueba basada en listas de comprobación, un Analista de Prueba experimentado utiliza una lista generalizada de elementos de alto nivel que se deben observar, comprobar o recordar, o un conjunto de reglas o criterios con los que se debe comprobar un objeto de prueba. Estas listas de comprobación se elaboran sobre la base de un conjunto de normas, experiencia y otras consideraciones. Una lista de comprobación estándar de la interfaz de usuario puede emplearse como base para probar una aplicación y es un ejemplo de una prueba basada en una lista de comprobación. En los proyectos ágiles, las listas de comprobación pueden construirse a partir de los criterios de aceptación de una historia de usuario.

Aplicabilidad

La prueba basada en listas de comprobación es más eficaz en proyectos con un equipo de prueba experimentado que esté familiarizado con el software que se está probando o que esté familiarizado con el área que abarca la lista de comprobación (por ejemplo, para aplicar con éxito una lista de comprobación de la interfaz de usuario, el Analista de Prueba puede estar familiarizado con la prueba de la interfaz de usuario pero no con el sistema específico que se está probando). Dado que las listas de comprobación son de alto nivel y tienden a carecer de los pasos detallados que se encuentran comúnmente en los casos y procedimientos de prueba, los conocimientos del probador se utilizan para cubrir las carencias. Al eliminar los pasos detallados, las listas de comprobación son de bajo mantenimiento y pueden aplicarse a múltiples entregas similares.

Las listas de comprobación son muy adecuadas para proyectos en los que el software se entrega y cambia con rapidez. Esto ayuda a reducir tanto el tiempo de preparación como el de mantenimiento de la documentación de la prueba. Se pueden utilizar para cualquier nivel de prueba y también se utilizan para las pruebas de regresión y las pruebas de humo.

Limitaciones/Dificultades

La naturaleza de alto nivel de las listas de comprobación puede afectar a la reproducibilidad de los resultados de la prueba. Es posible que varios probadores interpreten las listas de comprobación de manera diferente y sigan distintos enfoques para cumplir con los elementos de la lista de comprobación. Esto puede causar resultados diferentes, aunque se utilice la misma lista de comprobación. Esto puede dar lugar a una cobertura más amplia, pero a veces se sacrifica la reproducibilidad. Las listas de comprobación también pueden dar lugar a un exceso de confianza en cuanto al nivel de cobertura que se alcanza, ya que las pruebas reales dependen del criterio del probador. Las listas de comprobación pueden obtenerse a partir de casos de prueba o listas más detalladas y tienden a crecer con el tiempo.

Se requiere un mantenimiento para garantizar que las listas de comprobación cubran los aspectos importantes del software que se está probando.

Cobertura

La cobertura puede determinarse dividiendo el número de elementos de la lista de comprobación probados por el número total de elementos de la lista de comprobación y expresando la cobertura como un porcentaje. La cobertura es tan buena como la lista de comprobación pero, debido a la naturaleza de alto nivel de la lista de comprobación, los resultados variarán en función del Analista de Prueba que la ejecute.

Tipos de Defectos

Los defectos habituales que se encuentran con esta técnica provocan fallos derivados de la variación de los datos, la secuencia de los pasos o el flujo de trabajo general durante la prueba.

3.3.3 Prueba Exploratoria

La prueba exploratoria se caracteriza por el hecho de que el probador aprende simultáneamente sobre el objeto de prueba y sus defectos, planifica el trabajo de prueba que se va a realizar, diseña y ejecuta las pruebas, e informa de los resultados. El probador ajusta dinámicamente los objetivos de la prueba durante la ejecución y prepara sólo documentación ligera. [Whittaker09]

Aplicabilidad

Una buena prueba exploratoria está planificada, es interactiva y creativa. Requiere poca documentación sobre el sistema que se va a probar y suele utilizarse en situaciones en que la documentación no está disponible o no es adecuada para otras técnicas de prueba. Con frecuencia, la prueba exploratoria se utiliza para complementar otras técnicas de prueba y servir de base para el desarrollo de casos de prueba adicionales. La prueba exploratoria se utiliza con frecuencia en los proyectos Agile para que las pruebas de historias de usuarios se realicen de forma flexible y rápida con una documentación mínima. Sin embargo, la técnica también puede aplicarse a proyectos que utilizan un CVDS secuencial.

Limitaciones/Dificultades

La cobertura de la prueba exploratoria puede ser esporádica y la reproducibilidad de las pruebas realizadas puede ser difícil. El uso de contratos de prueba para señalar las áreas que deben cubrirse en una sesión de prueba y el uso de un cronómetro para determinar el tiempo permitido para la prueba son métodos utilizados para gestionar la prueba exploratoria. Al final de una sesión o conjunto de sesiones de prueba, el Jefe de Prueba puede celebrar una sesión informativa para recopilar los resultados de la prueba y determinar los contratos de prueba para sesiones posteriores.

Otra dificultad de las sesiones exploratorias es hacer un seguimiento preciso de las mismas en un sistema de gestión de pruebas. Esto se hace a veces creando casos de prueba que son en realidad sesiones exploratorias. Esto permite que el tiempo asignado a la prueba exploratoria y la cobertura planificada sea monitorizada con los otros esfuerzos de prueba.

Dado que la reproducibilidad puede ser difícil con la prueba exploratoria, esto también puede causar problemas cuando se necesita recordar los pasos para reproducir un fallo. Algunas organizaciones utilizan la capacidad de captura/reproducción de las herramientas de automatización de pruebas para registrar los pasos dados por un probador exploratorio. Esto proporciona un registro completo de todas las actividades realizadas durante la sesión exploratoria (o cualquier sesión de prueba basada en la experiencia). El análisis de los detalles para encontrar la causa real de un fallo puede ser tedioso, pero al menos hay un registro de todos los pasos que se han dado.

Pueden utilizarse otras herramientas para capturar sesiones de pruebas exploratorias, pero éstas no registran los resultados esperados porque no capturan la interacción de la interfaz gráfica de usuario. En este caso, los resultados esperados deben consignarse para poder realizar un análisis adecuado de los defectos, si fuera necesario. En general, se recomienda que también se tomen notas mientras se realizan las pruebas exploratorias para apoyar la reproducibilidad cuando fuera necesario.

Cobertura

Los contratos pueden ser diseñados para tareas, objetivos y resultados específicos. Las sesiones exploratorias se planifican entonces para lograr esos criterios. En el contrato también se puede determinar dónde centrar el esfuerzo de la prueba, lo que está dentro y fuera del alcance de la sesión de prueba, y qué recursos se deben comprometer para completar las pruebas previstas. Una sesión puede utilizarse para centrarse en determinados tipos de defectos y otras zonas potencialmente problemáticas que pueden abordarse sin la formalidad de las pruebas con guion.

Tipos de Defectos

Los defectos habituales que se encuentran en las pruebas exploratorias son los problemas basados en escenarios que fueron omitidos durante las pruebas de adecuación funcional programadas, los problemas que se encuentran entre los límites funcionales y los problemas relacionados con el flujo de trabajo. Durante la prueba exploratoria también se descubren, a veces, problemas de rendimiento y seguridad.

3.3.4 Técnicas de Prueba Basada en Defectos

Una técnica de prueba basada en los defectos es aquella en la que el tipo de defecto buscado se utiliza como base para el diseño de la prueba, con pruebas obtenidas sistemáticamente a partir de lo que se conoce sobre el tipo de defecto. A diferencia de las pruebas de caja negra, que obtienen las pruebas a partir de la base de la prueba, las pruebas basadas en los defectos obtienen las pruebas a partir de listas que se centran en los defectos. En general, las listas pueden organizarse en tipos de defectos, causas fundamentales, síntomas de fallo y otros datos relacionados con los defectos. Las listas estándar se aplican a múltiples tipos de software y no son específicas de un producto. El uso de estas listas ayuda a aprovechar los conocimientos estándar de la industria para obtener pruebas concretas. Al adherirse a las listas específicas de la industria, se puede hacer un seguimiento de la métrica relativa a la ocurrencia de defectos en todos los proyectos e incluso en todas las organizaciones. Las listas de defectos más comunes son las que son específicas de una organización o proyecto y utilizan conocimientos y experiencia específicos.

En la prueba basada en defectos también se pueden utilizar listas de riesgos identificados y escenarios de riesgo como base para la realización de pruebas selectivas. Esta técnica de prueba permite al Analista de Prueba centrarse en un tipo específico de defecto o trabajar sistemáticamente con una lista de defectos conocidos y comunes de un tipo determinado. A partir de esta información, el Analista de Prueba crea los casos de prueba y las condiciones de prueba que harán que el defecto se manifieste, si existe.

Aplicabilidad

La prueba basada en el defecto puede aplicarse en cualquier nivel de prueba, pero normalmente se utiliza durante la prueba de sistema.

Limitaciones/Dificultades

Existen numerosas taxonomías de defectos y pueden centrarse en tipos particulares de pruebas, como la usabilidad. Es importante elegir una taxonomía que sea aplicable al software que se está probando, si es que hay alguna disponible. Por ejemplo, puede ser que no haya ninguna taxonomía disponible para el software innovador. Algunas organizaciones han compilado sus propias taxonomías de defectos probables o frecuentes. Cualquiera que sea la taxonomía utilizada, es importante definir la cobertura prevista antes de iniciar la prueba.

Cobertura

La técnica proporciona criterios de cobertura que se utilizan para determinar cuándo se han identificado todos los casos de prueba útiles. Los elementos de cobertura pueden ser elementos estructurales, elementos de especificación, escenarios de uso o cualquier combinación de éstos, dependiendo de la lista de defectos. En la práctica, los criterios de cobertura de las técnicas de ensayo basadas en los defectos tienden a ser menos sistemáticos que los de las técnicas de prueba de caja negra, en el sentido de que sólo se dan reglas generales para la cobertura y la decisión específica sobre lo que constituye el límite de la cobertura útil es discrecional. Al igual que en el caso de otras técnicas, los criterios de cobertura no significan que todo el conjunto de pruebas esté completo, sino más bien que los defectos que se están considerando ya no sugieren ninguna prueba útil basada en esa técnica.

Tipos de defectos

Los tipos de defectos descubiertos suelen depender de la taxonomía en uso. Por ejemplo, si se utiliza una lista de defectos de la interfaz de usuario, es probable que la mayoría de los defectos descubiertos estén relacionados con la interfaz de usuario, pero pueden descubrirse otros defectos como subproducto de las pruebas específicas.

3.4 Aplicación de la Técnica más Adecuada

Las técnicas de prueba de caja negra y las basadas en la experiencia son más efectivas cuando se utilizan de forma conjunta. Las técnicas basadas en la experiencia llenan los vacíos en la cobertura de la prueba que resultan de cualquier debilidad sistemática en las técnicas de prueba de caja negra.

No hay una técnica perfecta para todas las situaciones. Es importante que el Analista de Prueba comprenda las ventajas y desventajas de cada técnica y pueda seleccionar la mejor técnica o conjunto de técnicas para la situación, teniendo en cuenta el tipo de proyecto, el calendario, el acceso a la información, las aptitudes del probador y otros factores que pueden influir en la selección.

En el análisis de cada técnica de prueba de caja negra y basada en la experiencia (véanse las secciones 3.2 y 3.3, respectivamente), la información proporcionada en "aplicabilidad", "limitaciones/dificultades" y "cobertura" orienta al Analista de Prueba en la selección de las técnicas de prueba más apropiadas para aplicar.

4. Prueba de las Características de Calidad del Software - 180 minutos

Palabras Clave

Español	Inglés
accesibilidad	accessibility
compatibilidad	compatibility
adecuación funcional	functional appropriateness
completitud funcional	functional completeness
corrección funcional	functional correctness
adecuación funcional	functional suitability
interoperabilidad	interoperability
capacidad de ser aprendido	learnability
operabilidad	operability
Inventario de Medición de Usabilidad Software (IMUS)	Software Usability Measurement Inventory (SUMI)
usabilidad	usability
protección contra errores del usuario	user error protection
experiencia de usuario	user experience
estética de interfaz de usuario	user interface aesthetics
Inventario de Análisis y Medición de Sitios Web (IAMSW)	Website Analysis and MeasureMent Inventory (WAMMI)

Objetivos de Aprendizaje para "Prueba de las Características de Calidad del Software"

4.1 Introducción		
		No hay objetivos de aprendizaje
4.2 Características de Calidad para la Prueba en el Dominio de Negocio		
AP-4.2.2	**(K2)**	Definir los defectos típicos a los que se debe dirigir para las características de completitud, corrección y pertinencia funcional.
AP-4.2.3	**(K2)**	Definir cuándo se deben probar las características de completitud, corrección y pertinencia funcional en el ciclo de vida de desarrollo de software.
AP-4.2.4	**(K2)**	Explicar los enfoques que serían adecuados para verificar y validar tanto la implementación de los requisitos de usabilidad como el cumplimiento de las expectativas del usuario.
AP-4.2.5	**(K2)**	Explicar el papel del Analista de Prueba en la prueba de interoperabilidad, incluyendo la identificación de los defectos a los que se debe dirigir.
AP-4.2.6	**(K2)**	Explicar el rol del Analista de Prueba en la prueba de portabilidad, incluyendo la identificación de los defectos a los que se debe dirigir.
AP-4.2.7	**(K4)**	Determinar las condiciones de prueba necesarias para verificar las características de calidad funcionales y/o no funcionales dentro del ámbito del Analista de Prueba para un determinado conjunto de requisitos.

4.0 Introducción

Si bien en el capítulo anterior se describieron técnicas específicas disponibles para el probador, en el presente capítulo se considera la aplicación de esas técnicas para evaluar las características utilizadas para describir la calidad de las aplicaciones o sistemas de software.

Este programa de estudio expone las características de calidad que pueden ser evaluadas por un Analista de Prueba. Los atributos que debe evaluar el Analista de Prueba Técnicas se consideran en el programa de estudios de Probador Certificado del ISTQB®, Nivel Avanzado, Analista de Prueba Técnicas [CTAL-TTA].

La descripción de las características de calidad de producto contenida en la norma ISO 25010 [ISO25010] se utiliza como guía para describir las características. El modelo de calidad de software de la ISO divide la calidad del producto en diferentes características de calidad del producto, cada una de las cuales puede tener subcaracterísticas. Éstas se muestran en la siguiente tabla, junto con una indicación de las características/subcaracterísticas que abarcan los programas de estudio del Analista de Prueba y del Analista de Prueba técnicas:

Característica	Subcaracterísticas	Analista de Prueba	Analista de Prueba Técnicas
adecuación funcional	• completitud funcional • corrección funcional • pertinencia funcional	**X**	
fiabilidad	• madurez • disponibilidad • tolerancia a defectos • recuperabilidad		**X**
usabilidad	• capacidad de reconocimiento de la pertinencia • capacidad de ser aprendido • capacidad de ser operado • protección ante error de usuario • estética de interfaz de usuario • accesibilidad	**X**	
eficiencia de desempeño	• comportamiento temporal • utilización de recursos • capacidad		**X**
mantenibilidad	• modularidad • capacidad de reutilización • analizabilidad • capacidad de ser modificado • capacidad de ser probado		**X**
portabilidad	• adaptabilidad • instalanbilidad • reemplazabilidad	**X**	**X**
seguridad	• confidencialidad • integridad • no repudiación • responsabilidad • autenticidad		**X**

Característica	Subcaracterísticas	Analista de Prueba	Analista de Prueba Técnicas
compatibilidad	• coexistencia		**X**
	• interoperabilidad	**X**	

Las áreas de responsabilidad del Analista de Prueba y del Analista de Prueba Técnicas se muestran en el cuadro anterior. Si bien esta asignación de trabajo puede variar en las diferentes organizaciones, es la que se sigue en el programa de estudio asociado de ISTQB®.

Se deben reconocer los riesgos típicos de todas las características y subcaracterísticas de calidad que se tratan en esta sección, de manera que se pueda formular y documentar una estrategia de prueba adecuada. La prueba de las características de calidad requiere una atención especial a la coordinación temporal del CVDS, a las herramientas necesarias, a la disponibilidad de software, a la documentación y a los conocimientos técnicos. Sin una estrategia para tratar cada característica y sus necesidades de prueba únicas, el probador puede no tener una planificación adecuada, una rampa ascendente y un tiempo de ejecución de la prueba incorporados en el calendario [Bath14]. Algunas de estas pruebas, por ejemplo, las pruebas de usabilidad, pueden requerir la asignación de recursos humanos especiales, una planificación extensiva, laboratorios dedicados, herramientas específicas, competencias de prueba especializadas y, en la mayoría de los casos, una cantidad de tiempo significativa. En algunos casos, la prueba de usabilidad puede ser realizada por un grupo de expertos en usabilidad o en experiencia de usuario independiente.

Aunque el Analista de Prueba puede no ser responsable de las características de calidad que requieren un enfoque más técnico, es importante que el Analista de Prueba sea consciente de las demás características y comprenda las áreas de superposición de las pruebas. Por ejemplo, un objeto de prueba que falla en la prueba de rendimiento probablemente también fallará en la prueba de usabilidad si es demasiado lento para que el usuario lo utilice eficazmente. Del mismo modo, un objeto de prueba con problemas de interoperabilidad con algunos componentes probablemente no esté listo para las pruebas de portabilidad, ya que esto tenderá a ocultar los problemas más básicos cuando se modifique el entorno.

4.1 Características de Calidad para la Prueba en el Dominio de Negocio

La prueba de adecuación funcional es un foco primario para el Analista de Prueba. La prueba de adecuación funcional se centra en "qué" hace el objeto de prueba. La > para la prueba de adecuación funcional es generalmente alguna forma de base de prueba como los requisitos, una especificación, un dominio específico de conocimientos especializados o una necesidad implícita. La prueba funcional varía según el nivel de prueba en el que se realiza y también puede estar condicionada por el CVDS. Por ejemplo, una prueba funcional realizada durante la prueba de integración pondrá a prueba la funcionalidad de los módulos de interfaz que implementan una única función definida. En el nivel de prueba del sistema, la prueba funcional incluye la comprobación de la funcionalidad del sistema en su conjunto. En el caso de los sistemas de sistemas, la prueba de adecuación funcional se centrará principalmente en la prueba extremo a extremo en todos los sistemas integrados. Durante las pruebas de adecuación funcional se emplea una amplia variedad de técnicas de prueba (véase el capítulo 3).

En un entorno ágil, la prueba de adecuación funcional suele incluir lo siguiente:

- Prueba de la funcionalidad específica (por ejemplo, historias de usuario) planificada para la implementación en la iteración particular.
- Prueba de regresión para toda la funcionalidad que no ha sido objeto de cambio.

Además de la prueba de adecuación funcional que se trata en esta sección, hay también ciertas características de calidad que forman parte del área de responsabilidad del Analista de Prueba que se consideran no funcionales (centradas en "cómo" el objeto de prueba proporciona la funcionalidad) áreas de prueba.

4.1.1 Prueba de Corrección Funcional

La corrección funcional implica la verificación de la adhesión de la aplicación a los requisitos especificados o implícitos y también puede incluir la exactitud del cómputo. La prueba de corrección emplea muchas de las técnicas de prueba explicadas en el capítulo 3 y a menudo utiliza la especificación o un sistema legado como oráculo de prueba. La prueba de corrección se puede realizar en cualquier nivel de prueba y está dirigida al tratamiento incorrecto de datos o situaciones.

4.1.2 Prueba de Pertinencia Funcional

La prueba de pertinencia funcional consiste en evaluar y validar la idoneidad de un conjunto de funciones para las tareas específicas previstas. Estas pruebas pueden basarse en el diseño funcional (por ejemplo, casos de uso y/o historias de usuario). La prueba de pertinencia funcional suele realizarse durante la prueba de sistema, pero también puede llevarse a cabo durante las últimas etapas de la prueba de integración. Los defectos descubiertos en estas pruebas son indicaciones de que el sistema no podrá satisfacer las necesidades del usuario de una manera que se considere aceptable.

4.1.3 Prueba de Completitud Funcional

Se realizan pruebas de completitud funcional para determinar la cobertura de las tareas especificadas y los objetivos de usuario por la funcionalidad implementada. La trazabilidad entre los elementos de especificación (por ejemplo, requisitos, historias de usuario, casos de uso) y la funcionalidad implementada (por ejemplo, función, unidad, flujo de trabajo) es esencial para poder determinar la completitud requerida. La medición de la completitud funcional puede variar en función del nivel de prueba concreto y/o del CVDS utilizado. Por ejemplo, la completitud funcional para una iteración ágil puede basarse en las historias de usuario y características implementadas. La completitud funcional para la prueba de integración del sistema puede centrarse en la cobertura de casos de negocio de alto nivel.

La determinación de la integridad funcional suele estar soportada por herramientas de gestión de pruebas si el Analista de Prueba mantiene la trazabilidad entre los casos de prueba y los elementos de la especificación funcional.

Los niveles de completitud funcional inferiores a los esperados son indicios de que el sistema no se ha implementado en su totalidad.

4.1.4 Prueba de Interoperabilidad

La prueba de interoperabilidad verifica el intercambio de información entre dos o más sistemas o componentes. La prueba se centra en la capacidad de intercambiar información y posteriormente utilizar la información que se ha intercambiado. La prueba debe abarcar todos los entornos de destino previstos (incluidas las variaciones en el hardware, el software, el middleware, el sistema operativo, etc.) para garantizar que el intercambio de datos funcione correctamente. En realidad, esto sólo puede ser factible para un número relativamente pequeño de entornos. En ese caso, la prueba de interoperabilidad puede limitarse a un grupo representativo seleccionado de entornos. La especificación de la prueba de interoperabilidad requiere que se identifiquen, configuren y pongan a disposición del equipo de prueba las combinaciones de los entornos de destino previstos. Esos entornos se prueban luego mediante una selección de casos de prueba funcionales que practican los diversos puntos de intercambio de datos presentes en el entorno.

La interoperabilidad se relaciona con la forma en que los diferentes componentes y sistemas de software interactúan entre sí. El software con buenas características de interoperabilidad puede integrarse con otros sistemas sin necesidad de grandes cambios. El número de cambios y el esfuerzo requerido para implementar y probar esos cambios puede utilizarse como medida de la interoperabilidad.

La prueba de la interoperabilidad del software puede, por ejemplo, centrarse en las siguientes prestaciones de diseño:

- Utilización de normas de comunicación en toda la industria, como XML.
- Capacidad de detectar automáticamente las necesidades de comunicación de los sistemas con los que interactúa y ajustarlas en consecuencia.

La prueba de interoperabilidad puede ser particularmente significativa para lo siguiente:

- Productos y herramientas de software comerciales de distribución masiva disponibles en el mercado.
- Aplicaciones basadas en un sistema de sistemas.
- Sistemas basados en la Internet de las Cosas.
- Servicios web con conectividad con otros sistemas.

Este tipo de prueba se realiza durante la integración de componentes y la prueba de sistema, centrándose en la interacción del sistema con su entorno. En el nivel de integración del sistema, este tipo de prueba se realiza para determinar el grado de interacción del sistema completamente desarrollado con otros sistemas. Dado que los sistemas pueden interoperar en múltiples niveles, el Analista de Prueba debe comprender estas interacciones y ser capaz de crear las condiciones que practicarán las diversas interacciones. Por ejemplo, si dos sistemas intercambian datos, el Analista de Prueba debe ser capaz de crear los datos necesarios y las transacciones requeridas para realizar el intercambio de datos. Es importante recordar que es posible que todas las interacciones no estén claramente especificadas en los documentos de requisitos. En cambio, muchas de esas interacciones se definirán únicamente en los documentos de arquitectura y diseño del sistema. El Analista de Prueba debe ser capaz y estar preparado para examinar esos documentos a fin de determinar los puntos de intercambio de información entre los sistemas y entre el sistema y su entorno, para asegurarse de que todos ellos sean probados. Técnicas como la partición de equivalencias, el análisis de valores frontera, las tablas de decisión, los diagramas de estado, los casos de uso y las pruebas de a pares son todas aplicables a las pruebas de interoperabilidad. Los defectos típicos encontrados incluyen el intercambio incorrecto de datos entre los componentes que interactúan.

4.1.5 Evaluación de la Usabilidad

Los analistas de prueba suelen estar en posición de coordinar y apoyar la evaluación de la usabilidad. Esto puede incluir la especificación de la prueba de usabilidad o actuar como moderador trabajando con los usuarios para llevar a cabo la prueba. Para hacerlo de forma eficaz, un Analista de Prueba debe comprender los principales aspectos, objetivos y enfoques que implican estos tipos de prueba. Por favor, consulte el programa de estudios de especialista en prueba de usabilidad de ISTQB® [ISTQB_FL_UT] para obtener más detalles además de la descripción proporcionada en esta sección.

Es importante comprender por qué los usuarios pueden tener dificultades para utilizar el sistema o no tienen una experiencia de usuario positiva (UX) (por ejemplo, con el uso de software para el entretenimiento). Para comprenderlo es necesario, en primer lugar, apreciar que el término "usuario" puede aplicarse a una amplia gama de diferentes tipos de personas, desde expertos en tecnología de la información hasta niños y personas con discapacidad.

4.1.5.1 Aspectos de la Usabilidad

En esta sección se abordan los siguientes tres aspectos:

- Usabilidad.
- Experiencia de usuario (UX).
- Accesibilidad.

Usabilidad

La prueba de usabilidad se centra en los defectos del software que afectan a la capacidad del usuario para realizar tareas a través de la interfaz de usuario. Esos defectos pueden afectar a la capacidad del usuario para alcanzar sus objetivos de manera eficaz, o eficiente, o con satisfacción. Los problemas de usabilidad pueden dar lugar a confusión, error, retraso o rotundo fracaso en la realización de alguna tarea por parte del usuario.

- Reconocimiento de la pertinencia (es decir, capacidad de ser comprendido) - atributos del software que afectan al esfuerzo requerido por parte del usuario para reconocer el concepto lógico y su aplicabilidad.
- Capacidad de ser aprendido - atributos del software que afectan el esfuerzo requerido por el usuario para aprender la aplicación.

- Operabilidad - atributos del software que afectan el esfuerzo requerido por el usuario para llevar a cabo las tareas de manera efectiva y eficiente.
- Estética de la interfaz de usuario (es decir, atractivo) - atributos visuales del software que son apreciados por el usuario.
- Protección ante error de usuario - grado en el que un sistema protege a los usuarios contra la comisión de errores.
- Accesibilidad (véase más abajo)

Experiencia de Usuario (UX)

La evaluación de la experiencia de usuario se refiere a toda la experiencia del usuario con el objeto de prueba, no sólo a la interacción directa. Esto es de particular importancia para los objetos de prueba, en los que factores como el disfrute y la satisfacción del usuario son fundamentales para el éxito del negocio.

Entre los factores típicos que influyen en la experiencia de usuario se encuentran los siguientes:

- Imagen de marca (es decir, la confianza de los usuarios en el fabricante).
- Comportamiento interactivo.
- La disponibilidad de asistencia por parte del objeto de prueba, incluyendo el sistema de ayuda, el apoyo y la formación.

Accesibilidad

Es importante considerar la accesibilidad al software para quienes tienen necesidades o restricciones particulares para su uso. Esto incluye a las personas con discapacidades. Las pruebas de accesibilidad deben tener en cuenta las normas pertinentes, como las Directrices de Accesibilidad al Contenido en la Web (WCAG), y la legislación, como las Leyes de Discriminación por Discapacidad (Irlanda del Norte, Australia), la Ley de Igualdad de 2010 (Inglaterra, Escocia, Gales) y la Sección 508 (EE.UU.). La accesibilidad, similar a la usabilidad, debe ser considerada cuando se realizan actividades de diseño. Las pruebas se realizan a menudo durante los niveles de integración y continúan a través de la prueba de sistema y en los niveles de prueba de aceptación. Los defectos suelen determinarse cuando el software no cumple con las normas o reglamentos establecidos para el software.

Las medidas típicas para mejorar la accesibilidad se centran en las oportunidades que se ofrecen a los usuarios con discapacidades para interactuar con la aplicación. Entre ellas figuran las siguientes:

- Reconocimiento de voz para las entradas.
- Asegurarse de que el contenido no textual que se presente al usuario tenga una alternativa textual equivalente.
- Permitir que el texto sea redimensionado sin pérdida de contenido o funcionalidad.

Las directrices de accesibilidad apoyan al Analista de Prueba al proporcionarle una fuente de información y listas de comprobación que pueden utilizarse para las pruebas (en [ISTQB_FL_UT] figuran ejemplos de directrices de accesibilidad). Además, hay disponibilidad de herramientas y complementos para los navegadores que ayudan a los probadores a identificar los problemas de accesibilidad, como la mala elección de colores en las páginas web que violan las pautas de daltonismo.

4.1.5.2 Enfoques para la Evaluación de la Usabilidad

La usabilidad, la experiencia de usuario y la accesibilidad pueden probarse mediante uno o más de los siguientes enfoques:

- Prueba de usabilidad.
- Revisiones de la usabilidad.
- Encuestas y cuestionarios de usabilidad.

Prueba de usabilidad

La prueba de usabilidad evalúa la facilidad con que los usuarios pueden utilizar o aprender a utilizar el sistema para alcanzar un objetivo específico en un contexto concreto. La prueba de usabilidad está dirigida a medir lo siguiente:

Eficacia - capacidad del objeto de prueba para permitir a los usuarios alcanzar objetivos específicos con exactitud y completitud en un contexto específico de uso.

Eficiencia - capacidad del objeto de prueba para permitir a los usuarios dedicar cantidades adecuadas de recursos en relación con la eficacia lograda en un contexto de utilización determinado.

Satisfacción - capacidad del objeto de prueba para satisfacer a los usuarios en un contexto de uso específico.

Es importante señalar que, a menudo, el diseño y la especificación de la prueba de usabilidad son realizados por el Analista de Prueba en cooperación con los probadores que tienen conocimientos especiales de pruebas de usabilidad, y los ingenieros de diseño de usabilidad que entienden el proceso de diseño centrado en el ser humano (véase [ISTQB_FL_UT] para más detalles).

Revisiones de la usabilidad

Las inspecciones y revisiones son un tipo de prueba realizadas desde la perspectiva de la usabilidad que ayudan a aumentar el nivel de participación del usuario. Esto puede ser rentable si se encuentran problemas de usabilidad en las especificaciones de requisitos y diseños al principio del CVDS. La evaluación heurística (inspección sistemática de un diseño de interfaz de usuario para la usabilidad) puede utilizarse para encontrar los problemas de usabilidad en el diseño, de forma que puedan ser atendidos como parte de un proceso de diseño iterativo. Esto implica que un pequeño grupo de evaluadores examine la interfaz y juzgue su conformidad con los principios de usabilidad reconocidos (la "heurística"). Las revisiones son más eficaces cuando la interfaz de usuario es más visible. Por ejemplo, las capturas de pantalla de muestra suelen ser más fáciles de entender e interpretar que la mera descripción de la funcionalidad que ofrece una pantalla determinada. La visualización es importante para una revisión adecuada de la usabilidad de la documentación.

Encuestas y cuestionarios de usabilidad

Se pueden utilizar técnicas de encuestas y cuestionarios para reunir observaciones y retroalimentación sobre el comportamiento de los usuarios con el sistema. Las encuestas estandarizadas y disponibles públicamente como el Inventario de Medición de Usabilidad Software (IMUS) (Software Usability Measurement Inventory, SUMI por sus siglas en inglés) y el Inventario de Análisis y Medición de Sitios Web (IAMSW) (Website Analysis and MeasureMent Inventory, WAMMI por sus siglas en inglés) permiten hacer comparaciones con una base de datos de mediciones de usabilidad anteriores. Además, dado que el IMUS proporciona medidas tangibles de usabilidad, esto puede proporcionar un conjunto de criterios de finalización/aceptación.

4.1.6 Prueba de Portabilidad

La prueba de portabilidad se refiere al grado en que un componente o sistema de software puede transferirse a su entorno previsto, ya sea inicialmente o desde un entorno existente.

La clasificación ISO 25010 de las características de calidad de producto incluye las siguientes subcaracterísticas de la portabilidad:

- Adaptabilidad.
- Instalanbilidad.
- Reemplazabilidad.

La tarea de identificar los riesgos y diseñar las pruebas para las características de portabilidad se comparte entre el Analista de Prueba y el Analista de Prueba Técnicas (véase [ISTQB_ALTTA_SYL] Sección 4.7).

4.1.6.1 Prueba de Instalabilidad

La prueba de instalabilidad se lleva a cabo en el software y los procedimientos escritos utilizados para instalar y desinstalar el software en su entorno de destino.

Los objetivos típicos de la prueba de instalabilidad centrados en el Analista de Prueba incluyen:

- Validar que las diferentes configuraciones del software pueden ser instaladas con éxito. En los casos en que se pueda configurar un gran número de parámetros, el Analista de Prueba puede diseñar la prueba utilizando la técnica de a pares de elementos para reducir el número de

combinaciones de parámetros probados y centrarse en las configuraciones particulares de interés (por ejemplo, las que se utilizan con frecuencia).
- Probar la corrección funcional de los procedimientos de instalación y desinstalación.
- Realizar pruebas funcionales después de una instalación o desinstalación para detectar cualquier defecto que pueda haberse introducido (por ejemplo, configuraciones incorrectas, funciones no disponibles).
- Identificar problemas de usabilidad en los procedimientos de instalación y desinstalación (por ejemplo, para validar que los usuarios reciben instrucciones comprensibles y mensajes de retroalimentación/error al ejecutar el procedimiento).

4.1.6.2 Prueba de Adaptabilidad

Las pruebas de adaptabilidad comprueban si una aplicación determinada puede funcionar correctamente en todos los entornos previstos (hardware, software, middleware, sistema operativo, etc.). El Analista de Prueba apoya la prueba de adaptabilidad diseñando pruebas que identifican las combinaciones de los entornos de destino previstos (por ejemplo, versiones de diferentes sistemas operativos móviles soportados, diferentes versiones de navegadores que pueden utilizarse). Estos entornos se prueban a continuación mediante una selección de casos de prueba funcionales que ejercitan los diversos componentes presentes en el entorno.

4.1.6.3 Prueba de Reemplazabilidad

La prueba de reemplazabilidad se centra en la capacidad de los componentes o versiones de software dentro de un sistema para ser intercambiados por otros. Esto puede ser particularmente pertinente para las arquitecturas de sistemas basados en la Internet de las Cosas, en las que el intercambio de diferentes dispositivos hardware y/o instalaciones de software es un hecho común. Por ejemplo, un dispositivo de hardware utilizado en un almacén para registrar y controlar los niveles de existencias puede ser reemplazado por un dispositivo hardware más avanzado (por ejemplo, con un escáner mejor) o el software instalado puede actualizarse con una nueva versión que permita que los pedidos de intercambio de existencias se emitan automáticamente al sistema de un proveedor.

La prueba de reemplazabilidad puede ser realizada por el Analista de Prueba en paralelo con la prueba de integración funcional, cuando se dispone de más de un componente alternativo para la integración en el sistema completo.

5. Revisiones - 120 minutos

Palabras Clave

Español	Inglés
revisión basada en lista de comprobación	checklist-based reviewing

Objetivos de Aprendizaje para "Revisiones"

5.1 Introducción		
		No hay objetivos de aprendizaje.
5.2 Utilización de Listas de Comprobación en las Revisiones		
AP-5.2.1	**(K3)**	Identificar los problemas en una especificación de requisitos de acuerdo con la información de la lista de comprobación proporcionada en el programa de estudio.
AP-5.2.2	**(K3)**	AP-5.2.2 (K3) Identificar los problemas en una historia de usuario de acuerdo con la información de la lista de comprobación proporcionada en el programa de estudio.

5.0 Introducción

Un proceso de revisión adecuado requiere planificación, participación y seguimiento. Los analistas de prueba deben participar activamente en el proceso de revisión, aportando sus puntos de vista únicos. Cuando se hacen correctamente, las revisiones pueden ser la mayor contribución individual, y la más rentable, a la calidad general de la entrega.

5.1 Utilización de Listas de Comprobación en las Revisiones

La revisión basada en listas de comprobación es la técnica más común utilizada por un Analista de Prueba al revisar la base de prueba. Las listas de comprobación se utilizan durante las revisiones para que los participantes comprueben puntos específicos durante la revisión. También pueden ayudar a despersonalizar la revisión (por ejemplo, "Esta es la misma lista de comprobación que utilizamos para cada revisión. No nos centramos sólo en el producto de **su** trabajo").

Las revisiones basadas en listas de comprobación pueden realizarse de forma genérica para todas las revisiones o pueden centrarse en características de calidad, áreas o tipos de documentos específicos. Por ejemplo, una lista de comprobación genérica podría verificar las propiedades generales del documento, como tener un identificador único, no tener referencias marcadas como "por determinar", un formato adecuado y elementos de conformidad similares. Una lista de comprobación específica para un documento de requisitos podría contener comprobaciones del uso adecuado de los términos "deberá" y "debería", comprobaciones de la capacidad de ser probado comprobación de cada requisito enunciado, y así sucesivamente.

El formato de los requisitos también puede indicar el tipo de lista de comprobación que se utilizará. Un documento de requisitos que tenga un formato de texto narrativo tendrá criterios de revisión diferentes a los que se basan en los diagramas.

Las listas de comprobación también pueden estar orientadas a un aspecto particular, por ejemplo:

- Un conjunto de competencias de programador/arquitecto o un conjunto de competencias de probador.
- En el caso del Analista de Prueba, la lista de verificación de las competencias del probador sería la más apropiada.
- Estas listas de comprobación podrían incluir elementos como los que se muestran en las secciones 5.2.1 y 5.2.2.
- Un determinado nivel de riesgo (por ejemplo, en sistemas de seguridad crítica) - las listas de comprobación incluirán normalmente la información específica necesaria para el nivel de riesgo.

- Una técnica de prueba específica - la lista de comprobación se centrará en la información necesaria para una técnica concreta (por ejemplo, las reglas que se representarán en una tabla de decisión).

5.1.1 Revisión de Requisitos

Los siguientes puntos son un ejemplo de lo que podría incluir una lista de comprobación orientada a los requisitos:

- Fuente del requisito (por ejemplo, persona, departamento).
- La capacidad de ser probado de cada uno de los requisitos.
- Criterios de aceptación de cada requisito.
- Disponibilidad de una estructura de llamada de casos de uso, si correspondiera.
- Identificación única de cada requisito, caso de uso o historia de usuario.
- Versión de cada requisito, caso de uso, historia de usuario.
- Trazabilidad para cada requisito con respecto a los requisitos de negocio/comercialización.
- Trazabilidad entre los requisitos y/o casos de uso (si correspondiera).
- Uso de una terminología coherente (por ejemplo, utiliza un glosario).

Es importante recordar que, si un requisito no presenta la capacidad de ser probado, es decir, si está definido de tal manera que el Analista de Prueba no puede determinar cómo probarlo, entonces ese requisito tiene un defecto.

Por ejemplo, un requisito que diga "El software debe ser muy amigable para el usuario" es incuestionable. ¿Cómo puede el Analista de Prueba determinar si el software es fácil de usar, o incluso muy fácil de usar? Si, en cambio, el requisito dice "El software debe cumplir las normas de usabilidad establecidas en el documento de normas de usabilidad, versión xxx", y si el documento de normas de usabilidad existe realmente, entonces es un requisito que presenta capacidad de ser probado. También es un requisito general porque este único requisito se aplica a todos los elementos de la interfaz. En este caso, este único requisito podría generar fácilmente muchos casos de prueba individuales en una aplicación no trivial. La trazabilidad desde este requisito, o tal vez desde el documento de normas de usabilidad, a los casos de prueba es también crítica porque si la especificación de usabilidad de referencia cambia, todos los casos de prueba deberán ser revisados y actualizados según sea necesario.

Un requisito tampoco presenta capacidad de ser probado si el probador es incapaz de determinar si la prueba pasó o falló, o es incapaz de construir una prueba que pueda pasar o fallar. Por ejemplo, "El sistema estará disponible el 100% del tiempo, 24 horas al día, 7 días a la semana, 365 (o 366) días al año" no se puede probar.

Una lista de comprobación[1] sencilla para la revisión de casos de uso puede incluir las siguientes preguntas:

1. ¿El comportamiento principal (camino) está claramente definido?
2. ¿Todos los comportamientos (caminos) alternativos están identificados, con el tratamiento de error?
3. ¿Los mensajes de la interfaz de usuario están definidos?
4. ¿Hay un solo comportamiento principal (debería haberlo, de lo contrario hay múltiples casos de uso)?
5. ¿Se puede probar cada comportamiento de forma individual?

5.1.2 Revisiones de Historias de Usuario

En un proyecto ágil, los requisitos suelen adoptar la forma de historias de usuario. Estas historias representan pequeñas unidades de funcionalidad demostrable. Mientras que un caso de uso es una transacción de usuario que atraviesa múltiples áreas de funcionalidad, una historia de usuario es una

[1] La pregunta de examen proporcionará un subconjunto de la lista de comprobación de casos de uso con la que responder a la pregunta.

característica más aislada y por lo general su desarrollo está limitado en el tiempo. Una lista de comprobación[2] para una historia de usuario podría incluir lo siguiente:

- ¿La historia es adecuada para la iteración objetivo?
- ¿La historia está escrita desde el punto de vista de la persona que la solicita?
- ¿Los criterios de aceptación están definidos y presentan capacidad de ser probados?
- ¿La prestación está claramente definida y diferenciada?
- ¿La historia es independiente de cualquier otra?
- ¿La historia está priorizada?
- ¿La historia responde al formato utilizado de forma habitual?:
- Como < tipo de usuario >, quiero < algún objetivo > de modo que < alguna razón > [Cohn04]

Si la historia define una nueva interfaz, entonces sería conveniente utilizar una lista de comprobación genérica de la historia (como la anterior) y una lista de comprobación detallada de la interfaz de usuario.

5.1.3 Adaptación de las Listas de Comprobación

Una lista de comprobación puede ser adaptada en base a lo siguiente:

- Organización (por ejemplo, teniendo en cuenta las políticas, normas, convenciones y limitaciones jurídicas de la empresa).
- Proyecto / esfuerzo de desarrollo (por ejemplo, enfoque, normas técnicas, riesgos).
- Nivel de riesgo del producto de trabajo que se está revisando.
- Técnicas de prueba que se utilizarán.

Las buenas listas de comprobación encontrarán problemas y también ayudarán a iniciar debates sobre otros temas que tal vez no se hayan mencionado específicamente en la lista de comprobación. El uso de una combinación de listas de comprobación es una forma sólida de asegurar que una revisión logre un producto de trabajo de la más alta calidad. El uso de la revisión basada en listas de comprobación con listas de comprobación estándar como las referidas en el programa de estudios de nivel básico y la elaboración de listas de comprobación específicas de la organización como las que se han mostrado anteriormente ayudarán al Analista de Prueba a ser eficaz en las revisiones.

[2] La pregunta del examen proporcionará un subconjunto de la lista de comprobación de la historia del usuario con la que responder a la pregunta.

6. Herramientas de Prueba y Automatización - 90 minutos

Palabras Clave

Español	Inglés
prueba guiada por palabra clave	keyword-driven testing
preparación de datos de prueba	test data preparation
diseño de prueba	test design
ejecución de prueba	test execution
guion de prueba	test script

Objetivos de Aprendizaje para "Herramientas de Prueba y Automatización"

6.1 Introducción		
		No hay objetivos de aprendizaje.
6.2 Automatización Guiada por Palabra Clave		
AP-6.2.1	**(K3)**	Para un escenario determinado determinar las actividades adecuadas para un Analista de Prueba en un proyecto de automatización guiado por palabra clave.
6.3 Tipos de Herramientas de Prueba		
AP-6.3.1	**(K2)**	Explicar el uso y los tipos de herramientas de prueba utilizadas en el diseño de pruebas, la preparación de datos de prueba y la ejecución de pruebas.

6.1 Introducción

Las herramientas de prueba pueden mejorar enormemente la eficiencia y la precisión de la prueba. En este capítulo se describen las herramientas de prueba y los enfoques de automatización que utiliza un Analista de Prueba. Cabe señalar que los Analistas de Prueba trabajan conjuntamente con los desarrolladores, los Ingenieros de Automatización de Pruebas y los Analistas de Pruebas Técnicas para crear soluciones de automatización de prueba. La automatización guiada por palabra clave, en particular, involucra al Analista de Prueba y aprovecha su experiencia en el negocio y la funcionalidad del sistema.

En el programa de estudio de Ingeniero de Automatización de Pruebas de Nivel Avanzado ISTQB® [ISTQB_ALTAE_SYL] se proporciona más información sobre el tema de la automatización de la prueba y el papel del Ingeniero de Automatización de Pruebas.

6.2 Automatización Guiada por Palabra Clave

El enfoque de prueba guiada por palabra clave es uno de los principales enfoques de automatización de prueba e involucra al Analista de Prueba en el suministro de los principales insumos: palabras clave y datos.

Las palabras clave (a veces denominadas palabras de acción de la prueba) se utilizan sobre todo, pero no exclusivamente, en las interacciones de negocio de alto nivel con un sistema (por ejemplo, "cancelar pedido"). Cada palabra clave suele utilizarse para representar una serie de interacciones detalladas entre un agente y el sistema sujeto a prueba. Se utilizan secuencias de palabras clave (incluidos los datos de prueba relevantes) para especificar los casos de prueba. [Buwalda02].

En la automatización de la prueba, una palabra clave se implementa como uno o más guiones de prueba ejecutables. Las herramientas leen los casos de prueba redactados como una secuencia de palabras clave que llaman a los guiones de prueba correspondientes que implementan la funcionalidad de la palabra clave. Los guiones se implementan de manera altamente modular para permitir una fácil asignación a palabras clave específicas. Se necesitan competencias de programación para implementar estos guiones modulares.

Las principales ventajas de la automatización de la prueba guiada por palabra clave son las siguientes:

- Las palabras clave que se relacionan con una aplicación o un dominio de negocio en particular pueden ser definidas por los expertos del dominio. Esto puede hacer que la tarea de especificación del caso de prueba sea más eficiente.
- Una persona con especialización principalmente en el dominio puede beneficiarse de la ejecución automática de casos de prueba (una vez que las palabras clave se han implementado como guiones) sin tener que entender el código de automatización subyacente.
- El uso de una técnica de codificación modular permite un mantenimiento eficiente de los casos de prueba por parte del Ingeniero de Automatización de Pruebas cuando se producen cambios en la funcionalidad y en la interfaz del software sujeto a prueba [Bath14].
- Las especificaciones de los casos de prueba son independientes de su implementación.

Los analistas de prueba suelen crear y mantener los datos de la palabra clave/la palabra de acción. Deben darse cuenta de que la tarea de desarrollo de guiones sigue siendo necesaria para implementar las palabras clave. Una vez que se han definido las palabras clave y los datos que se van a utilizar, el automatizador de la prueba (por ejemplo, el Analista de Pruebas técnicas o el ingeniero de automatización de pruebas) traduce las palabras clave de los procesos de negocio y las acciones de nivel inferior a guiones de prueba automatizados.

Si bien la automatización guiada por palabra clave suele ejecutarse durante la prueba de sistema, el desarrollo del código puede comenzar tan pronto como el diseño de la prueba. En un entorno iterativo, particularmente cuando se utiliza la integración continua/despliegue continuo, el desarrollo de la automatización de pruebas es un proceso continuo.

Una vez que se crean las palabras clave y los datos de entrada, el Analista de Prueba asume la responsabilidad de ejecutar los casos de prueba basados en palabras clave y de analizar cualquier fallo que pueda producirse.

Cuando se detecta una anomalía, el Analista de Prueba debe ayudar a investigar la causa del fallo para determinar si el defecto es con las palabras clave, los datos de entrada, el propio guion de automatización o con la aplicación que se está probando. Normalmente el primer paso en la resolución de problemas es ejecutar la misma prueba con los mismos datos manualmente para ver si el fallo está en la propia aplicación. Si esto no muestra un fallo, el Analista de Prueba debe revisar la secuencia de pruebas que condujo al fallo para determinar si el problema se produjo en un paso anterior (tal vez mediante la introducción de datos de entrada incorrectos), pero el defecto no apareció hasta más adelante en el procesamiento. Si el Analista de Prueba no puede determinar la causa del fallo, la información sobre la solución del problema deberá entregarse al Analista de Pruebas técnicas o al desarrollador para su posterior análisis.

6.3 Tipos de Herramientas de Prueba

Las herramientas de diseño de pruebas se utilizan para crear casos de prueba y datos de prueba para ser utilizados en las pruebas. Estas herramientas pueden funcionar a partir de formatos de documentos de requisitos específicos, modelos (por ejemplo, UML) o entradas proporcionadas por el Analista de Prueba. Las herramientas de diseño de pruebas suelen estar diseñadas y construidas para operar con formatos y herramientas particulares, como las herramientas específicas de gestión de requisitos.

Las herramientas de diseño de pruebas pueden proporcionar información para el uso del Analista de Prueba al determinar los tipos de pruebas que se necesitan para obtener el nivel de cobertura de pruebas específico deseado, la confianza en el sistema o las medidas de mitigación de riesgos de producto. Por ejemplo, las herramientas de árbol de clasificación generan (y muestran) el conjunto de combinaciones que se necesitan para alcanzar la cobertura total basándose en un criterio de cobertura seleccionado. Esta información puede ser utilizada por el Analista de Prueba para determinar los casos de prueba que deben ejecutarse.

6.3.1 Herramientas de Preparación de Datos de Prueba

Las herramientas de preparación de datos de prueba pueden aportar los siguientes beneficios:

- Analizar un documento como el documento de requisitos o incluso el código fuente para determinar los datos necesarios durante las pruebas para lograr un nivel de cobertura.

- Tomar un conjunto de datos de un sistema de producción y "limpiarlo" o anonimizarlo para eliminar cualquier información personal sin dejar de mantener la integridad interna de esos datos. Los datos depurados pueden entonces utilizarse para las pruebas sin riesgo de fuga de seguridad o de uso indebido de la información personal. Esto es particularmente importante cuando se requieren grandes volúmenes de datos realistas y cuando se aplican riesgos de seguridad y de privacidad de los datos.
- Generar datos de prueba sintéticos a partir de determinados conjuntos de parámetros de entrada (por ejemplo, para su utilización en pruebas aleatorias). Algunos de estas herramientas analizarán la estructura de la base de datos para determinar qué entradas se requerirán del Analista de Prueba.

6.3.2 Herramientas de Ejecución de Pruebas Automatizadas

Las herramientas de ejecución de pruebas son utilizadas por los Analistas de Prueba en todos los niveles de prueba para ejecutar pruebas automatizadas y comprobar el resultado de las mismas. El objetivo de usar una herramienta de ejecución de pruebas es habitualmente uno o más de los siguientes:

- Reducir costes (en términos de esfuerzo y/o tiempo).
- Realizar más pruebas.
- Realizar la misma prueba en diferentes entornos.
- Hacer que la ejecución de la prueba sea más repetible.
- Realizar pruebas que sería imposible ejecutar manualmente (es decir, pruebas para la validación de grandes volúmenes de datos).

Con frecuencia estos objetivos se solapan con los objetivos principales de aumentar la cobertura y reducir los costes.

El retorno de la inversión en herramientas de ejecución de pruebas suele ser mayor cuando se automatizan las pruebas de regresión debido al bajo nivel de mantenimiento esperado y a la ejecución repetida de las pruebas. La automatización de las pruebas de humo también puede ser un uso eficaz de la automatización debido al uso frecuente de las pruebas, la necesidad de un resultado rápido y, aunque el coste de mantenimiento puede ser mayor, la capacidad de tener una forma automatizada de evaluar una nueva construcción en un entorno de integración continua.

Normalmente se utilizan herramientas de ejecución de pruebas durante la prueba de sistema e integración. Algunas herramientas, en particular las herramientas de prueba de la API, también pueden utilizarse en las pruebas de componentes. Aprovechar las herramientas donde son más aplicables ayudará a mejorar el retorno de la inversión.

7. Referencias

7.1 Estándares

- [ISO25010] ISO/IEC 25010 (2014) Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) System and software quality models

  Chapter 4

- [ISO29119-4] ISO/IEC/IEEE 29119-4 Software and Systems Engineering – Software Testing – Part 4, Test Techniques, 2015

- [RTCA DO-178C/ED-12C]: Software Considerations in Airborne Systems and Equipment Certification, RTCA/EUROCAE ED12C, 2013.

  Chapter 1

7.2 Documentos ISTQB e IREB

• [IREB_CPRE]	IREB Certified Professional for Requirements Engineering, Foundation Level Syllabus, Version 2.2.2, 2017
• [ISTQB_AGILE_SYL]	ISTQB® Foundation Agile Software Testing, Version 2014
• [ISTQB_AL_OVIEW]	ISTQB® Advanced Level Overview, Version 2.0
• [ISTQB_ALTM_SYL]	ISTQB® Advanced Level Test Manager Syllabus, Version 2019
• [ISTQB_ALTAE_SYL]	ISTQB® Advanced Level Test Automation Engineer Syllabus, Version 2017
• [ISTQB_ALTTA_SYL]	ISTQB® Advanced Level Technical Test Analyst Syllabus, Version 2019
• [ISTQB_FL_SYL]	ISTQB® Foundation Level Syllabus, Version 2018
• [ISTQB_FL_UT]	ISTQB® Foundation Level Specialist Syllabus Usability Testing, Version 2018
• [ISTQB_GLOSSARY]	Standard glossary of terms used in Software Testing, Version 3.2, 2018

7.3 Referencias Bibliográficas

[Bath14] Graham Bath, Judy McKay, “The Software Test Engineer’s Handbook (2nd Edition)”, Rocky Nook, 2014, ISBN 978-1-933952-24-6

[Beizer95] Boris Beizer, “Black-box Testing”, John Wiley & Sons, 1995, ISBN 0-471-12094-4

[Black02]: Rex Black, “Managing the Testing Process (2nd edition)”, John Wiley & Sons: New York, 2002, ISBN 0-471-22398-0

[Black07]: Rex Black, “Pragmatic software testing: Becoming an effective and efficient test professional”, John Wiley and Sons, 2007, ISBN 978-0-470-12790-2

[Black09]: Rex Black, “Advanced Software Testing, Volume 1”, Rocky Nook, 2009, ISBN 978-1-933-952-19-2

[Buwalda02]: Hans Buwalda, “Integrated Test Design and Automation: Using the Test Frame Method”, Addison-Wesley Longman, 2002, ISBN 0-201-73725-6

[Cohn04]: Mike Cohn, “User Stories Applied: For Agile Software Development”, Addison-Wesley Professional, 2004, ISBN 0-321-20568-5

[Copeland04]: Lee Copeland, “A Practitioner's Guide to Software Test Design”, Artech House, 2004, ISBN 1-58053-791-X

[Craig02]: Rick David Craig, Stefan P. Jaskiel, “Systematic Software Testing”, Artech House, 2002, ISBN 1-580-53508-9

[Gilb93]: Tom Gilb, Graham Dorothy, “Software Inspection”, Addison-Wesley, 1993, ISBN 0-201-63181-4

[Koomen06]: Tim Koomen, Leo van der Aalst, Bart Broekman, Michiel Vroon "TMap NEXT, for result driven testing", UTN Publishers, 2006, ISBN 90-72194-80-2

[Kuhn16]: D. Richard Kuhn et al, “Introduction to Combinatorial Testing, CRC Press, 2016, ISBN 978-0-429-18515-1

[Myers11]: Glenford J. Myers, “The Art of Software Testing 3rd Edition”, John Wiley & Sons, 2011, ISBN: 978-1-118-03196-4

[Offutt16]: Jeff Offutt, Paul Ammann, Introduction to Software Testing – 2nd Edition, Cambridge University Press, 2016, ISBN 13: 9781107172012

[vanVeenendaal12]: Erik van Veenendaal, “Practical risk-based testing." Product Risk Management: The PRISMA Method ”, UTN Publishers, The Netherlands, ISBN 9789490986070

[Wiegers03]: Karl Wiegers, “Software Requirements 2”, Microsoft Press, 2003, ISBN 0-735-61879-8

[Whittaker03]: James Whittaker, “How to Break Software”, Addison-Wesley, 2003, ISBN 0-201-79619-8

[Whittaker09]: James Whittaker, “Exploratory software testing: tips, tricks, tours, and techniques to guide test design”, Addison-Wesley, 2009, ISBN 0-321-63641-4

7.4 Otras Referencias

Las siguientes referencias apuntan a información disponible en Internet y en otras fuentes. A pesar de que estas referencias fueron consultadas en el momento de la publicación de este programa de estudio de Nivel Avanzado, el ISTQB no se responsabiliza de la disponibilidad de tales referencias.

- Capítulo 3

  - Czerwonka, Jacek: www.pairwise.org

  - Bug Taxonomy: www.testingeducation.org/a/bsct2.pdf

  - Sample Bug Taxonomy based on Boris Beizer’s work: inet.uni2.dk/~vinter/bugtaxst.doc

  - Good overview of various taxonomies: testingeducation.org/a/bugtax.pdf

  - Heuristic Risk-Based Testing By James Bach

- Exploring Exploratory Testing, Cem Kaner and Andy Tinkham, www.kaner.com/pdfs/ExploringExploratoryTesting.pdf

- Pettichord, Bret, “An Exploratory Testing Workshop Report”, www.testingcraft.com/exploratorypettichord

- Capítulo 5

http://www.tmap.net/checklists-and-templates

8. Apéndice A

La siguiente tabla se ha obtenido a partir de la tabla completa proporcionada en la norma ISO 25010. Se centra únicamente en las características de calidad que figuran en el programa de estudio de Analista de Prueba, y compara los términos utilizados en la norma ISO 9126 (en la versión de 2012 del programa de estudios) con los de la norma más reciente ISO 25010 (en la versión actual).

ISO/IEC 25010	ISO/IEC 9126-1	Notas
Adecuación funcional	Funcionalidad	
Completitud funcional		
Corrección funcional	Exactitud	
Pertinencia funcional	Adecuación	
	Interoperabilidad	**Ha pasado a la Compatibilidad.**
Usabilidad		
Capacidad de reconocimiento de la pertinencia	Capacidad de ser entendido	**Nuevo nombre.**
Capacidad de ser aprendido	Capacidad de ser aprendido	
Capacidad de ser operado	Operabilidad	
Protección ante error de usuario		**Nueva subcaracterística.**
Estética de interfaz de usuario	Atractivo	**Nuevo nombre.**
Accesibilidad		**Nueva subcaracterística.**
Compatibilidad		**Nueva definición.**
Coexistencia		**Cubierta en Analista de Pruebas Técnicas.**
Interoperabilidad		

9. Índice Terminológico