

Capítulo 1

Fundamentos de las pruebas



Capítulo 1

- 1.1 ¿Qué es poner a prueba?
- 1.2 ¿Por qué son necesarias las pruebas?
- 1.3 Siete Principios de las Pruebas
- 1.4 Proceso de Prueba
- 1.5 La Psicología de las Pruebas





1.1 ¿Qué es poner a prueba?

Probar el software es una forma de evaluar su calidad y de reducir el riesgo de fallos en un entorno de operaciones o en producción.

El proceso de prueba también incluye actividades tales como planificar la prueba, analizar, diseñar e implementar pruebas, informar del avance y de los resultados de la prueba, y evaluar la calidad de un objeto de prueba.

Prueba estática

No se ejecuta un componente

VS

Prueba dinámica

Sí se ejecuta un componente

P/ Es cierta o falsa esta afirmación?

Las pruebas de software consisten en ejecutar el software y comprobar los resultados.

R/ Falsa, también implican validar, si el sistema es capaz de satisfacer las necesidades de los usuarios y de otros implicados en su(s) entorno(s) de operación.



1.1.1 Objetivos Típicos de las Pruebas

1. Evaluar productos de trabajo tales como requisitos, historias de usuario, diseño y código.
2. Verificar el cumplimiento de todos los requisitos especificados.
3. Validar si el objeto de prueba está completo y funciona como los usuarios y otros implicados esperan.
4. Generar confianza en el nivel de calidad del objeto de prueba.
5. Prevenir defectos.
6. Encontrar fallos y defectos.
7. Proporcionar suficiente información a los implicados para que puedan tomar decisiones informadas, especialmente en relación con el nivel de calidad del objeto de prueba.
8. Reducir el nivel de riesgo de calidad inadecuada del software (por ejemplo, fallos que se producen durante la operación que no han sido detectados anteriormente).
9. Cumplir con requisitos o normas contractuales, legales o reglamentarias, y/o verificar el cumplimiento de dichos requisitos o normas por parte del objeto de prueba.

1.1.1 Objetivos Típicos de las Pruebas

Los objetivos de la prueba pueden variar, dependiendo del contexto del componente o sistema que se está probando, el nivel de prueba y el modelo de ciclo de vida de desarrollo de software.

Criticidad del objeto de prueba

- Alta: médica, automotor, aeronáutica...
- Baja: juegos gratis...

Nivel de prueba

- Componente (código)
- Integración (interfaces)
- Aceptación (involucrando representantes de los usuarios)

Ciclo de Vida de Desarrollo de Software

- Secuencial (Cascada, V, ...)
- Ágil (Scrum, ...)

1.1.2 Las pruebas y la depuración

Las Pruebas (Probador) y la depuración (**Desarrollador**) son diferentes



En el desarrollo Ágil, una misma persona puede estar en el rol de desarrollador y en el de probador en diferentes pasos del desarrollo

Capítulo 1

- 1.1 ¿Qué es poner a prueba?
- 1.2 ¿Por qué son necesarias las pruebas?
- 1.3 Siete Principios de las Pruebas
- 1.4 Proceso de Prueba
- 1.5 La Psicología de las Pruebas





1.2 ¿Por qué son necesarias las pruebas?

Ejercicio:

¿Qué respondería a la pregunta anterior?

Utilidad de las pruebas

- Permiten medir la calidad del software
- Aportan confianza a la calidad del software
- Al detectar defectos, su corrección incrementará la calidad del software
- Mejora la calidad de sistemas futuros al entender la causa de los defectos detectados en proyectos anteriores.

1.2.1 Aportes de las Pruebas al Éxito

Los probadores deben participar desde el principio del proyecto

Participar en las revisiones

- Especificaciones funcionales
- Requisitos
- Historias de usuario



Participar en las revisiones de diseño

- Talleres
- Revisiones de la arquitectura funcional
- Revisiones de la arquitectura técnica

Ventajas

- Detección temprana de defectos
- Asegurar la capacidad de ser probado de las bases de prueba
- Usar habilidades complementarias



1.2.1 Aportes de las Pruebas al Éxito

Los probadores también deben participar durante el desarrollo

«Programación en parejas»

- Una persona a cargo de la codificación (rol = desarrollador)
- Una persona a cargo de comentar, asesorar y revisar (role = probador)

Revisión del código

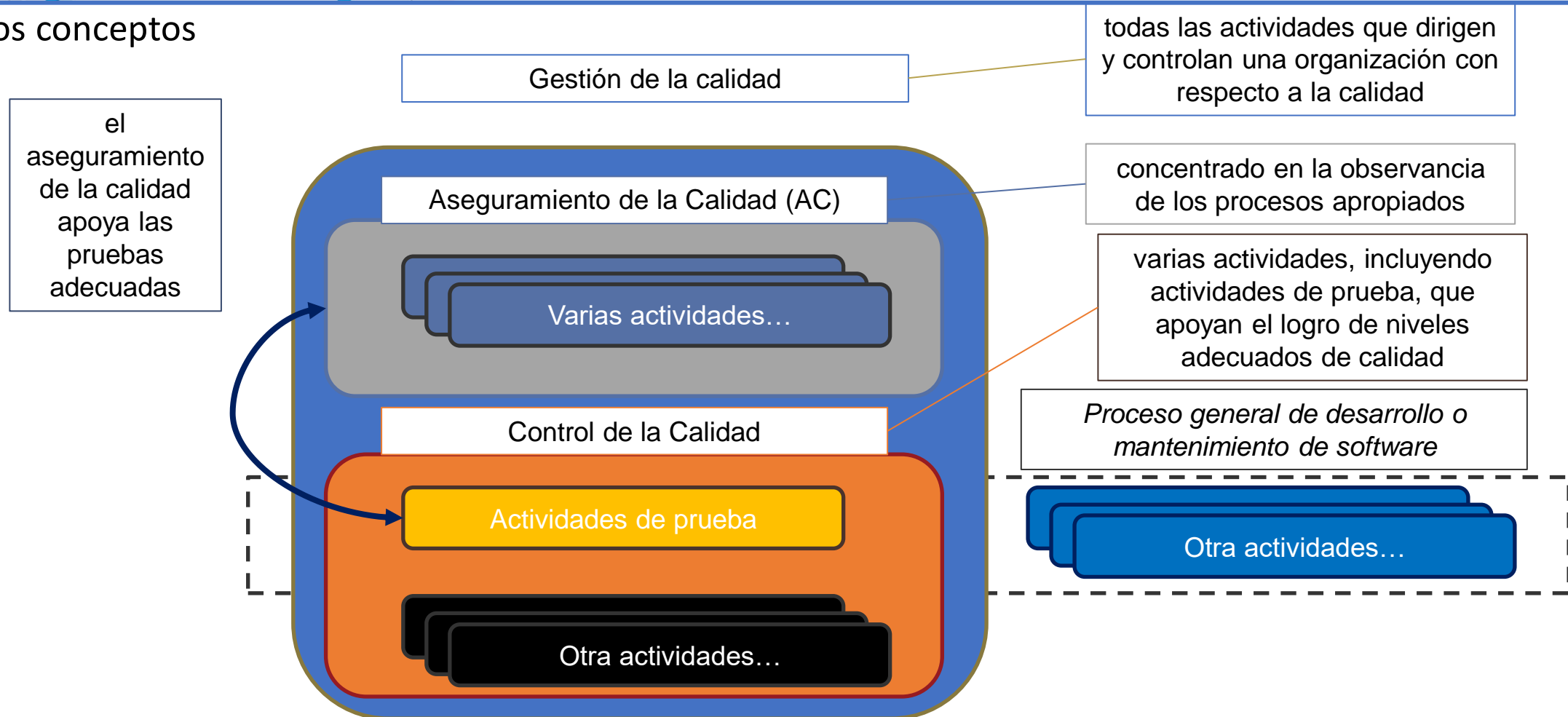
- Manual o automatizado

Automatización de prueba unitaria

- Una persona en el rol de un probador pero con habilidades de desarrollo

1.2.2 Aseguramiento de la Calidad y las Pruebas

•Varios conceptos



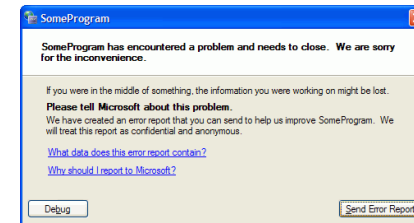
1.2.2 Errores, defectos y fallos

Del error al fallo

Error (equivocación):
origen humano

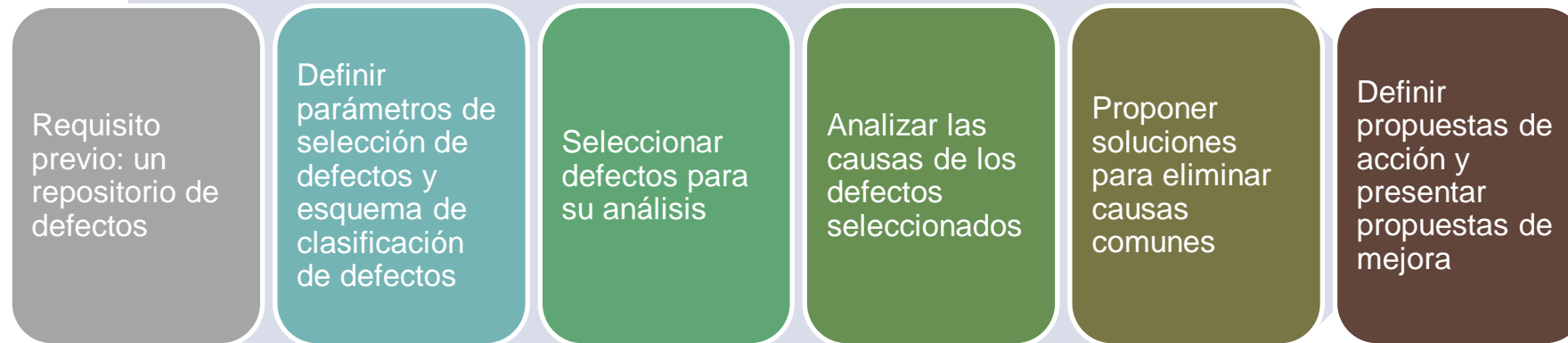
Defecto (bug): en el código o
la documentación, como
resultado de un error

Fallo: resultado de ejecutar un código
defectuoso; el sistema funciona o no
funciona o funciona de manera diferente



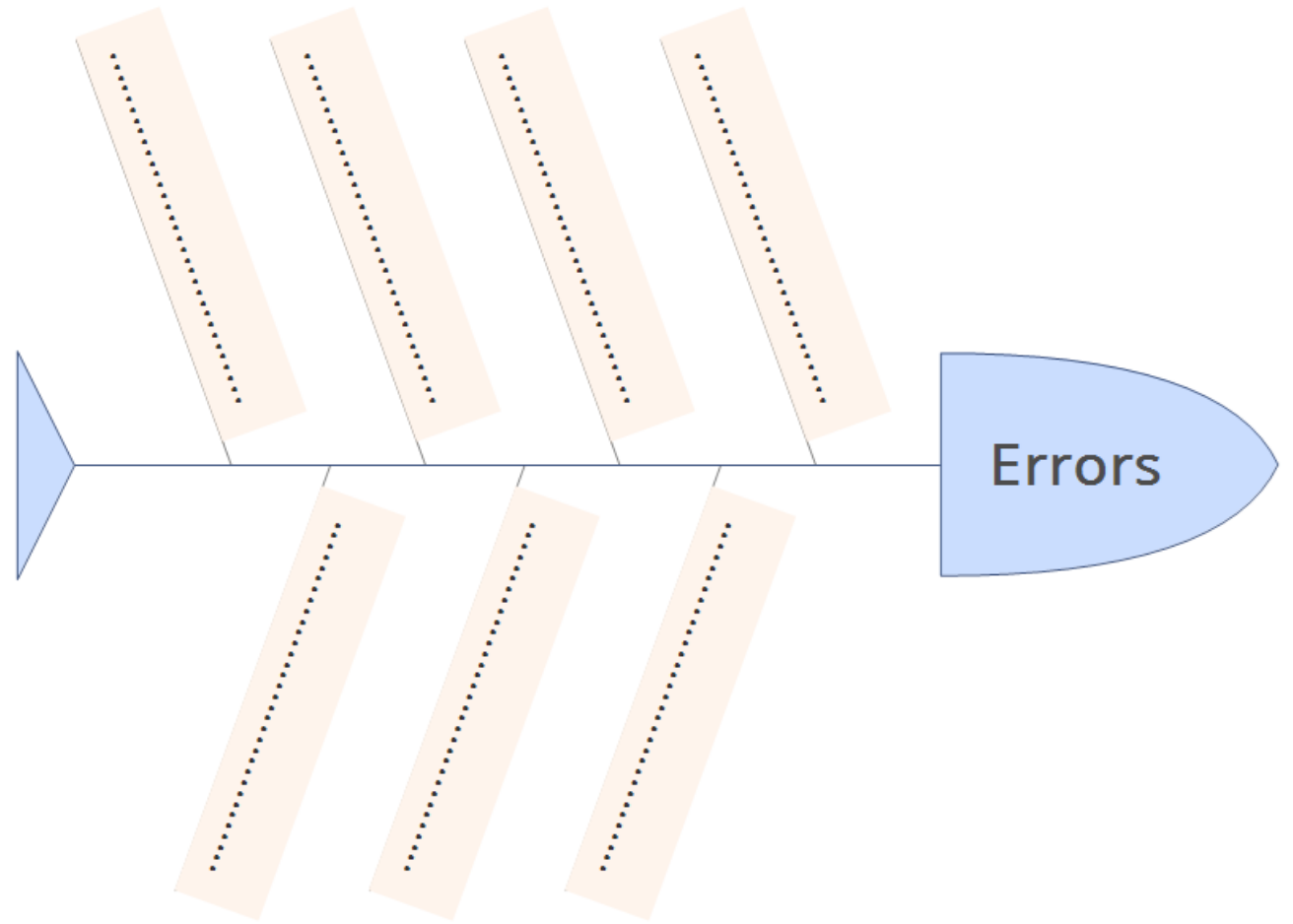
1.2.3 Defectos, Causa Raíz y Efectos

Uso del Análisis de Causa Raíz (ACR) para detectar y eliminar las causas de los defectos

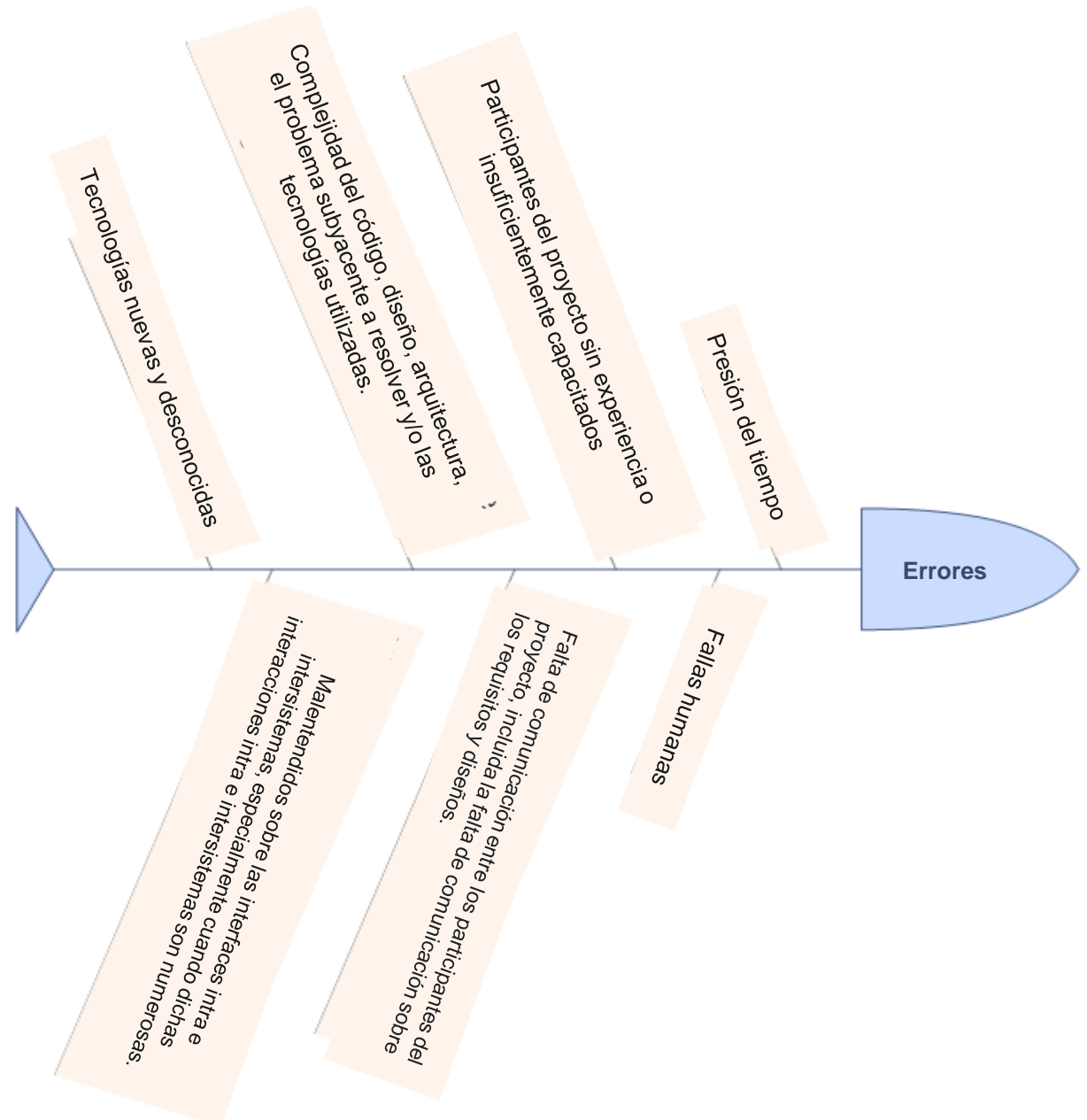


1.2.3 Errores, defectos y fallos

- Ejercicio:
Complete el diagrama de FishBone con las causas más comunes de errores en los proyectos



•Causas proporcionadas por la ISTQB





1.2.3 Errores, defectos y fallos

¡Los resultados de las pruebas pueden estar equivocados!

Falsos positivos

- Pruebas reportadas como fallidas, con defectos que en realidad no lo son

Falsos negativos

- son pruebas que no detectan defectos que deberían haber detectado

1.2.4 Defectos, Causas Raíz y Efectos

•Ejercicio: piense en un defecto y use los «5 por qué» para identificar sus causas raíz

•Mencione un defecto:

Your defect:

?

...

?

...

?

...

?

...

?

...



1.2.4 Defectos, Causas Raíz y Efectos

Ejercicio: vincule cada término de la lista a la izquierda a una categoría en la columna a la derecha

pagos de intereses incorrectos

cálculo incorrecto en el código

quejas del cliente

ambigüedad en la historia de usuario

falta de conocimiento por parte del propietario del producto

defecto original

fallo

defecto

efectos

causas raíz del defecto original

1.2.4 Defectos, Causas Raíz y Efectos

Ejercicio: vincule cada término de la lista a la izquierda a una categoría en la columna a la derecha

pagos de intereses incorrectos

cálculo incorrecto en el código

quejas del cliente

ambigüedad en la historia de usuario

falta de conocimiento por parte del propietario del producto

defecto original

fallo

defecto

efectos

causas raíz del defecto original

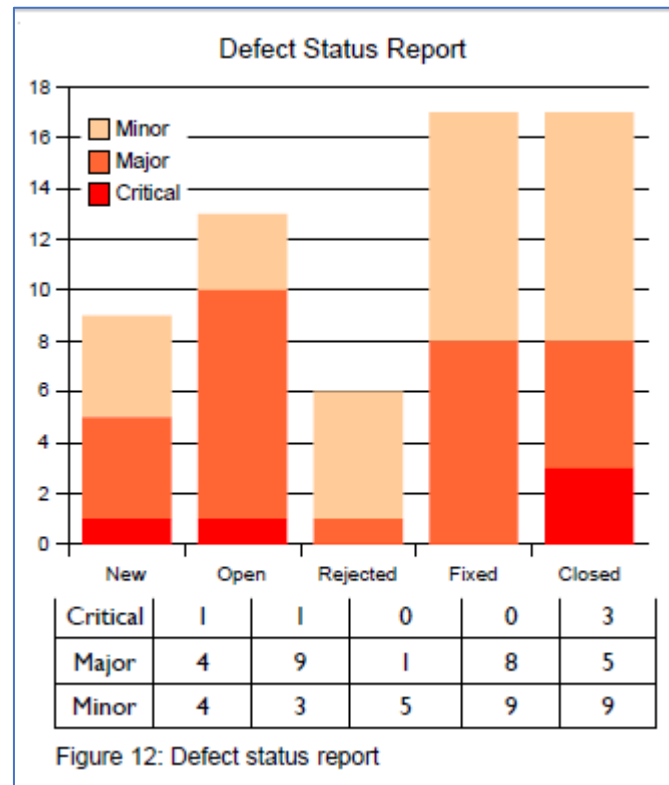
Capítulo 1

- 1.1 ¿Qué es poner a prueba?
- 1.2 ¿Por qué son necesarias las pruebas
- 1.3 Siete Principios de las Pruebas
- 1.4 Proceso de Prueba
- 1.5 La Psicología de las Pruebas



1.3.1 Las pruebas muestran la presencia de defectos, no su ausencia

Informe de defectos: una información necesaria, pero no suficiente



La corrección de los defectos encontrados mejora la calidad del software

A pesar de que

- Es necesario medir el alcance de las características de software funcionales y no funcionales cubiertas por la prueba
- La estrategia de prueba debe permitir que las pruebas hallen el máximo número de defectos
- Los resultados positivos de las pruebas no significan “cero defectos”.



1.3.2 Las pruebas exhaustivas son imposibles

Qué son las Pruebas exhaustivas

- Un enfoque de prueba en el que el juego de pruebas comprende todas las combinaciones de valores de entrada y condiciones previas.



1.3.2 Las pruebas exhaustivas son imposibles

- Es imposible probarlo todo
- Un número excesivo de combinaciones de prueba
 - Funcionalidad
 - Estudios de viabilidad comercial específicos
 - Entornos de producción
 - Conjunto de datos
 - Instrucciones del código
- Limitaciones de tiempo y de presupuesto

Es necesario determinar lo que se debe probar, hasta qué punto es mensurable (cobertura funcional o técnica, anomalías residuales) y con qué recursos.



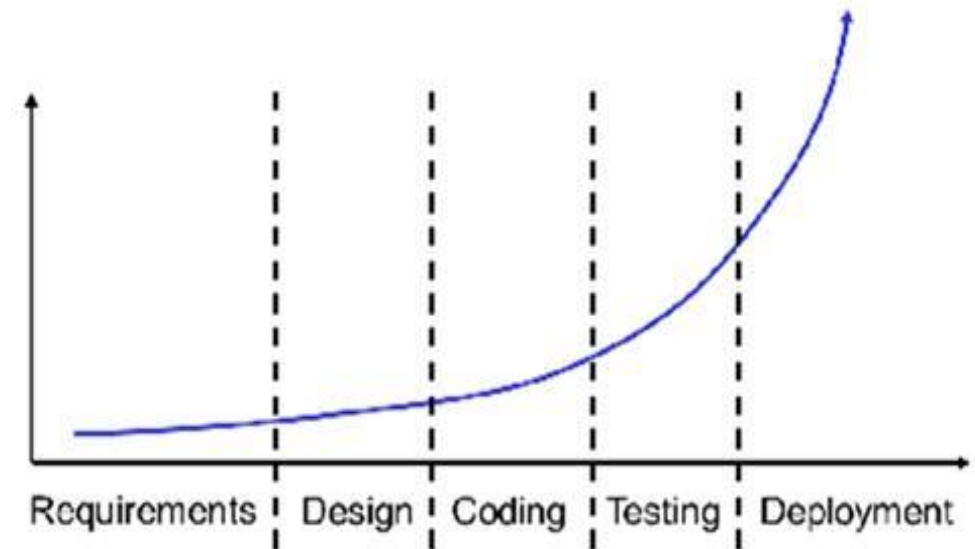
1.3.3 Las pruebas tempranas ahorran tiempo y dinero

- Descubrir y eliminar los defectos en la etapa más temprana posible
- Numerosos defectos se introducen muy temprano, incluso durante la especificación funcional
- La detección temprana de los defectos evitará que se propaguen y multipliquen en las fases posteriores del proyecto
- Los costos de corrección relacionados con un defecto aumentan a medida que se encuentran más tarde

El objetivo debe ser una reducción máxima del tiempo entre la introducción de un defecto y su detección

1.3.3 Las pruebas tempranas ahorran tiempo y dinero

Mientras **más tarde** se detecte un defecto **mayor será su costo** para la corrección del mismo. Por lo que las pruebas buscan detectar estos defectos antes de que se realice el paso a producción.



1.3.4 Los defectos se agrupan

- “Los defectos son como una piara de cerdos”



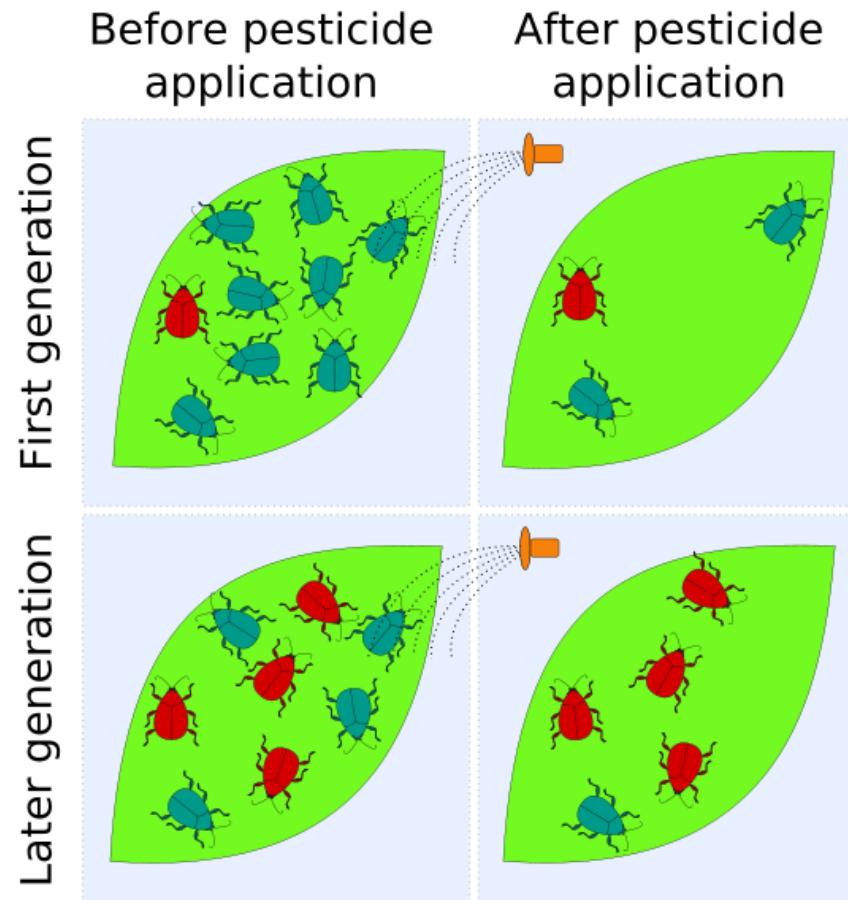


1.3.4 Los defectos se agrupan

- Un principio no probado pero útil para aplicarse
- Determinar el esfuerzo de prueba al comienzo del proyecto
 - Basado en resultados de proyectos similares o versiones anteriores
 - Basado en la experiencia de arquitectos, desarrolladores y probadores
- Determinar el esfuerzo de prueba durante la fase de ejecución
 - Ejecutar la prueba de los módulos con la mayoría de los bugs con mayor prioridad

1.3.5 Cuidado con la paradoja del pesticida

- Los casos de prueba necesitan ser actualizados



- Los pesticidas, al igual que el mismo juego de pruebas, no matan a todos los insectos
- Después de varias aplicaciones, los pesticidas ya no matan a los insectos que se volvieron resistentes



1.3.6 Las pruebas dependen del contexto

Es necesario tener en cuenta los contextos antes de decidir qué enfoque de prueba aplicar

Contexto
“empresarial” de uso
del software

- ¿En qué área comercial se usará el software?
- ¿Cuál es su criticidad?

Contexto(s) de
operación del
software

- ¿En qué entornos tendrá que operar el software?
- ¿En qué plataformas de hardware y software?

Antecedentes del
proyecto

- ¿Cuáles son los requisitos de tiempo? (tiempo para comercializarlo)
- ¿Cuál es la asignación presupuestaria?
- ¿De qué manera se desarrolló el proyecto? (¿Ágil?)

1.3.7 La ausencia de errores es una falacia/creencia equivocada

- La ausencia de errores no refleja la calidad del software



- Los defectos que no se detectaron durante las pruebas aún podrían ser encontrados en una fase posterior.
- Si el software no cumple con los requisitos del cliente final, para este estará defectuoso, incluso si no se encontraron defectos durante la prueba

Capítulo 1

- 1.1 ¿Qué es poner a prueba?
- 1.2 ¿Por qué son necesarias las pruebas
- 1.3 Siete Principios de las Pruebas
- 1.4 Proceso de Prueba
- 1.5 La Psicología de las Pruebas





1.4 Proceso de Prueba

Conjunto de actividades de prueba comunes con las cuales es más probable que la prueba alcance los objetivos establecidos.



1.4.1 Proceso de Prueba en Contexto

La descripción y la implementación del proceso de prueba dependen del «contexto»

Dominio empresarial

- Seguridad crítica, o no
- Regulado por normas y estándares

Ciclo de vida de desarrollo de software

- Secuencial (Ciclo V)
- Iterativa (Ágil)

Limitaciones de funcionamiento

- Presupuestos y recursos
- Plazos
- Complejidad

1.4.2 Actividades y tareas de prueba

Se necesitan 7
grupos de
actividades

1. Planificación de prueba

- Cómo gestionar las actividades de prueba en el proyecto

2. Monitorización y control de la prueba

- Dar seguimiento a la implementación del Plan de Pruebas y ajustarlo si es necesario

3. Análisis de prueba

- Identificar qué probar

4. Diseño de prueba

- Documentar los casos de prueba de alto nivel

5. Implementación de prueba

- Prepararse para iniciar la ejecución de prueba

6. Ejecución de prueba

- ¡Ejecute los casos de prueba, registre los resultados e informe los defectos (si los hay)!

7. Compleción de la prueba

- Informe final sobre las actividades de prueba e identificación de posibles mejoras

1.4.2 Actividades y tareas de prueba

Planificación de prueba

- Véa el 5.2
- Objetivo: definir los objetivos de la prueba y el enfoque para cumplir los objetivos de prueba





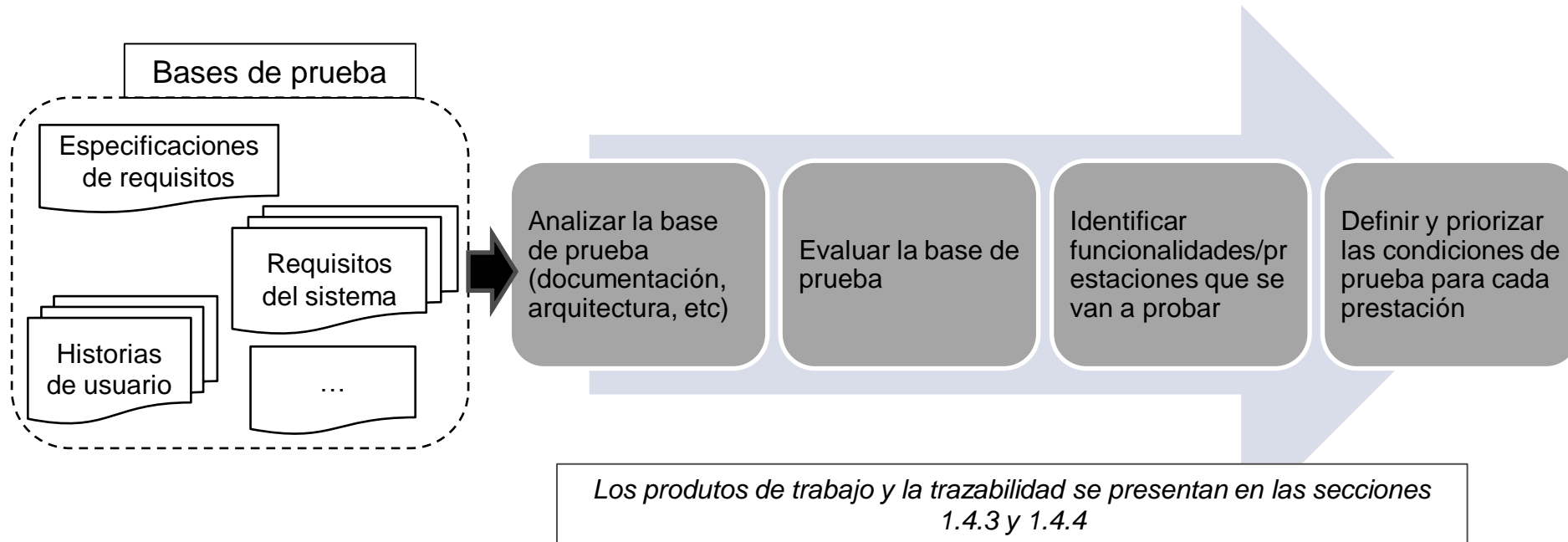
1.4.2 Actividades y tareas de prueba

Monitorización/seguimiento y control de la prueba

- Véa el 5.3
- Objetivos:
 - Comparación continua del avance real con el esperado de acuerdo al plan de prueba, utilizando cualquier métrica de seguimiento.
 - Tomar las acciones necesarias para cumplir con los objetivos del plan de prueba

1.4.2 Actividades y tareas de prueba

Análisis de prueba (Qué probar?)





1.4.2 Actividades y tareas de prueba

- **Observaciones sobre el diseño de casos de prueba**
- Algunos objetos de prueba pueden ser tratados de forma más adecuada definiendo condiciones de prueba en lugar de pasos detallados. Utilizar condiciones de prueba para pruebas improvisadas (“unscripted testing”)
- Los criterios de paso/fallo deben estar definidos de forma clara
- Los casos de prueba deben poder ser comprendidos por otros probadores de tal forma que puedan entender el objetivo de la prueba y su importancia
- Los casos de prueba deben poder ser comprendidos por otros implicados (por ejemplo, desarrolladores, auditores o expertos del negocio/industria) para revisiones y mejora
- Los casos de prueba deben cubrir todas las interacciones del software con actores. No sólo las interacciones a través de la interfaz de usuario
- Las interfaces entre los diversos objetos de prueba deben ser probadas

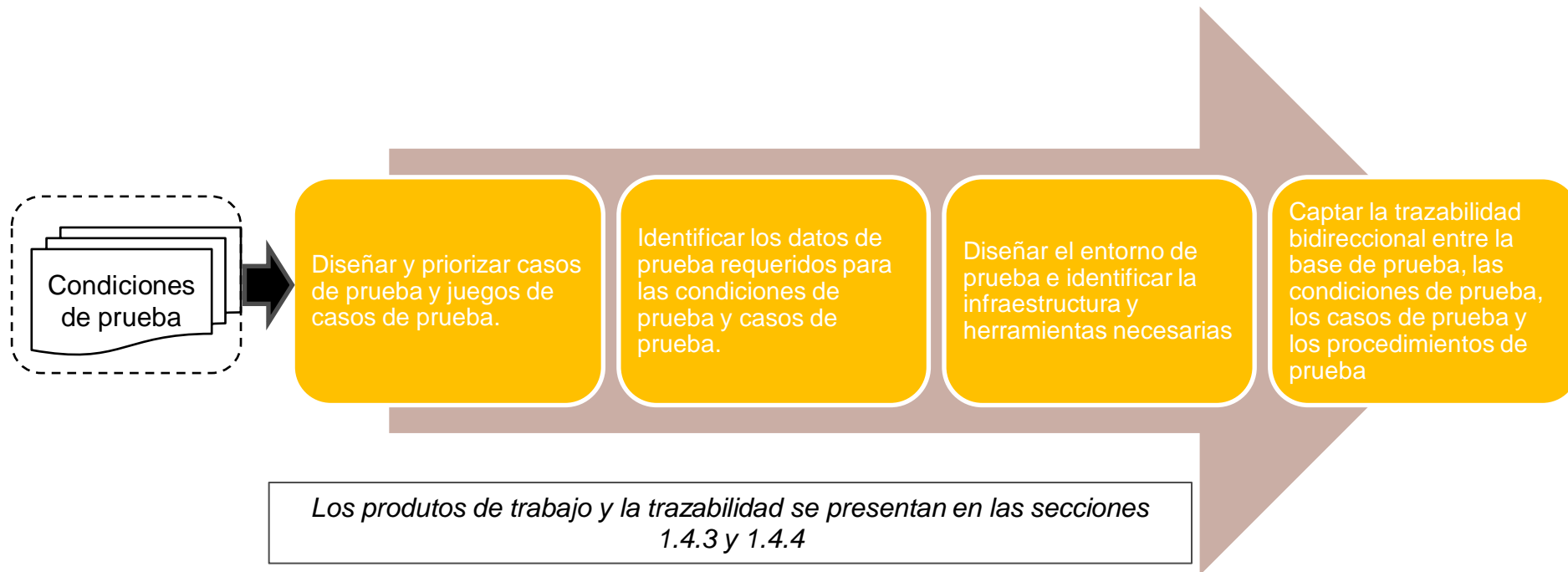


1.4.2 Actividades y tareas de prueba

- Casos de prueba
 - La descripción de un caso de prueba consiste en la siguiente información (de acuerdo al IEEE 829)
 - **Una identificación única**
 - **Elementos de prueba**
 - Una lista de precondiciones (situación requerida)
 - **Un conjunto de valores de entrada**
 - **Un conjunto de resultados esperados correspondientes**
 - Pos condiciones esperadas
 - **Interdependencia con otros casos de prueba**
 - **Requisitos especiales de procedimiento**
 - Referencia a los requisitos que serán objeto de prueba
 - **Requisitos de entorno**

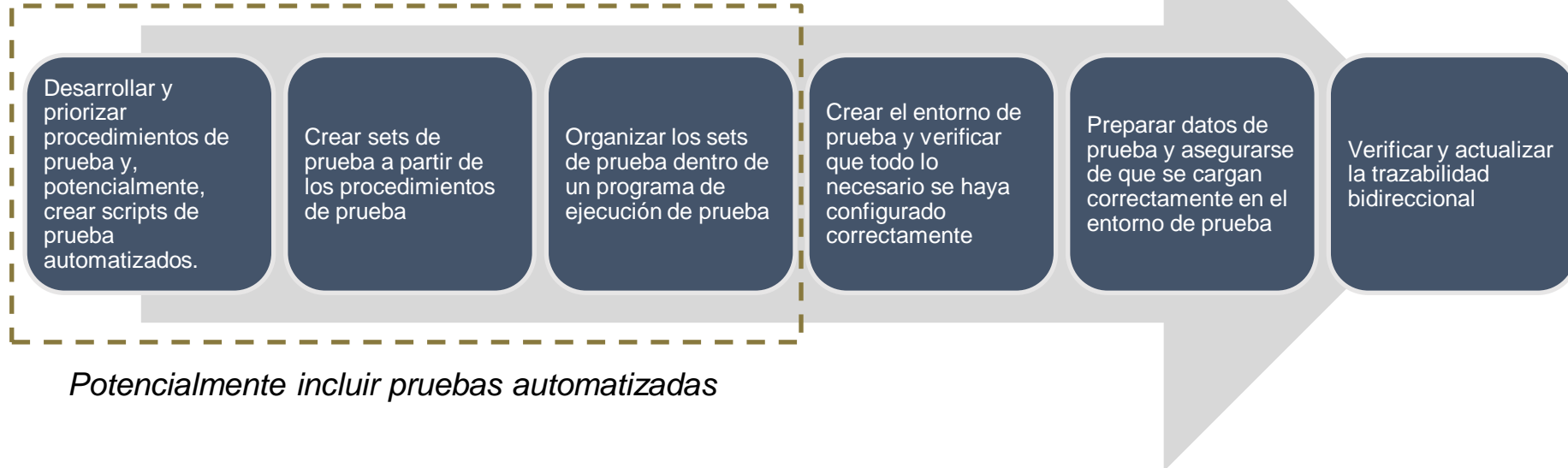
1.4.2 Actividades y tareas de prueba

Diseño de las pruebas (Cómo probar?)



1.4.2 Actividades y tareas de prueba

Implementación de prueba (Está todo preparado para realizar la prueba?)



1.4.2 Actividades y tareas de prueba

Ejecución de prueba

Registrar los ID y las versiones de los elementos de prueba o del objeto de prueba, las herramientas de prueba

Ejecutar pruebas ya sea manualmente o empleando herramientas de ejecución de prueba

Comparar los resultados reales con los resultados esperados

Analizar las anomalías para establecer sus posibles causas

Reportar los defectos basados en los fallos observados

Registrar el resultado de la ejecución de prueba (p. ej., pasa, falla, bloqueada).

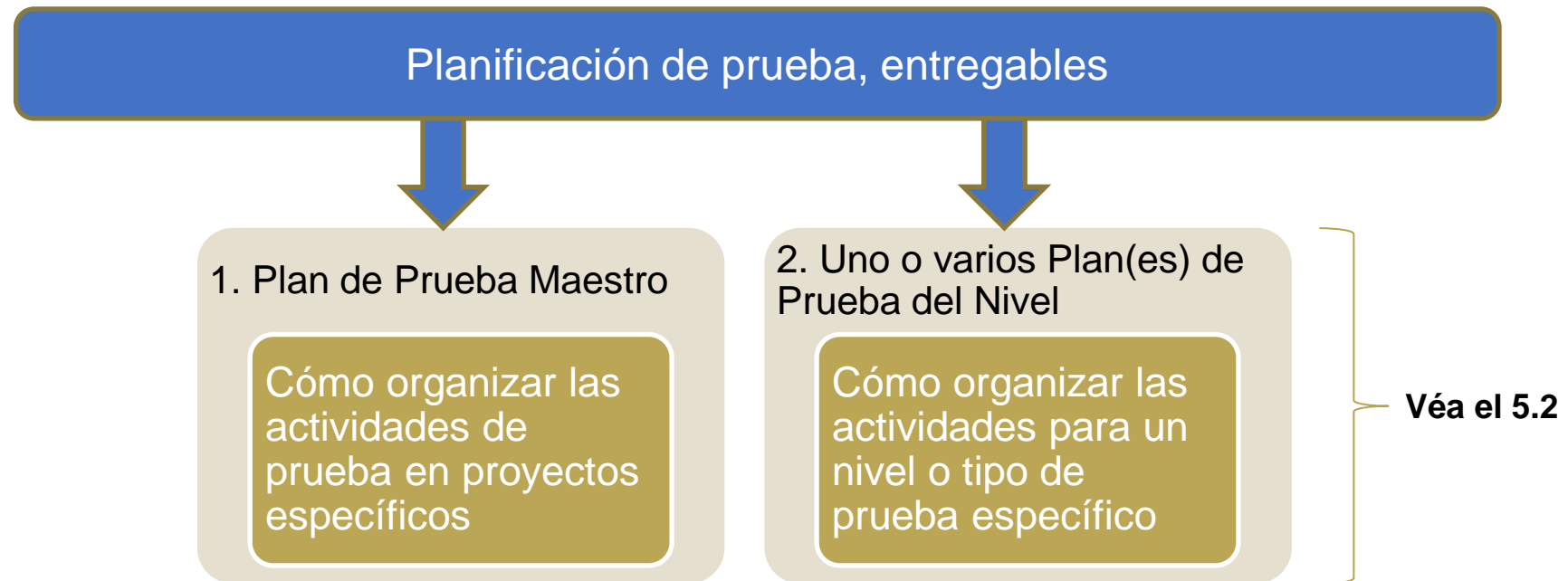
Repetir las actividades de prueba (p. ej., pruebas de confirmación y/o regresión)

Verificar y actualizar la trazabilidad bidireccional

1.4.2 Actividades y tareas de prueba



1.4.3 Entregables de prueba



1.4.3 Entregables de prueba

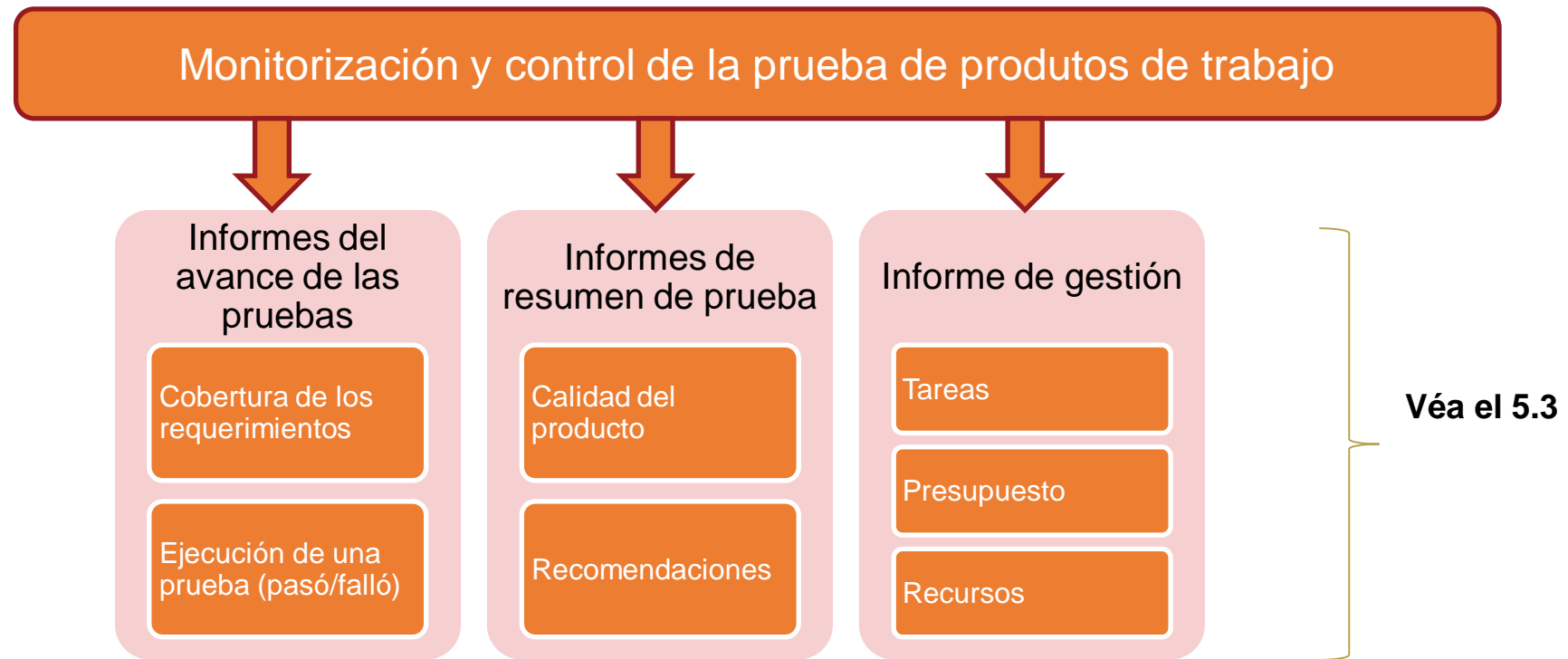
•Planificación de prueba de productos de trabajo

1	INTRODUCTION.....
1.1	PURPOSE.....
1.2	REFERENCES.....
2	PROJECT OVERVIEW.....
2.1	BUSINESS TTM MILESTONES.....
2.2	KEY PROJECT AND TESTING MILESTONES
3	TEST ITEMS.....
4	FEATURES TO BE TESTED.....
5	FEATURES NOT TO BE TESTED.....
6	TEST STRATEGY.....
6.1	CRITICITY OF TESTED FEATURES
6.2	TEST EFFORTS.....
6.3	TEST LEVELS.....
6.4	TESTING TECHNIQUES
6.5	PRIORITISATION OF TESTS EXECUTION
6.6	TESTS AUTOMATION
6.7	TESTS PROGRESS MONITORING AND CONTROL, METRICS.....
6.8	CONFIGURATION MANAGEMENT CONSIDERATIONS
6.9	DEFECTS MANAGEMENT
6.10	WAY OF USING TESTING TOOLS.....
6.11	QUALITY CONSIDERATIONS.....
7	ENVIRONMENTAL NEEDS.....

•Muestra de plan de prueba:

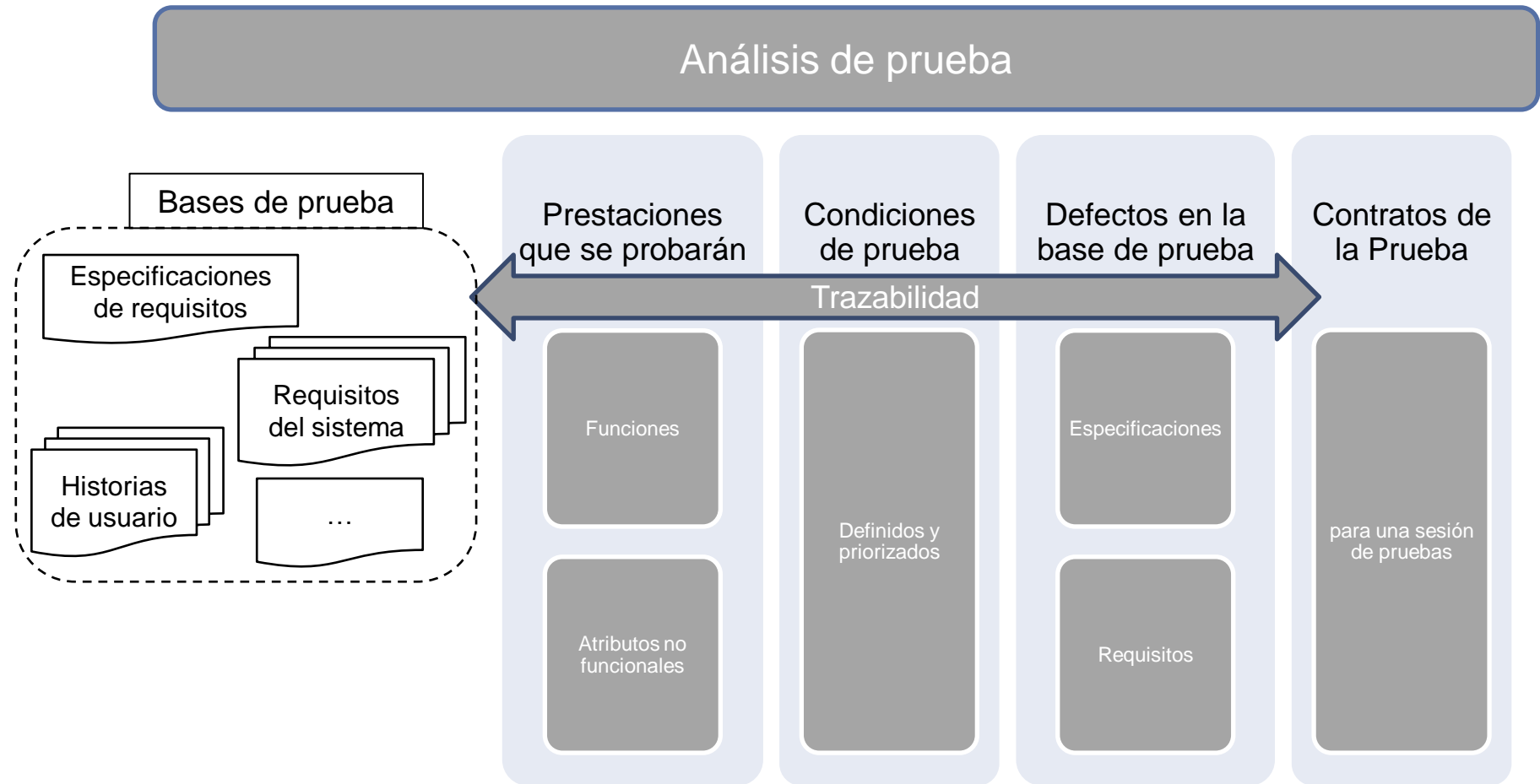
8	STAFFING AND TRAINING NEEDS.....
9	TESTING TASKS AND RESPONSIBILITIES.....
9.1	ROLES AND RESPONSIBILITIES
9.2	TEST PREPARATION.....
9.3	TEST EXECUTION
10	SUSPENSION CRITERIA AND RESUMPTION REQUIREMENTS.....
11	TESTS DELIVERABLES
12	RISKS AND CONTINGENCIES.....
13	ITEM PASS/FAIL CRITERIA.....
14	ADDITIONAL CONTEXT SPECIFIC INFORMATION
	APPENDIX A - GLOSSARY.....

1.4.3 Entregables de prueba



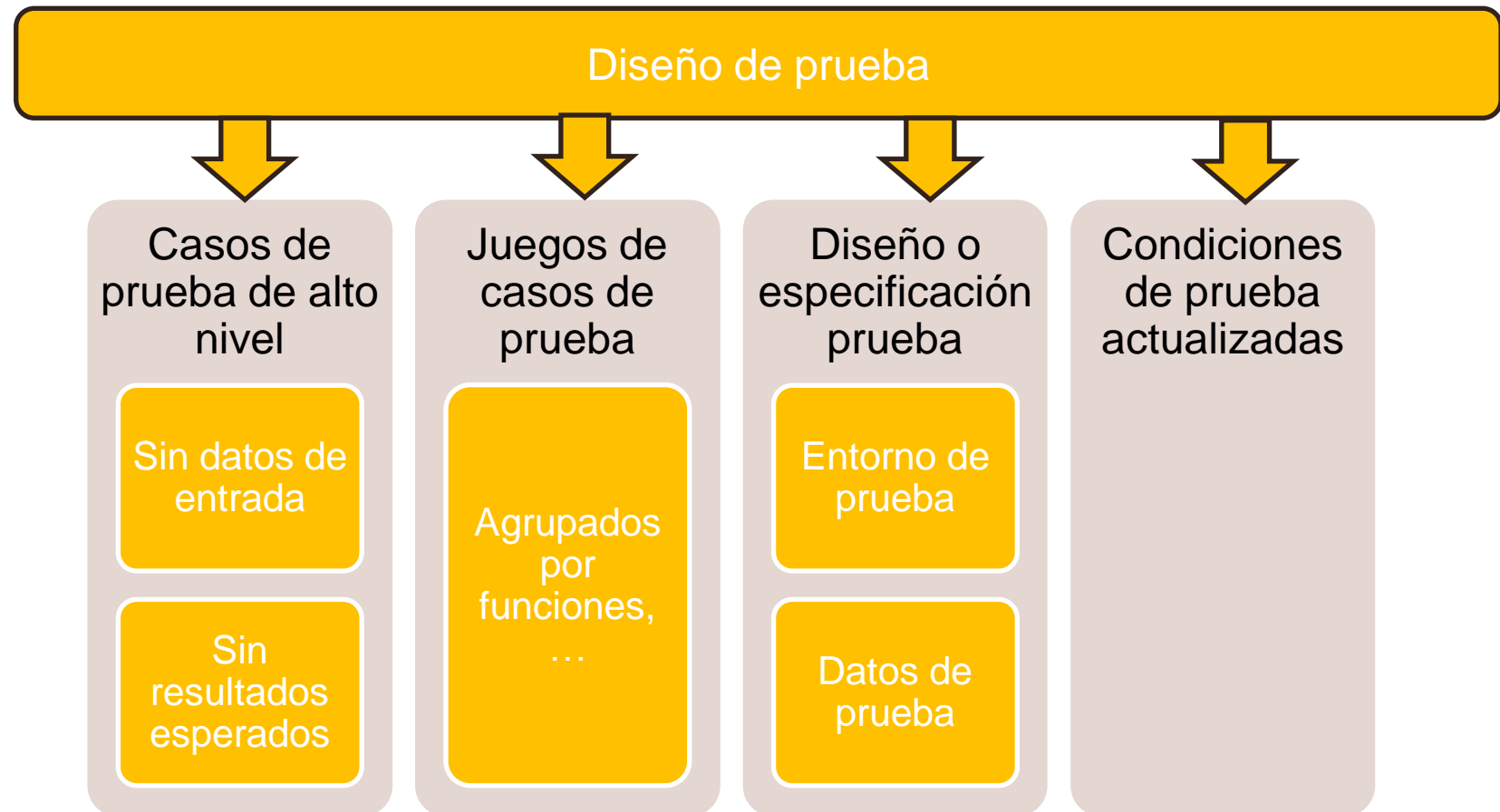
1.4.3 Entregables de prueba

Análisis de prueba de productos de trabajo



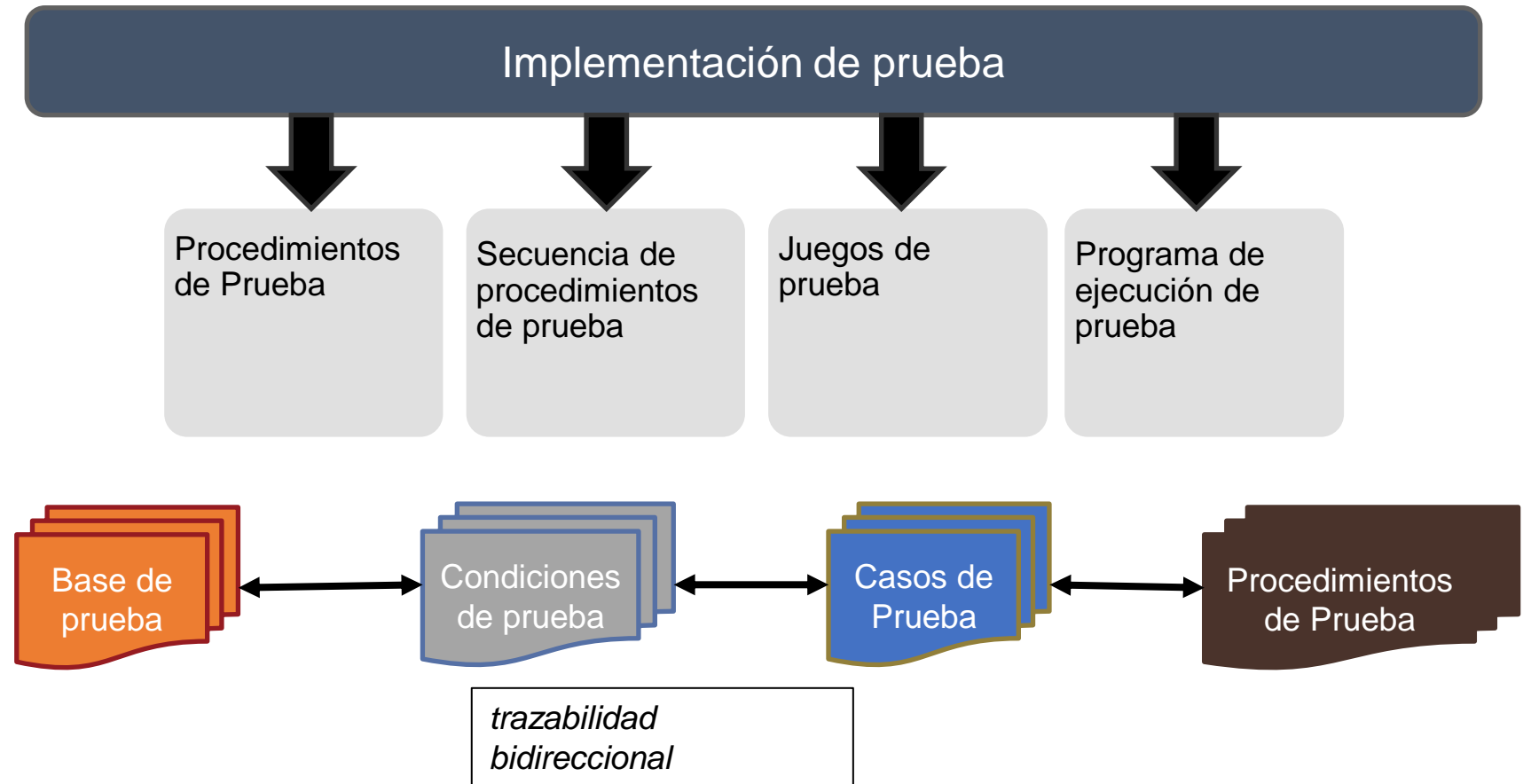
1.4.3 Entregables de prueba

Diseño de prueba
de productos de
trabajo



1.4.3 Entregables de prueba

Implementación
de prueba de
productos de
trabajo



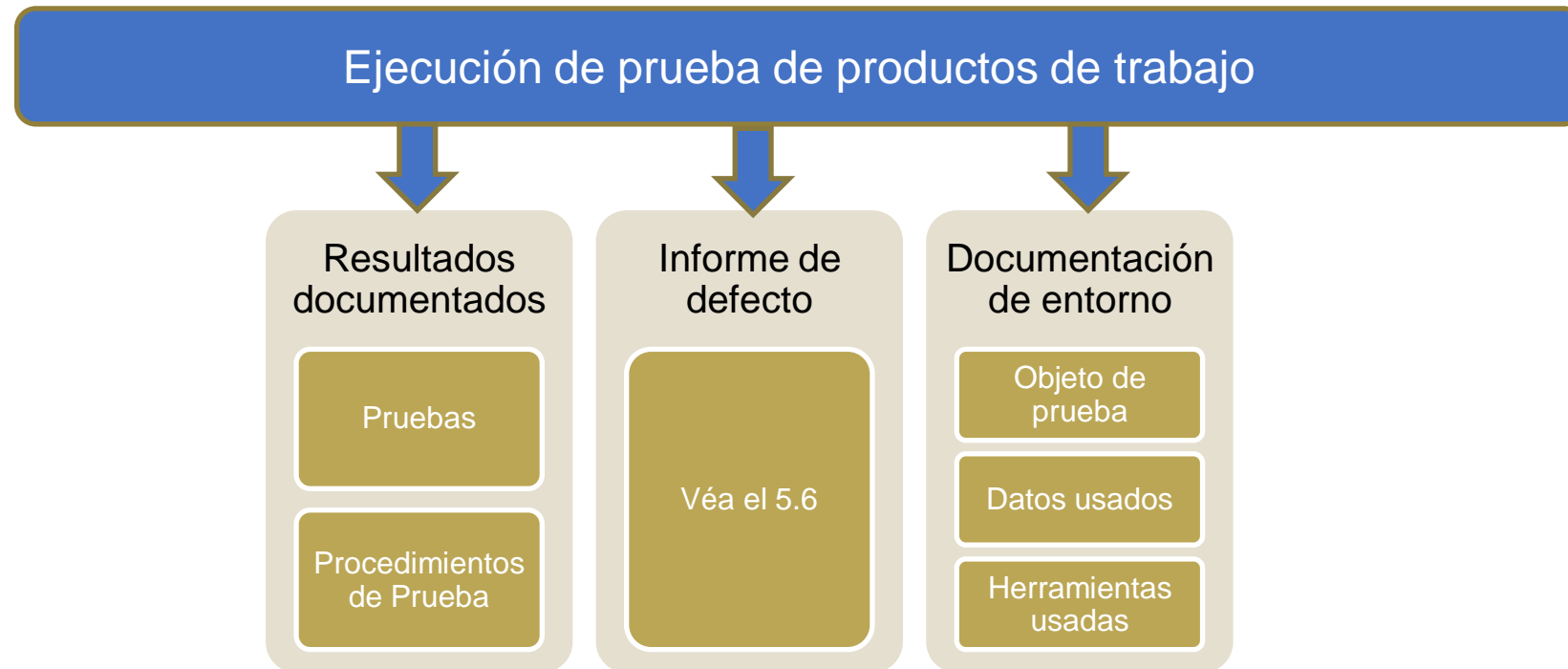
1.4.3 Entregables de prueba

Implementación de prueba de productos de trabajo

Details	Design Steps *	Test Script	Attachments	Req Coverage	Linked Defects
	Step Name	Description	Expected		
	Step 1	Launch Internet Explorer. Type www.orange.fr and click Go	The main page of Orange Web site is displayed. Three links are available : Mobile, Fixe and Internet.		
	Step 2	Click on the link <u>Mobile</u> .	A new page is displayed. Several pictures are displayed, showing mobile phones.		

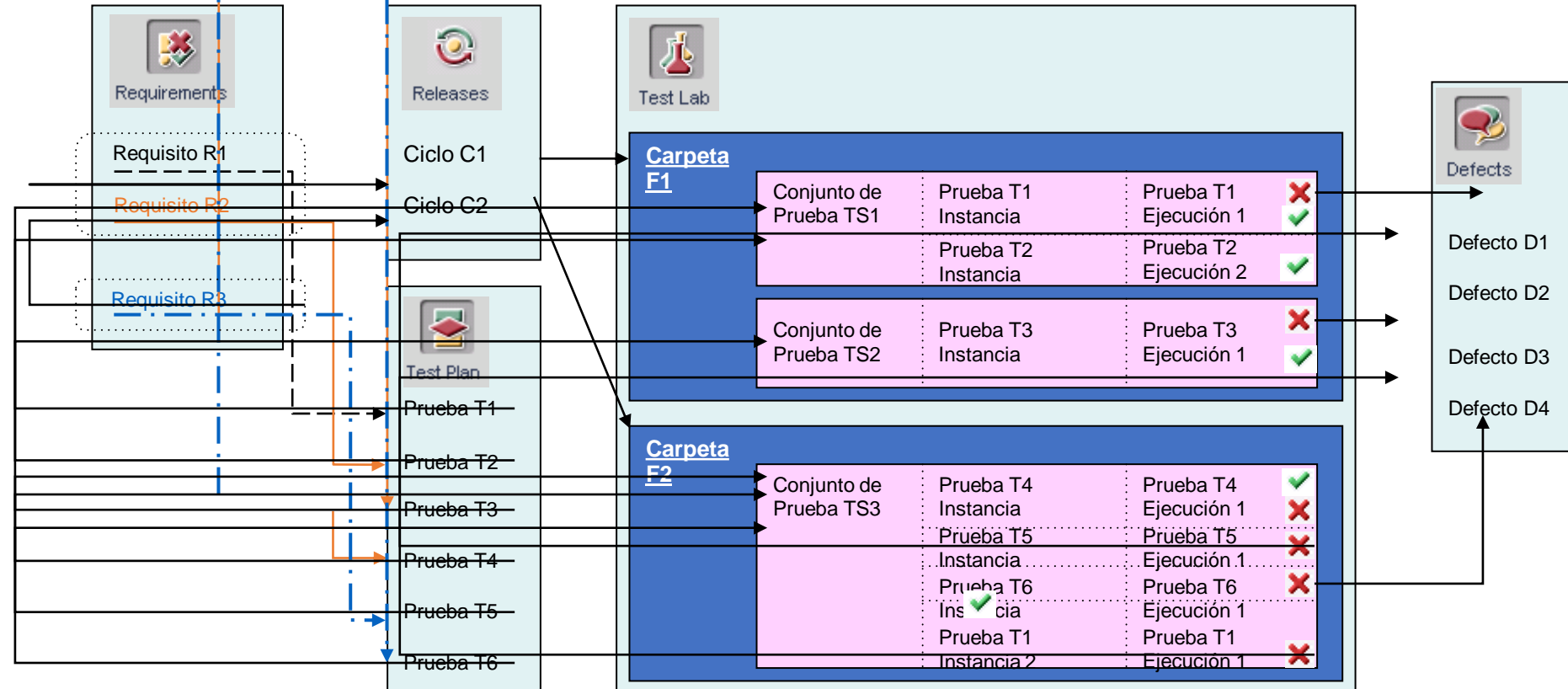
Ejemplo de una herramienta de gestión de pruebas

1.4.3 Entregables de prueba



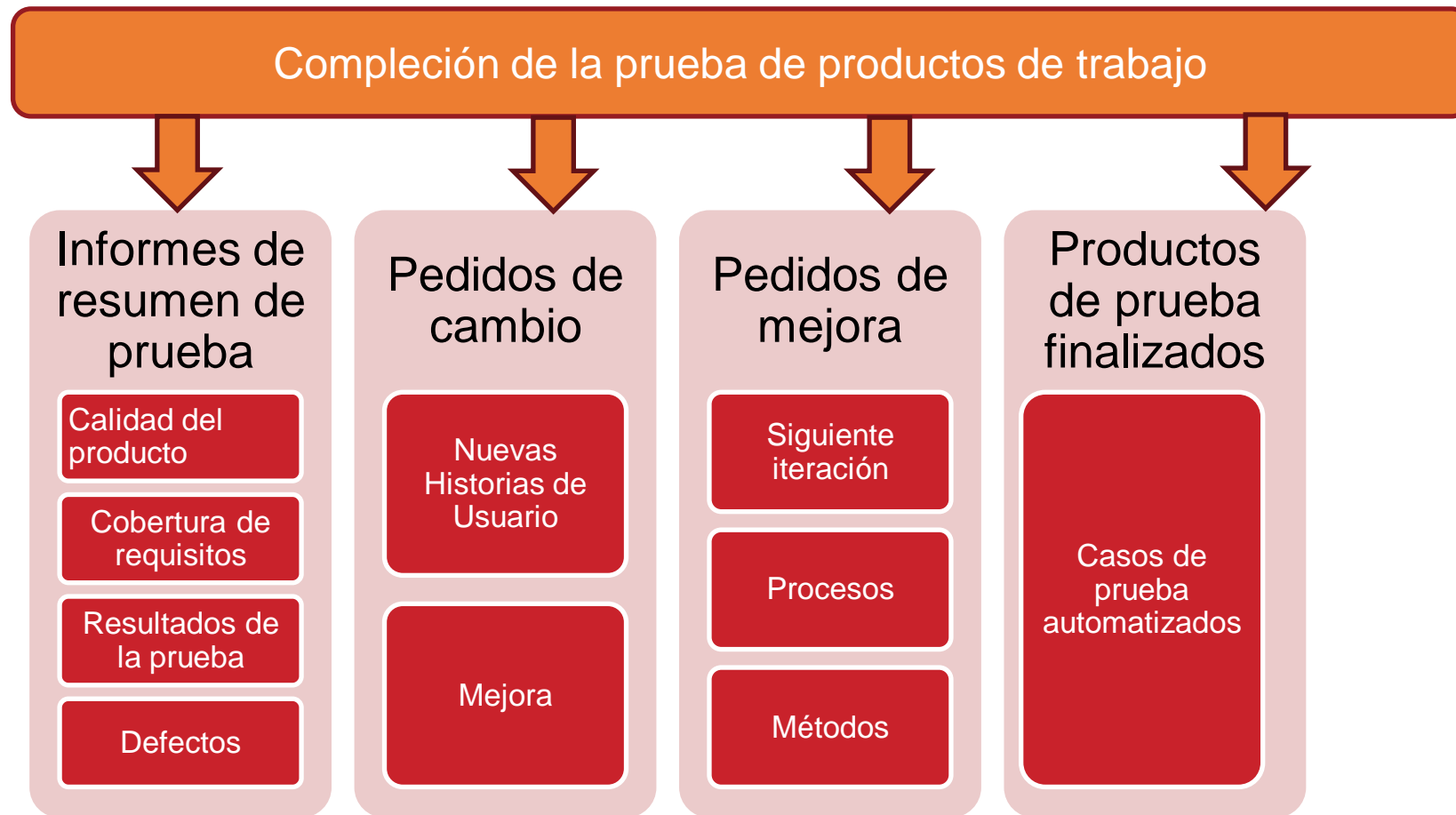
1.4.3 Entregables de prueba


- Ejecución de prueba de productos de trabajo



Ejemplo: el estado de cobertura de requisitos se actualiza durante la ejecución de prueba con una herramienta como HP ALM

1.4.3 Entregables de prueba





1.4.4 Trazabilidad entre las bases de prueba y los productos de trabajo de prueba

La monitorización de pruebas se basa en la trazabilidad, para apoyar:

Analizar el impacto de los cambios.

Hacer que la prueba sea auditable

Cumplir con los criterios de dirección de TI

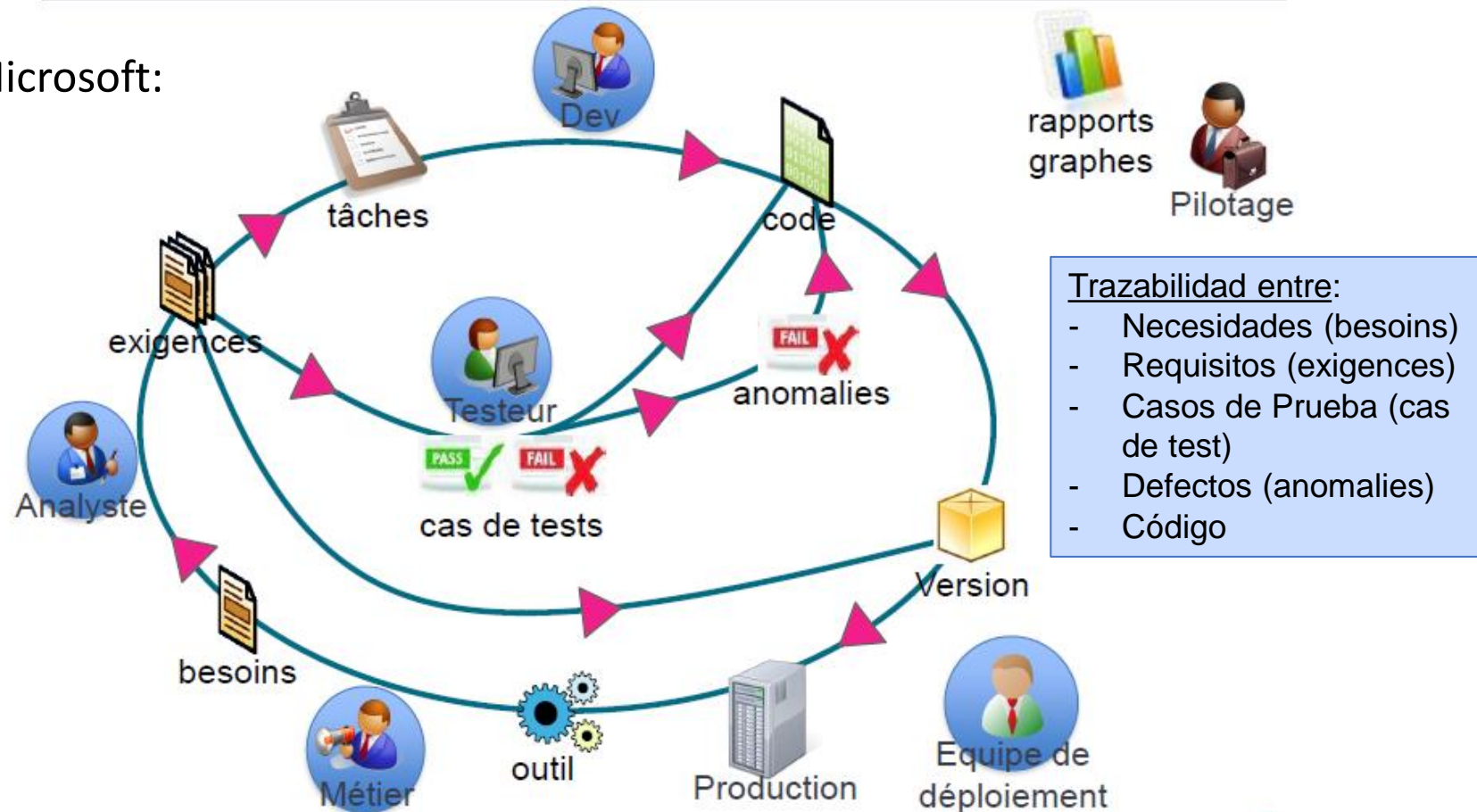
Mejorar la comprensión de los informes de avance de la prueba y los informes resumen de prueba

Describirle a las partes interesadas los aspectos técnicos de las pruebas en términos que puedan entender

Proporcionar información para evaluar la calidad del producto, la capacidad del proceso y el avance del proyecto en relación con los objetivos comerciales

1.4.4 Trazabilidad entre las bases de prueba y los productos de trabajo de prueba

•Ejemplo con Microsoft:



Capítulo 1

- 1.1 ¿Qué es poner a prueba?
- 1.2 ¿Por qué son necesarias las pruebas
- 1.3 Siete Principios de las Pruebas
- 1.4 Proceso de Prueba
- 1.5 La Psicología de las Pruebas





1.5.1 La Psicología Humana y las Pruebas

Las pruebas deben tener en cuenta algunos aspectos psicológicos

Sesgo de confirmación

- Definición: dificultad para aceptar información que no esté de acuerdo con las creencias actuales
- Ejemplo: como los desarrolladores esperan que su código sea correcto, tienen un sesgo de confirmación que dificulta la aceptación de que el código sea incorrecto



1.5.1 La Psicología Humana y las Pruebas

La psicología humana tiene efectos importantes sobre las pruebas de software

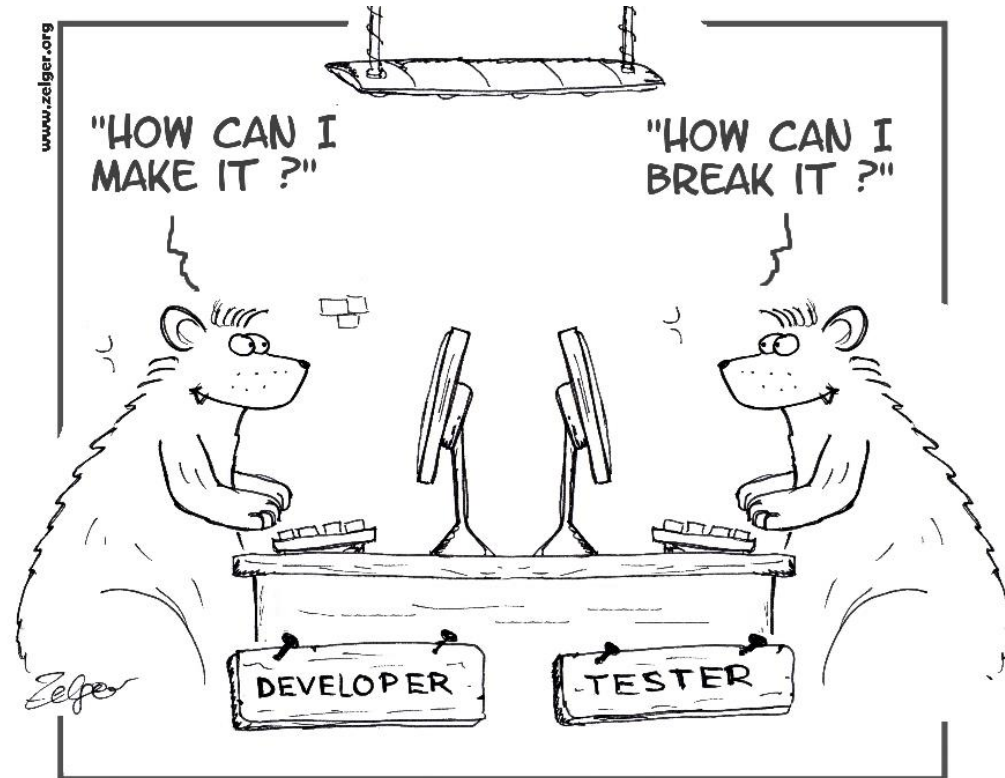
Las habilidades interpersonales son esenciales

- Comience con colaboración en lugar de batallas
- Haga incapié sobre los beneficios de las pruebas
- Comunique los resultados de la prueba y otros hallazgos de una manera neutral y centrada en los hechos, sin criticar a la persona que creó el elemento defectuoso
- Redacte informes de defectos objetivos y fácticos, y revise los hallazgos
- Trate de comprender cómo se sienten las otras personas y las razones por las cuales reaccionan negativamente ante la información
- Confirme que la otra persona ha entendido lo que le ha dicho y viceversa

1.5.2 Los modos de pensar de los probadores y desarrolladores

Los probadores y desarrolladores tienen objetivos diferentes

diseñar y construir un producto



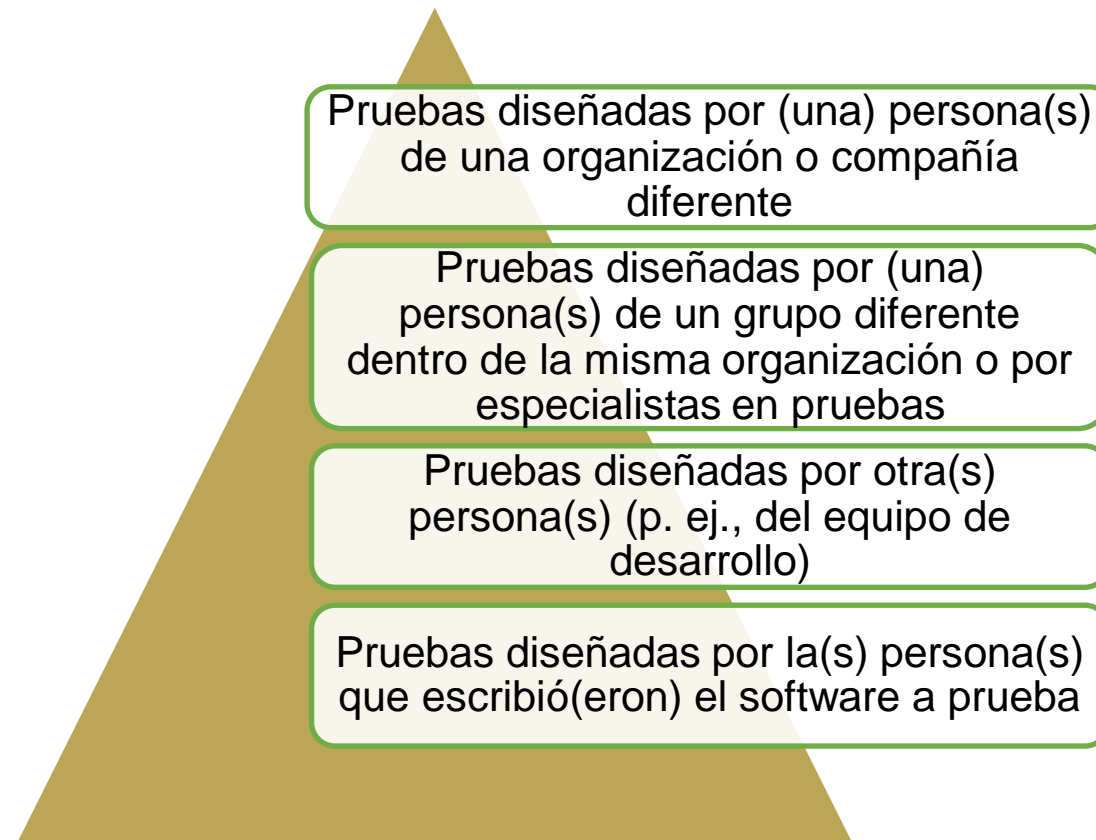
encontrar defectos

They weren't so much different, but they had different goals

1.5.2 Los modos de pensar de los probadores y desarrolladores

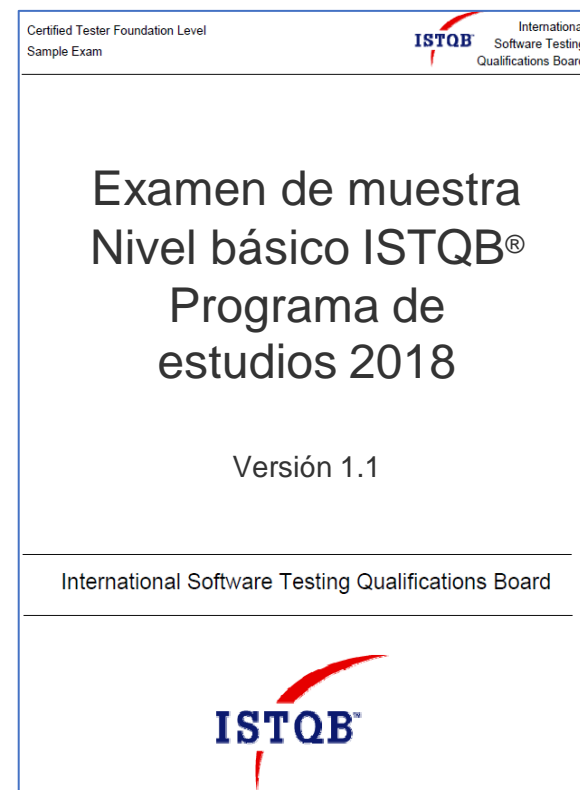
•Ejemplo: Grado de independencia (de bajo a alto nivel)

- Un desarrollador puede probar su propio código, con un modo de pensar adecuado, pero a menudo es una mala práctica



Ejercicio: ISTQB Examen de Muestra A

- Preguntas de la
1 a la 8



Capítulo 1

Fundamentos de las pruebas

