



Curso de QA & Software Testing

Parte II



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



Capacitador

Ronnel Vélez Manzano

Gerente de QA & SW Testing

Miembro del HASTQB



Certificaciones: SCRUM MASTER

ISTQB | CTFL | Certified Tester Foundation Level

ISTQB | CTAL-TM | Certified Tester Advanced Level - Test Manager

ISTQB | CTFL- AT | Certified Agile Tester

E-mails: rvelez@crnova.com

rvelez@hastqb.org

ronnel.velez@gmail.com

Servicios Computacionales Novacomp

San Jose – Costa Rica

Web: <http://www.crnova.com>

Teléfono (Costa Rica): (506) 2216-5800

Fax: (506) 2216-5900



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



Instrucciones

El presente material de formación ha sido elaborado por Servicios Computacionales Novacomp. Está basado en el programa de estudios de:

- IREB: International Requirements Engineering Board
- ISTQB: International Software Testing Qualifications Board
- Manifesto for Agile Software Development
- Scrum Alliance

Las marcas contenidas e incorporadas en los documentos son propiedad de sus propietarios respectivos incluso si no son mencionados de forma explícita.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



Pruebas a través del ciclo de vida software

- Modelos de desarrollo software
- Niveles de prueba
- Tipos de pruebas:
 - Pruebas funcionales
 - Pruebas no funcionales
 - Pruebas estructurales
 - Pruebas asociadas al cambio.
- Otros tipos de pruebas.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



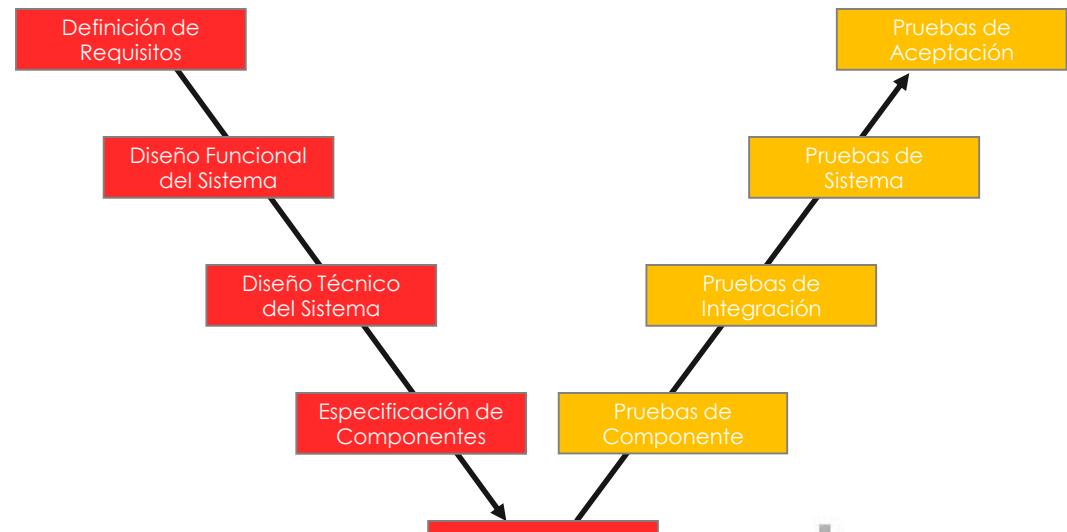
ORACLE Gold Partner

III. Pruebas a través del ciclo de vida software

01. Modelos de desarrollo software

Pruebas a través del modelo-V general (Modelo de desarrollo secuencial)

- El modelo-V general es el modelo de desarrollo software **más utilizado**
- Desarrollo y pruebas son **dos ramas iguales**
 - Cada nivel de desarrollo tiene su correspondiente nivel de pruebas
- Las pruebas (rama derecha) son diseñadas en paralelo al desarrollo software (rama izquierda).
- Las actividades del proceso de pruebas tienen lugar a lo largo del ciclo de vida software completo



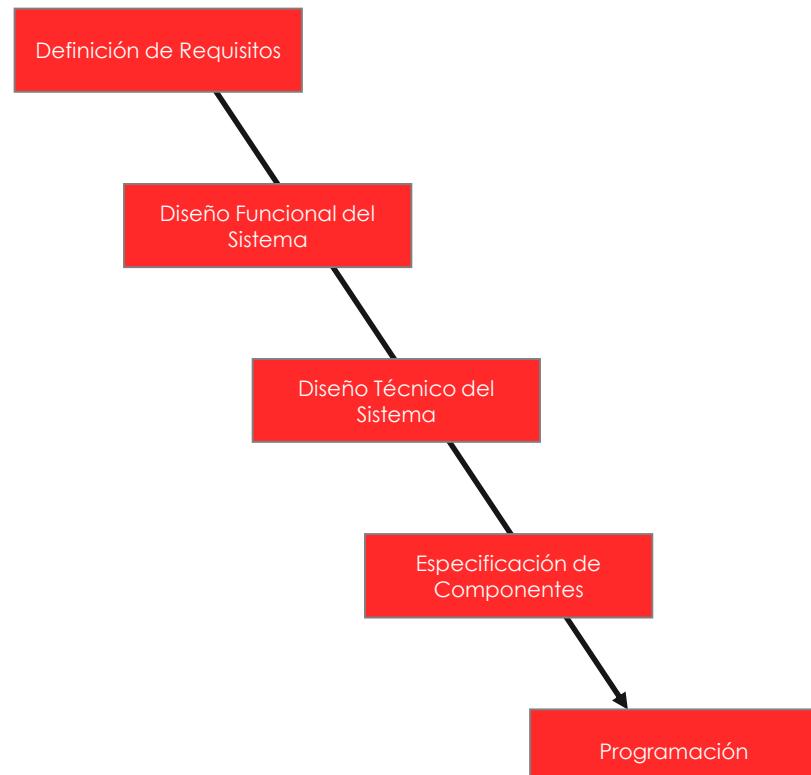
III. Pruebas a través del ciclo de vida software

01. Modelos de desarrollo software

Pruebas a través del modelo-V general

Rama de desarrollo software

- Definición de requisitos
 - Documentos de especificación
- Diseño funcional del sistema
 - Diseño del flujo funcional del programa
- Diseño técnico del sistema
 - Definición de arquitectura/interfaces
- Especificación de los componentes
 - Estructura de los componentes
- Programación
 - Creación de código ejecutable



Microsoft Partner

Gold Cloud Platform
 Silver Cloud Productivity
 Silver Collaboration and Content
 Silver Application Development
 Silver Small Business

ISTQB
 International Software Testing Qualifications Board
 Silver Partner

ORACLE
 Gold Partner

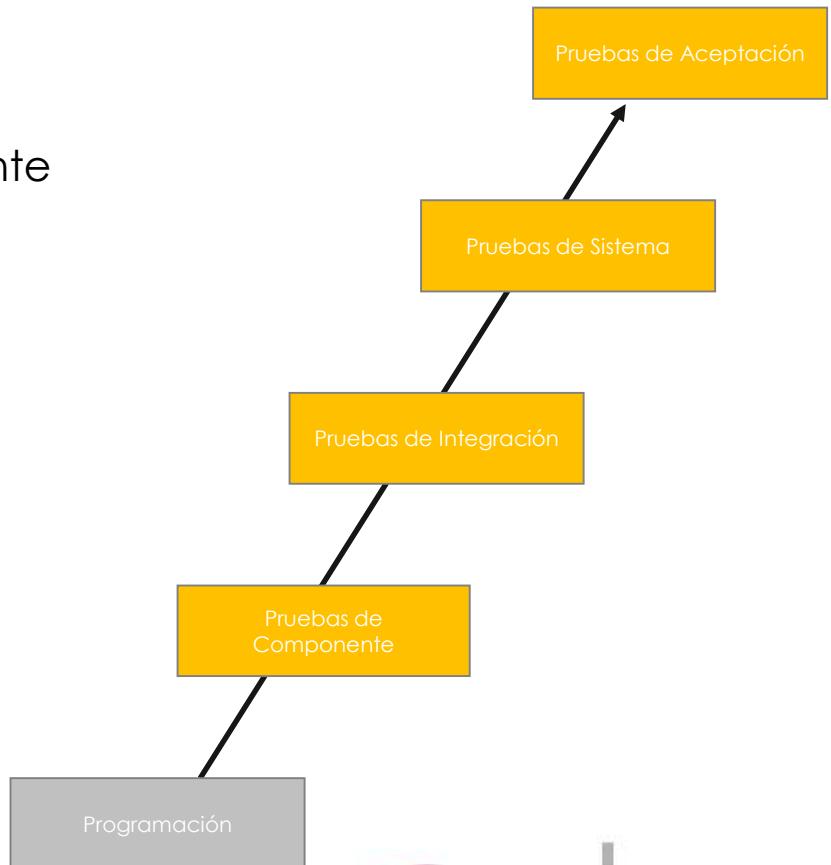
III. Pruebas a través del ciclo de vida software

01. Modelos de desarrollo software

Pruebas a través del modelo-V general

Rama de pruebas software

- Pruebas de aceptación
 - Pruebas formales de los requisitos del cliente
- Pruebas de sistema
 - Sistema integrado, especificaciones
- Pruebas de integración
 - Interfaces de componentes
- Pruebas de componente
 - Funcionalidad del componente



III. Pruebas a través del ciclo de vida software

01. Modelos de desarrollo software

Verificación vs. Validación

Verificación

- Comprobación de la conformidad con los requisitos establecidos (definición según ISO 9000)
- Cuestión clave: ¿Se ha procedido correctamente en la construcción del sistema?
- ¿Hemos sumado 1 más 1 correctamente?

Validación

- Comprobación de la idoneidad para el uso esperado (definición según ISO 9000)
- Cuestión clave: ¿Hemos construido el sistema software correcto?
- ¿El objetivo era sumar 1 más 1 o deberíamos haber restado?



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



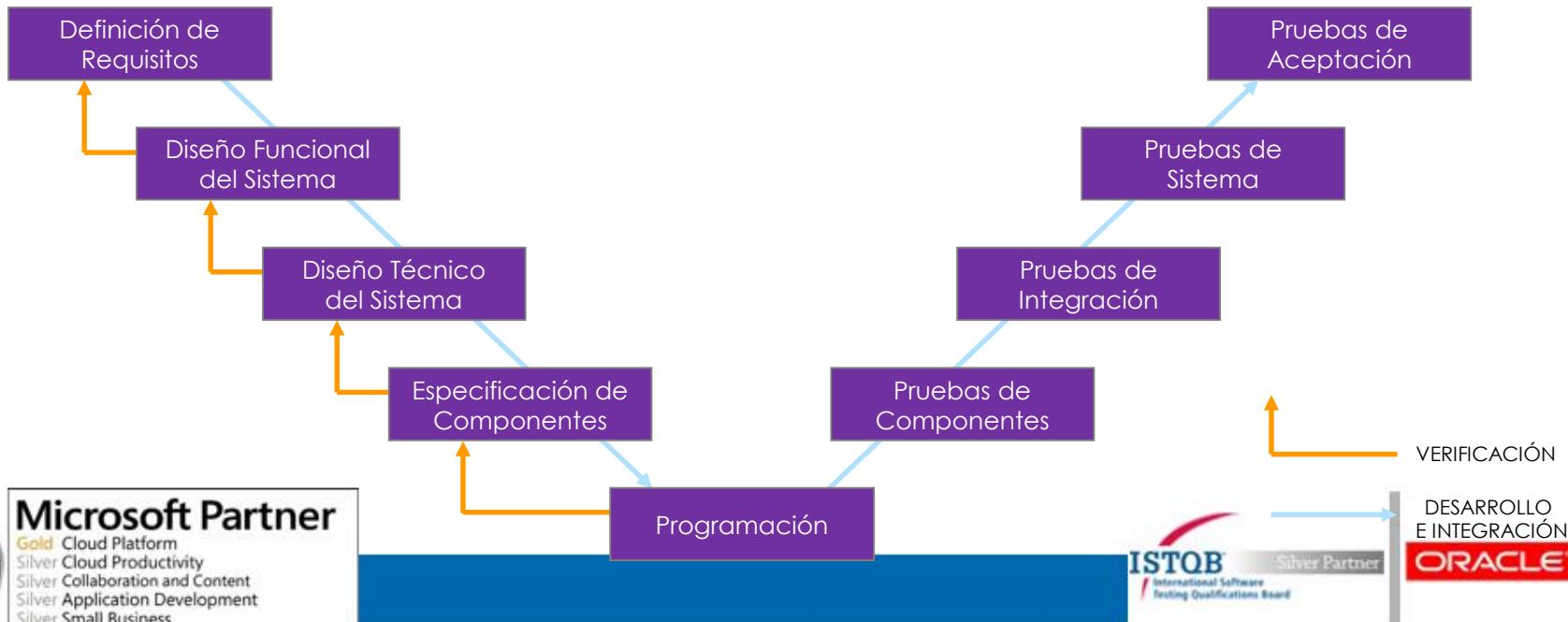
ORACLE Gold Partner

III. Pruebas a través del ciclo de vida software

01. Modelos de desarrollo software

Verificación dentro del modelo-V general

- Cada nivel de desarrollo se verifica respecto de los contenidos del nivel que le precede
 - Verificar: comprobar la evidencia, substanciar
- Verificar significa comprobar si los requisitos y definiciones de niveles previos han sido implementados correctamente

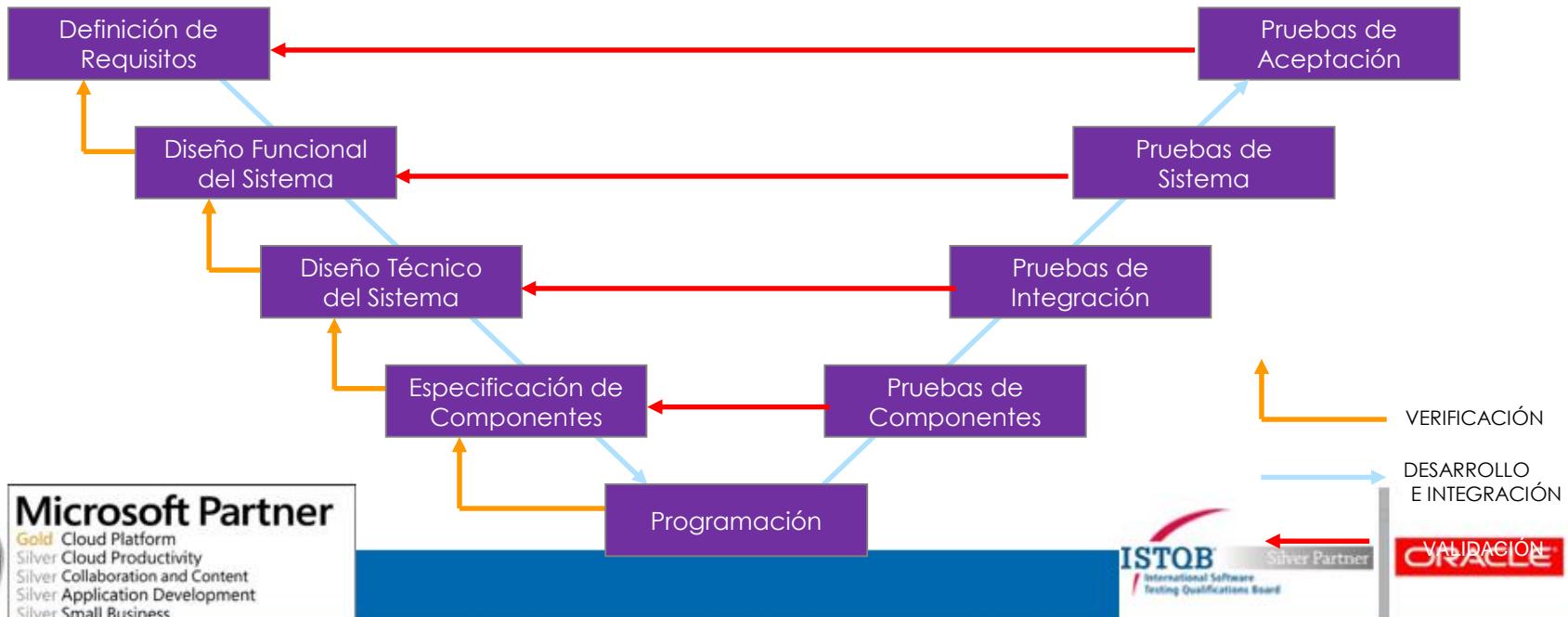


III. Pruebas a través del ciclo de vida software

01. Modelos de desarrollo software

Validación dentro del modelo-V general

- La validación se refiere a la corrección de cada nivel de desarrollo
 - Validar: dar prueba de la aportación de valor
- Validar significa comprobar lo adecuado de los resultados de un nivel de desarrollo

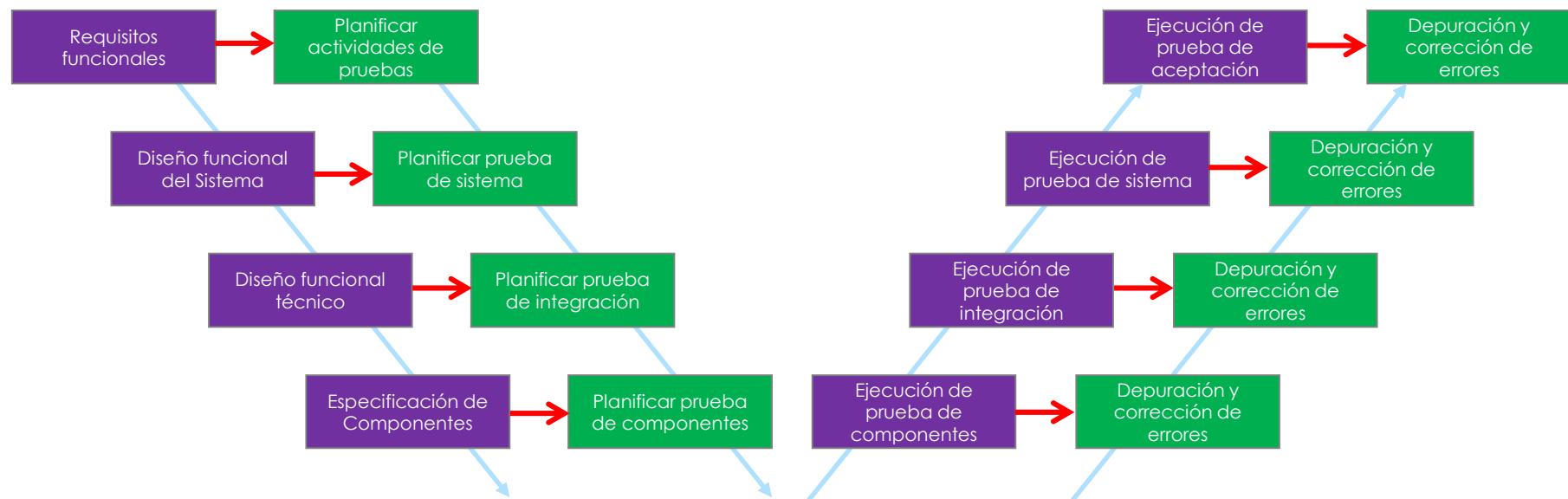


III. Pruebas a través del ciclo de vida software

01. Modelos de desarrollo software

Pruebas en el modelo-W

- El modelo-W puede ser visto como una extensión del modelo-V general
- El modelo-W pone de manifiesto de forma clara que ciertas actividades del aseguramiento de la calidad se desarrollarán en paralelo con el proceso de desarrollo



III. Pruebas a través del ciclo de vida software

01. Modelos de desarrollo software

Modelos iterativos: tipos de modelos iterativos (1)

Desarrollo software iterativo:

- Las actividades: definición de requisitos, diseño, desarrollo y pruebas se segmentan en pasos reducidos y se ejecutan de forma continua
- Se debe alcanzar el consentimiento con el cliente tras cada iteración con el objeto de poder modificar el rumbo del proyecto si fuera necesario

Ejemplo de modelos iterativos más conocidos:

Modelo prototipado: desarrollo rápido de una representación del sistema que pudiera ser objeto de uso, seguida de modificaciones sucesivas hasta que el sistema sea finalizado

Desarrollo rápido de aplicaciones (“Rapid Application Development” - (RAD)): La interfaz de usuario se implementa utilizando una funcionalidad previamente construida (“out-of-the-box ”) simulando la funcionalidad que posteriormente será desarrollada

Proceso unificado (“Rational Unified Process” - (RUP)): modelo orientado a objeto y producto de la compañía Rational/IBM. Principalmente aporta el lenguaje de modelado UML y soporte al Proceso Unificado

Programación extrema (“Extreme Programming” - (XP)): el desarrollo y las pruebas tienen lugar sin una especificación de requisitos formalizada



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



III. Pruebas a través del ciclo de vida software

01. Modelos de desarrollo software

Ejemplos de modelos iterativos agiles más conocido:

SCRUM – descripción del proceso



Microsoft Partner

Gold Cloud Platform
 Silver Cloud Productivity
 Silver Collaboration and Content
 Silver Application Development
 Silver Small Business

ISTQB
 International Software Testing Qualifications Board

Silver Partner

ORACLE Gold Partner

III. Pruebas a través del ciclo de vida software

01. Modelos de desarrollo software

Modelos iterativos: Características

- Características de los modelos iterativos:
- Cada iteración contribuye con una **característica adicional** del sistema a desarrollar
- Cada iteración puede ser **probada por separado**
- Las **pruebas de regresión** y la **automatización de pruebas** son elementos/factores de gran relevancia
- En cada iteración, la verificación (relación con el nivel precedente) y la validación (grado de corrección del producto dentro del nivel actual) se pueden efectuar por separado



III. Pruebas a través del ciclo de vida software

01. Modelos de desarrollo software

Modelos iterativos: desarrollo guiado por pruebas (“Test driven development”)

- Basado en: juegos de casos de prueba
 - Preparación de **ciclos de prueba (“Test cycle”)**
 - Pruebas **automatizadas** con el uso de herramientas de prueba
- Desarrollo de acuerdo a casos de prueba
 - Preparación temprana de versiones de **componentes** para su **prueba**
 - **Ejecución automática** de pruebas
 - **Corrección de defectos** para versiones futuras
 - **Repetición** de juegos de prueba hasta que no se detecten defectos

Primero se diseñan las **pruebas**, a continuación se **codifica (programa)** el software



III. Pruebas a través del ciclo de vida software

01. Modelos de desarrollo software

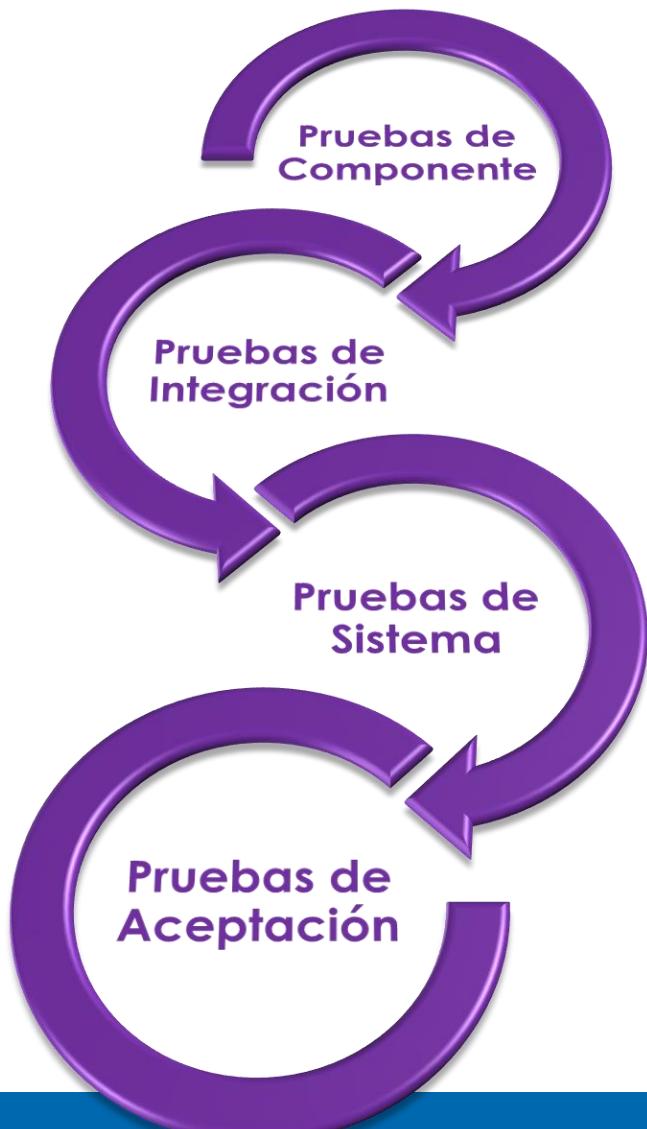
Principios de todos los modelos

- **Cada actividad** de desarrollo debe ser **probada**
 - Ninguna porción del software puede quedar sin ser probada, si ha sido desarrollada tanto de forma secuencial ("en un procedimiento") o iterativa
- Cada **nivel de prueba** debería ser probado de forma específica
 - Cada nivel de prueba cuenta con objetivos de prueba propios
 - Las pruebas llevadas a cabo en cada nivel deben reflejar estos objetivos
- El **proceso de prueba comienza** mucho **antes** de la **ejecución de las pruebas**
 - Tan pronto como el desarrollo comienza puede comenzar la preparación de las pruebas correspondientes
 - También es el caso de las revisiones de documentos comenzando por los conceptos, especificación y el diseño global (en conjunto)



III. Pruebas a través del ciclo de vida software

02. Niveles de Pruebas



Microsoft Partner

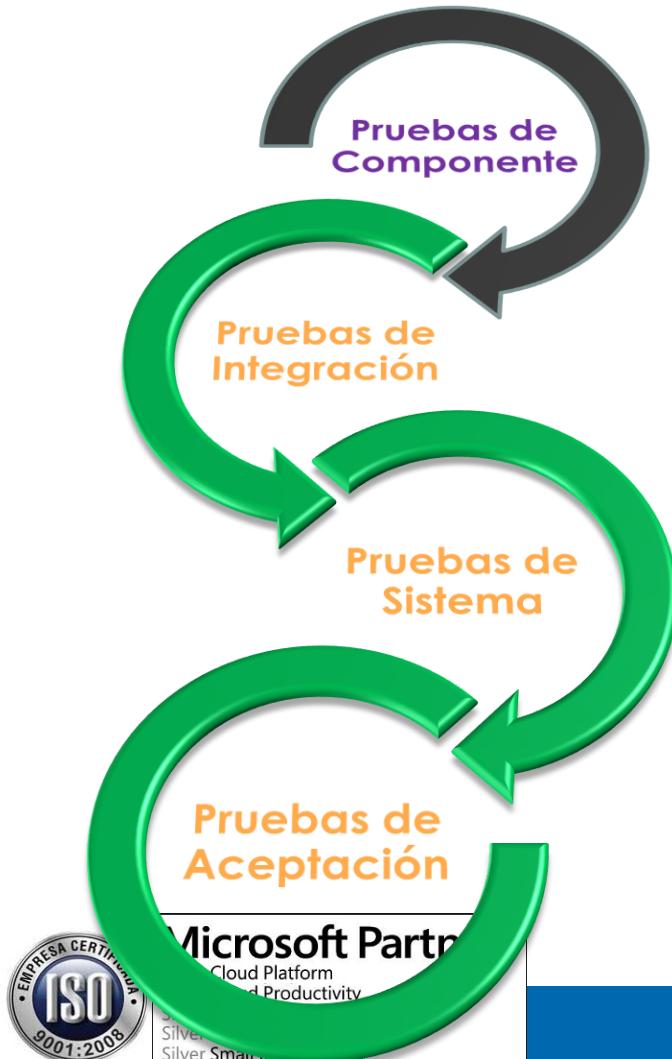
Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business

 ISTQB
International Software
Testing Qualifications Board
Silver Partner

 ORACLE
Gold
Partner

III. Pruebas a través del ciclo de vida software

02. Niveles de Pruebas



Pruebas de Componente

Pruebas de componentes software individuales

Bases de prueba

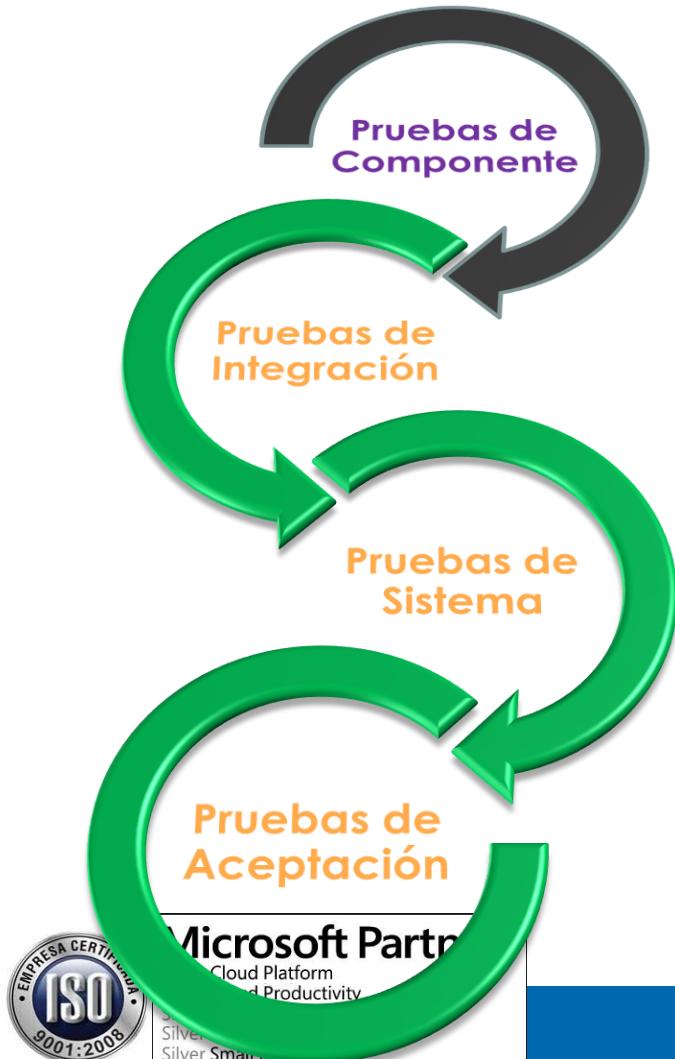
- Requisitos de componente
- Diseño detallado
- Código

Objetos de prueba típicos

- Componentes / clases / unidades / módulos
- Programas
- Conversión de datos / migración de programas

III. Pruebas a través del ciclo de vida software

02. Niveles de Pruebas

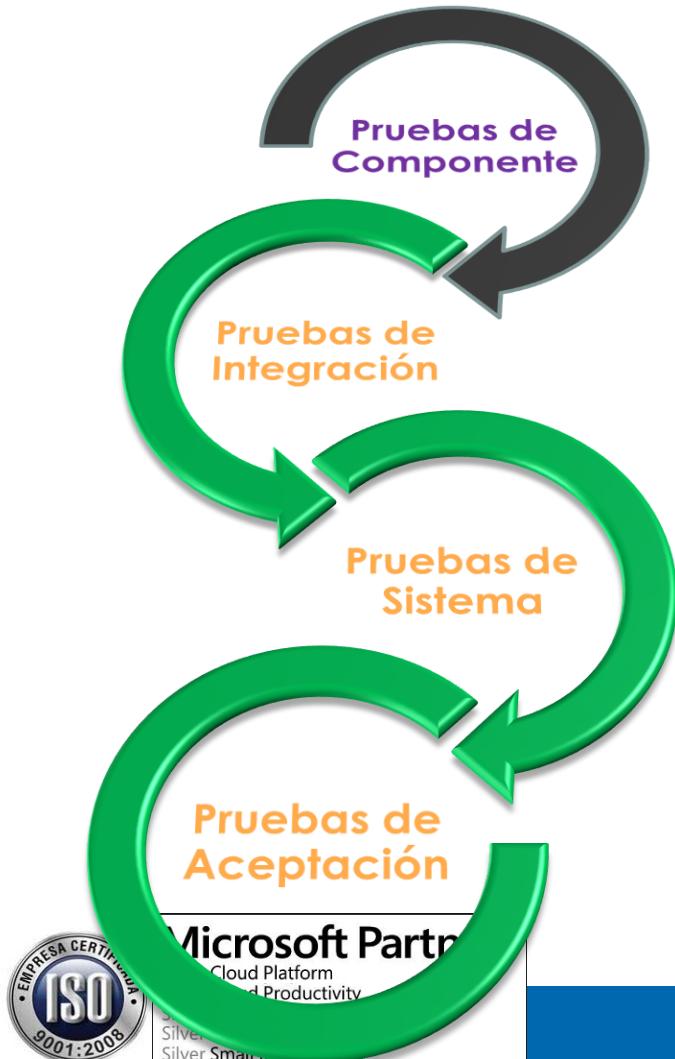


Pruebas de Componente:

- Prueba de componente (pruebas unitarias)
 - Prueba de cada componente tras su realización/construcción
- Dadas las convenciones de cada lenguaje de programación para la asignación de nombres a sus respectivos componentes, se podrá hacer referencia a un componente como:
 - **Prueba de módulo (“module test”)** (por ejemplo en C)
 - **Prueba de clase (“class test”)** (por ejemplo en Java o C++)
 - **Prueba de unidad (“unit test”)** (por ejemplo en Pascal)
- Los componentes son referidos como módulos, clases o unidades. Dado que los desarrolladores posiblemente pudieran estar involucrados en la ejecución de pruebas, éstas también son denominadas pruebas de desarrollador (developer's test)

III. Pruebas a través del ciclo de vida software

02. Niveles de Pruebas



Pruebas de Componente: Alcance

Sólo se prueban **componentes individuales**

- Un componente puede estar constituido por un conjunto de unidades más pequeñas
- Los objetos de prueba no siempre pueden ser probados en solitario.

Cada componente es probado de **forma independiente**

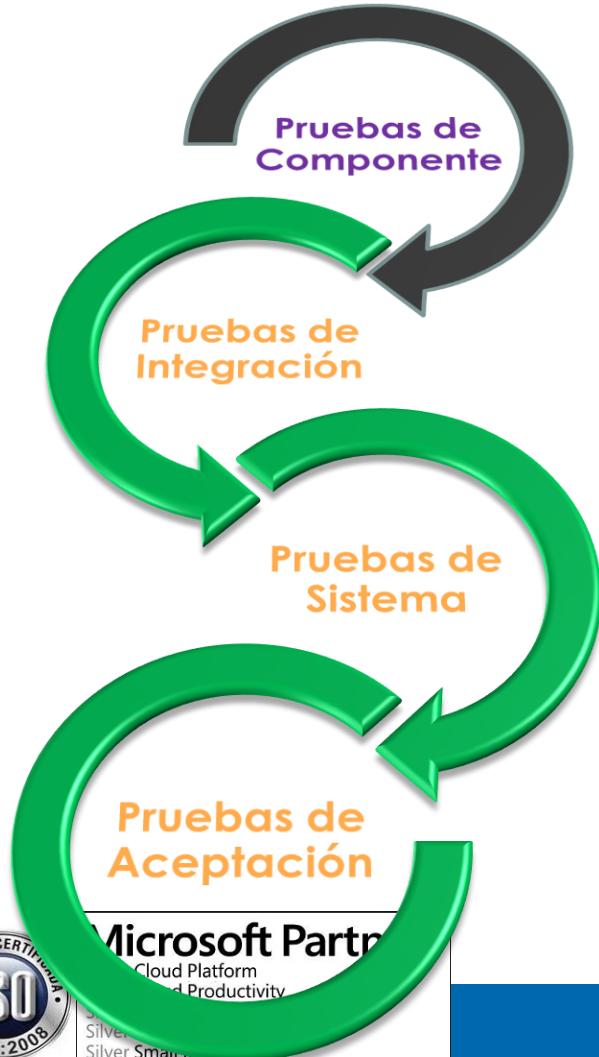
- Descubriendo fallos producidos por defectos internos
- Los efectos cruzados entre componentes quedan fuera del alcance de estas pruebas

Los **casos de prueba** podrán ser obtenidos a partir de:

- Especificaciones de componente
- Diseño software
- Modelo de datos

III. Pruebas a través del ciclo de vida software

02. Niveles de Pruebas



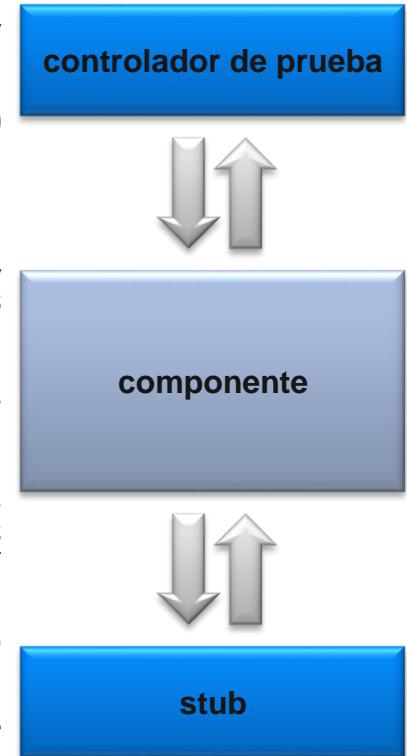
Pruebas de componente: Arnés de pruebas

La ejecución de pruebas de componente requiere de ("drivers") y stubs

- ▶ **Un controlador ("driver")** gestiona/trata la interfaz de un componente
 - ▶ Los controladores ("drivers") simulan datos de entrada, registran datos de salida y aportan un arnés de pruebas ("test harness")
 - ▶ Los controladores ("drivers") utilizan herramientas de programación
- ▶ Un **stub** reemplaza o simula un componente que aún no se encuentra disponible o que no es parte del objeto de prueba ("test object")

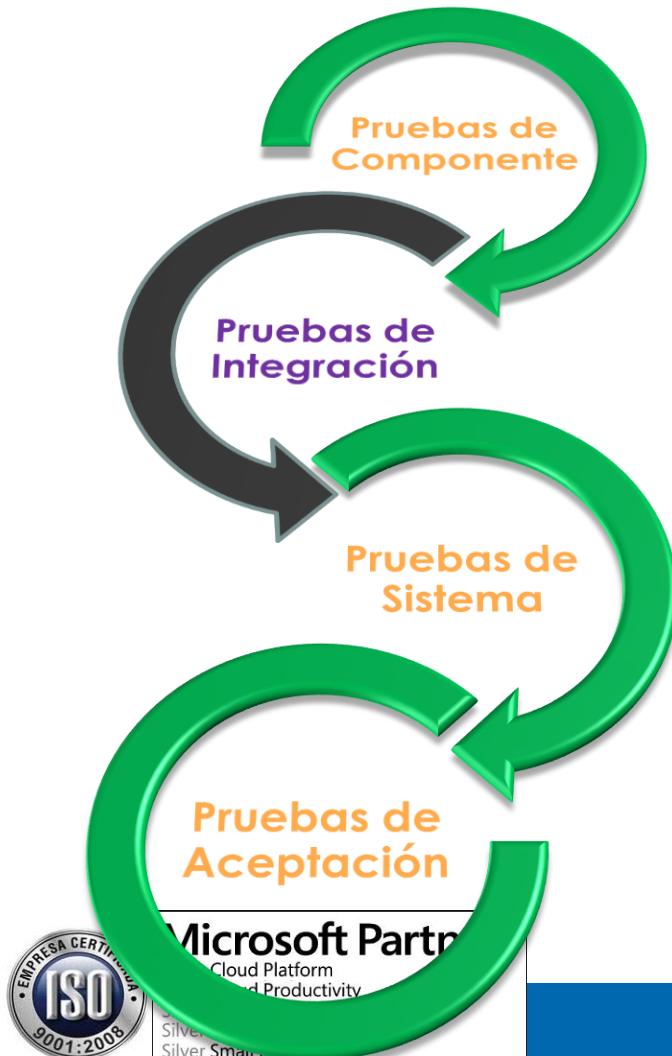
Para **programar controladores y/o stubs**:

- ▶ Se debe contar con **conocimientos de programación**
- ▶ Se necesita disponer del código fuente
- ▶ Podrán ser necesarias herramientas especiales



III. Pruebas a través del ciclo de vida software

02. Niveles de Pruebas



Pruebas de Integración

Pruebas realizadas con el objeto de poner en evidencia defectos en las interfaces e interacciones entre componentes o sistemas integrados.

Bases de prueba

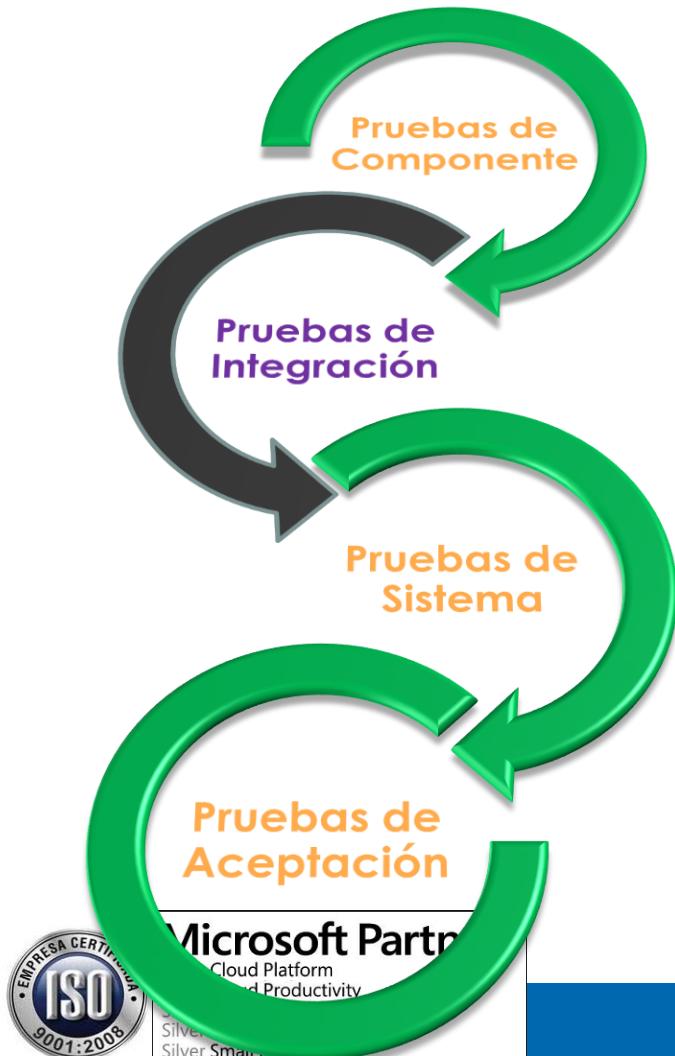
- ▶ Diseño software y del sistema
- ▶ Arquitectura
- ▶ Flujos de trabajo (“workflows”)
- ▶ Casos de uso

Objetos de prueba típicos

- ▶ Implementación de subsistema de base de datos
- ▶ Infraestructura
- ▶ Interfaces

III. Pruebas a través del ciclo de vida software

02. Niveles de Pruebas



Pruebas de Integración

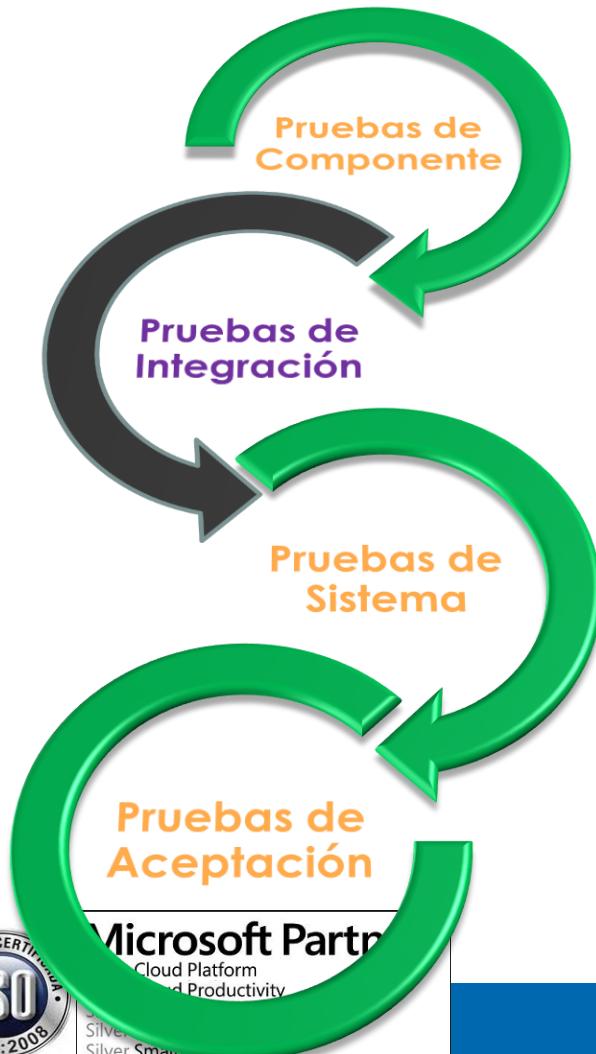
Cada componente ya ha sido probado en lo referente a su funcionalidad interna (prueba de componente). Las pruebas de integración comprueban la interacción entre elementos software (componentes) entre distintos sistemas o hardware y software (normalmente tras las pruebas de sistema)

La integración es la actividad en la cual se combinan componentes software individuales en subsistemas más amplios o en una serie de sistemas

Puede ser ejecutadas por desarrolladores, probadores o ambos

III. Pruebas a través del ciclo de vida software

02. Niveles de Pruebas



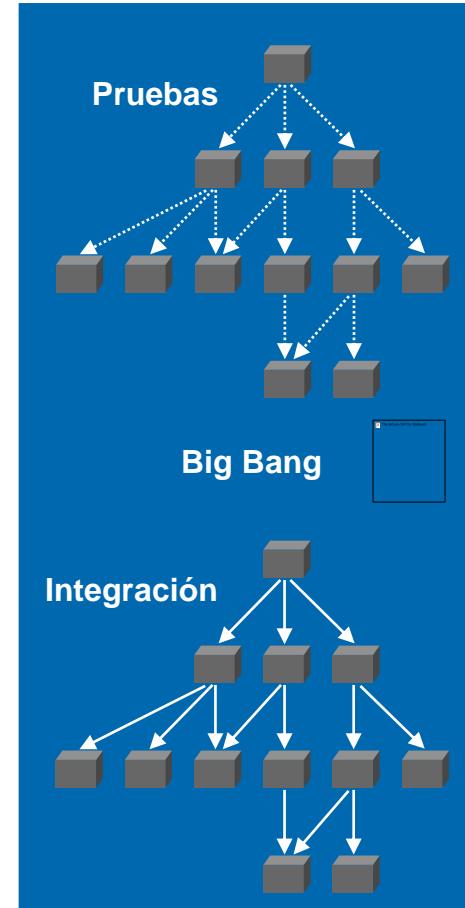
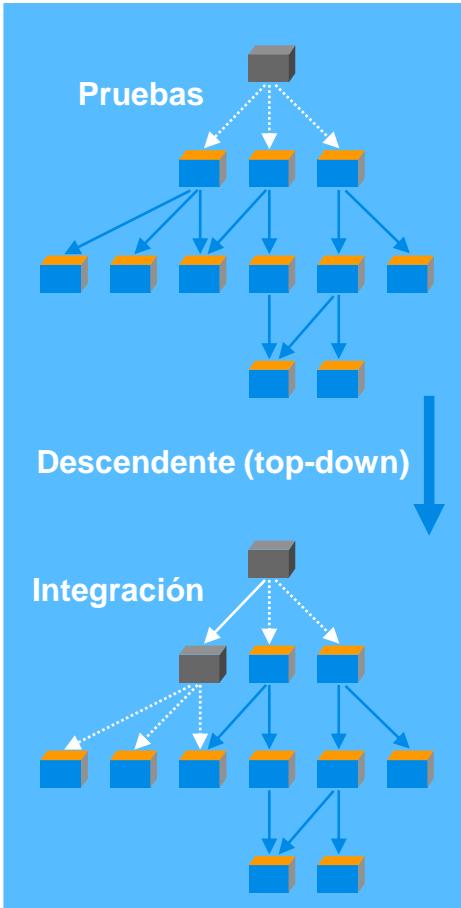
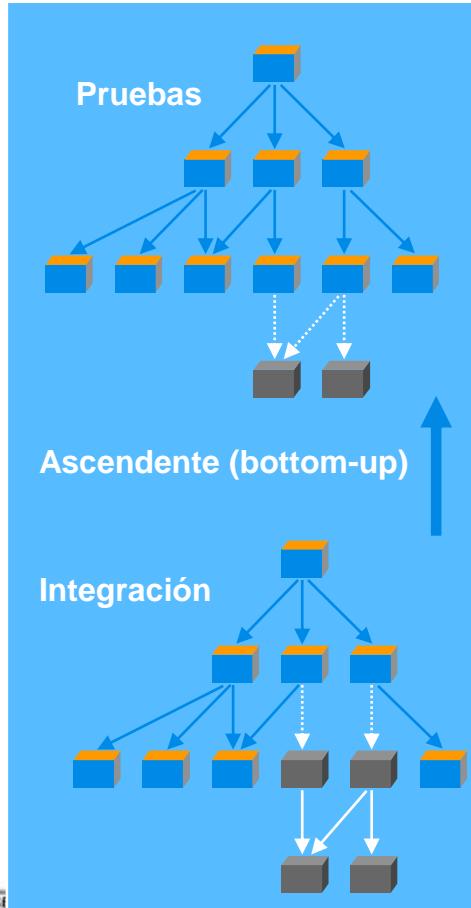
Pruebas de Integración: Alcance

- ▶ Las pruebas de integración asumen que los componentes **ya han sido probados**
- ▶ Las pruebas de integración comprueban la interacción mutua entre componentes (subsistemas) software entre sí:
 - ▶ Interfaces con otros **componentes**
 - ▶ Interfaces **GUIs / APIs**
- ▶ Las pruebas de integración comprueban las interfaces con el entorno del sistema
 - ▶ En la mayoría de los casos la interacción probada es el comportamiento del componente y el **entorno simulado**
- ▶ Los **casos de prueba** podrán ser obtenidos a partir de:
 - ▶ **Especificación de interfaces**
 - ▶ Diseño de la arquitectura (diseño)
 - ▶ Modelo de datos

III. Pruebas a través del ciclo de vida software

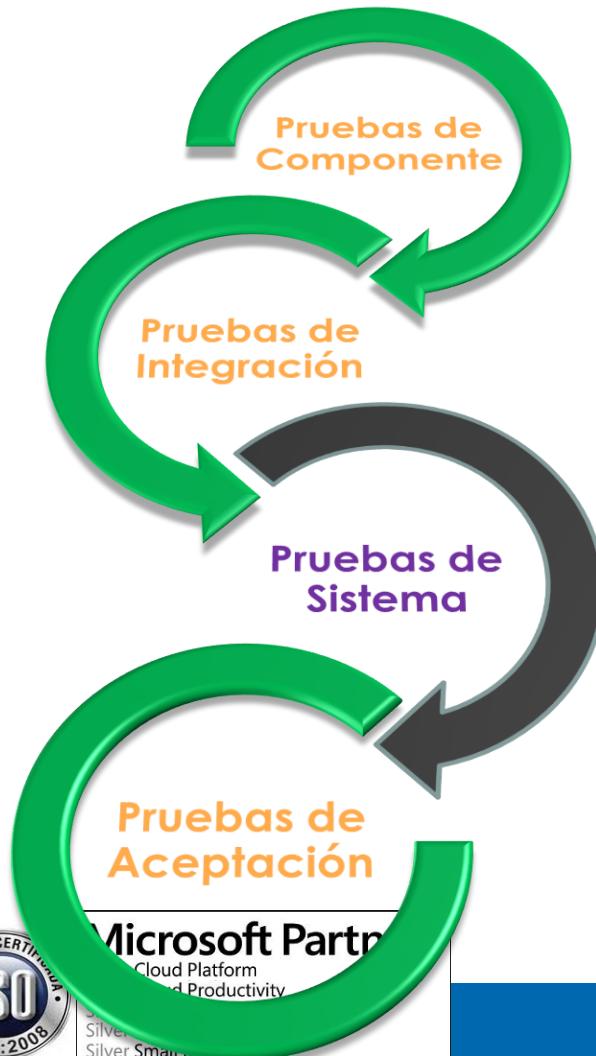
02. Niveles de Pruebas

Pruebas de integración: Estrategias



III. Pruebas a través del ciclo de vida software

02. Niveles de Pruebas



Pruebas de Sistema

Proceso de probar un sistema integrado con el objeto de comprobar el cumplimiento de requisitos especificados

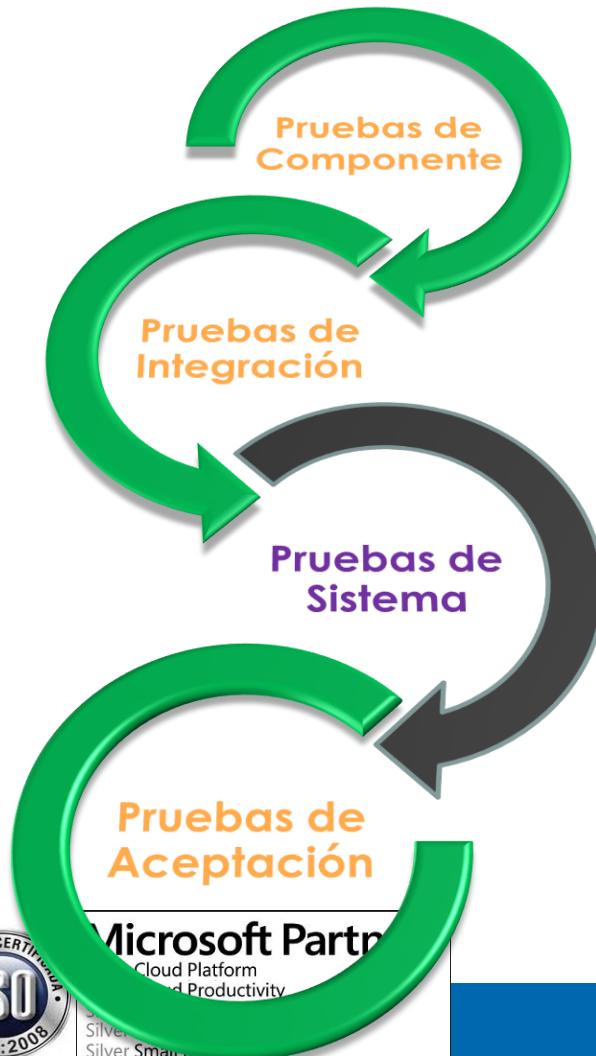
► Bases de prueba

- ▶ Especificación de requisitos software y de sistema
- ▶ Casos de uso
- ▶ Especificación funcional
- ▶ Informes de análisis de riesgos ("risk analysis reports")
- ▶ Manuales de sistema, usuario y operaciones
- ▶ Configuración del sistema
- ▶ Datos de configuración ("configuration data")

► Las pruebas de sistema significa el comportamiento del sistema completo

III. Pruebas a través del ciclo de vida software

02. Niveles de Pruebas

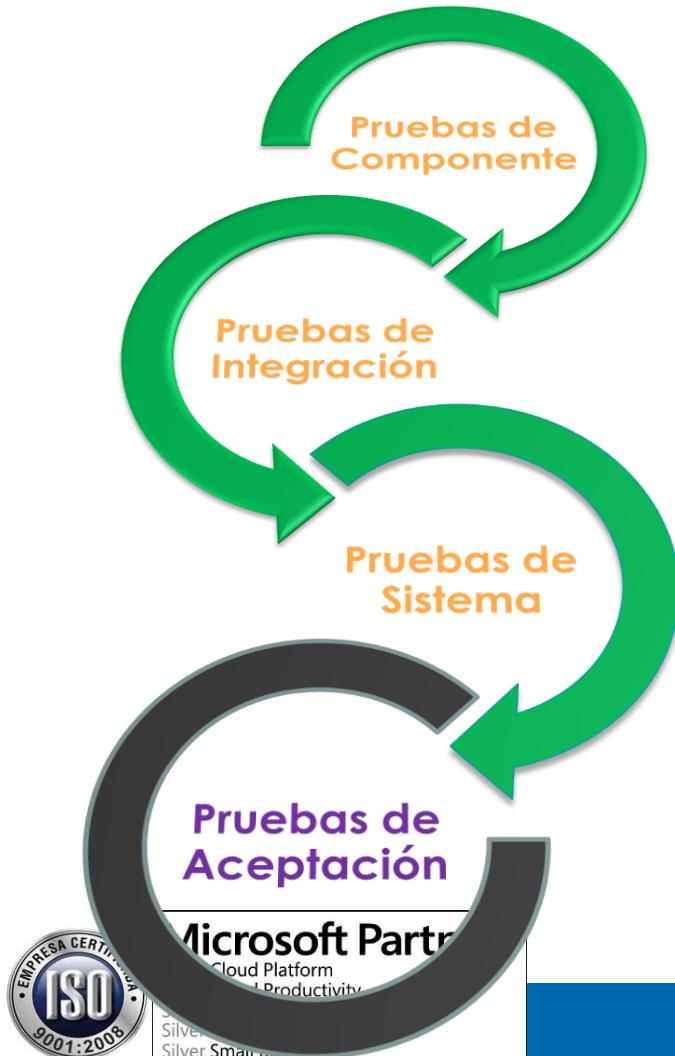


Pruebas de Sistema: Alcance

- ▶ Prueba de un sistema integrado desde el **punto de vista del usuario**
 - ▶ Implementación completa y correcta de los requisitos
- ▶ El **entorno de pruebas** debería coincidir preferiblemente con el **entorno real**
 - ▶ No son necesarios stubs o controladores ("drivers")
 - ▶ Todas las interfaces externas son probadas en condiciones reales
- ▶ **¡No se realizan pruebas en el entorno real!**
 - ▶ Los defectos inducidos podrían dañar el entorno real
 - ▶ Un software objeto de despliegue se encuentra en un estado de cambio constante. La mayoría de las pruebas no serán reproducibles

III. Pruebas a través del ciclo de vida software

02. Niveles de Pruebas



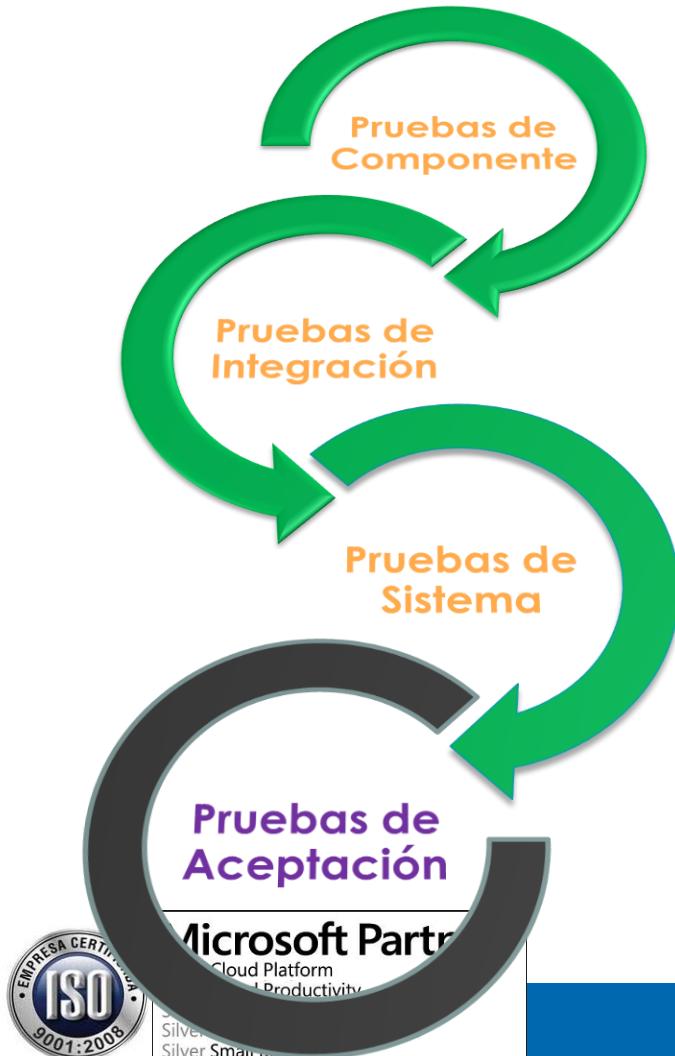
Pruebas de Aceptación

Pruebas formales con respecto a las necesidades de usuario, requisitos y procesos de negocio dirigidas a determinar si el sistema satisface o no los criterios de aceptación y a habilitar al usuario, cliente u otra entidad autorizada a determinar si acepta o no el sistema

- ▶ Bases de prueba
 - ▶ Requisitos de usuario
 - ▶ Requisitos de sistema
 - ▶ Casos de uso
 - ▶ Procesos de negocio
 - ▶ Informes de análisis de riesgo
- ▶ Objetos de prueba típicos
 - ▶ Procesos de negocio en sistemas completamente integrados
 - ▶ Procesos de operaciones y mantenimiento
 - ▶ Procedimientos de usuario
 - ▶ Formularios
 - ▶ Informes

III. Pruebas a través del ciclo de vida software

02. Niveles de Pruebas



Pruebas de Aceptación: Tipos

Pruebas de aceptación contractual y de regulación

- ▶ ¿El software satisface todos los requisitos contractuales?
 - ▶ Con la aceptación formal se cumplen hitos legales: comienzo de fase de garantía, hitos de abono (pago), acuerdos de mantenimiento, etc.
 - ▶ Criterios de aceptación verificables definidos en el momento del acuerdo contractual constituyen una garantía para ambas partes.
 - ▶ Las pruebas de aceptación deben tener en cuenta normas y reglamentos gubernamentales, legales, industriales y de otro tipo.

Pruebas de aceptación de usuario

- ▶ Normalmente el **cliente** selecciona **casos de prueba** para las pruebas de aceptación
- ▶ Normalmente se verifica la adecuación al uso del sistema por parte de usuarios de negocio, "los clientes conocen su negocio"

III. Pruebas a través del ciclo de vida software

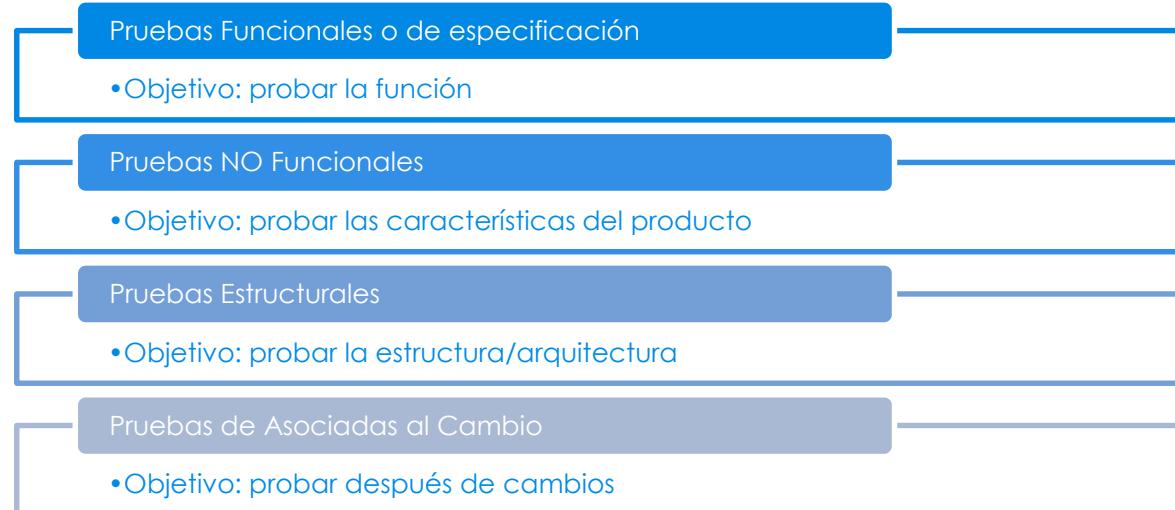
03. Tipos de pruebas

Tipos de pruebas y niveles de pruebas

Niveles de prueba

- En la sección anterior se han explicado los distintos niveles de pruebas, es decir pruebas de componente, pruebas de integración, etc.
- En cada nivel de prueba los objetivos de las pruebas tienen un foco diferente;
- Por lo tanto, se aplican distintos tipos de pruebas durante los distintos niveles de pruebas

Tipos de pruebas



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



III. Pruebas a través del ciclo de vida software

03. Tipos de pruebas

A. Pruebas de la función (Pruebas funcionales) (1)

- **Objetivo:** la función del objeto de prueba
- La funcionalidad puede ser vinculada a los **datos de entrada y salida** de un objeto de prueba
- Los **métodos de caja negra (“black box”)** se utilizan en el diseño de casos de prueba relevantes
- Las pruebas son para verificar los requisitos funcionales (establecidos en las especificaciones, conceptos, casos de estudio, reglas de negocio o documentos relacionados)
- **Ámbito de aplicación**
- Las pruebas funcionales se pueden llevar a cabo en todos los niveles de prueba



III. Pruebas a través del ciclo de vida software

03. Tipos de pruebas

A. Pruebas de la función (Pruebas funcionales) (2)

- Ejecución

- El objeto de prueba es ejecutado utilizando combinaciones de datos de prueba derivados/generados a partir de los casos de prueba
- Los resultados de la ejecución de la prueba son comparados con los resultados esperados



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

III. Pruebas a través del ciclo de vida software

03. Tipos de pruebas

B. Pruebas no funcionales (1)

- Objetivo: características del producto software

- ¿De qué forma el software lleva a cabo sus funciones?
A menudo, las características de calidad no funcionales (ISO 9126: fiabilidad, usabilidad, eficiencia, mantenibilidad, portabilidad) son vagas, incompletas o inexistentes, dificultando las pruebas asociadas a las mismas

- Ámbito de aplicación

- Las pruebas no funcionales se pueden llevar a cabo en todos los niveles
- Pruebas no funcionales típicas:
 - Pruebas de carga ("load testing") / pruebas de rendimiento ("performance testing") / pruebas de volumen ("volume testing") / pruebas de estrés ("stress testing")
 - Pruebas de seguridad (efectos adversos) del software ("Testing of safety features")
 - Prueba de fiabilidad y robustez ("reliability and robustness testing")
 - Pruebas de usabilidad ("usability testing")



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business

 ISTQB
International Software Testing Qualifications Board
Silver Partner

 ORACLE
Gold Partner

III. Pruebas a través del ciclo de vida software

03. Tipos de pruebas

B. Pruebas no funcionales (2)

Ejecución

- La conformidad con los requisitos no funcionales se miden utilizando requisitos funcionales (seleccionados)



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

III. Pruebas a través del ciclo de vida software

03. Tipos de pruebas

B. Pruebas no funcionales (pruebas de desempeño o eficiencia) (3)

Este tipo de prueba permite determinar el rendimiento de un producto software midiendo lo rápido que realiza una tarea determinada bajo condiciones particulares de trabajo.

Adicionalmente este tipo de prueba permite validar y verificar otros atributos que son parte de la calidad del sistema, tales como la escalabilidad, fiabilidad y uso de los recursos.

Las pruebas de rendimiento pueden servir para diferentes propósitos, es por ello que dependiendo del objetivo perseguido podemos diseñar distintos tipos de pruebas.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business

 ISTQB
International Software
Testing Qualifications Board
Silver Partner

 ORACLE
Gold
Partner

III. Pruebas a través del ciclo de vida software

03. Tipos de pruebas

B. Pruebas no funcionales (pruebas de desempeño o eficiencia) (4)

Prueba de carga (“load test”)

Sistema expuesto a una carga (carga mínima, más usuarios/transacciones)

Prueba de rendimiento (“performance test”)

Rapidez con la cual un sistema ejecuta una determinada función

Prueba de volumen (“volume test”)

Procesamiento de grandes cantidades de datos/ficheros

Prueba de estrés (“stress test”)

Reacción a la sobrecarga/recuperación tras el retorno a un carga normal

Prueba de estabilidad (“stability test”)

Rendimiento en “modo de operación continua”

Prueba de robustez (“test for robustness”)

Reacción a entradas erróneas o datos no especificados

Reacción a fallos hardware/recuperación ante situaciones de desastre



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



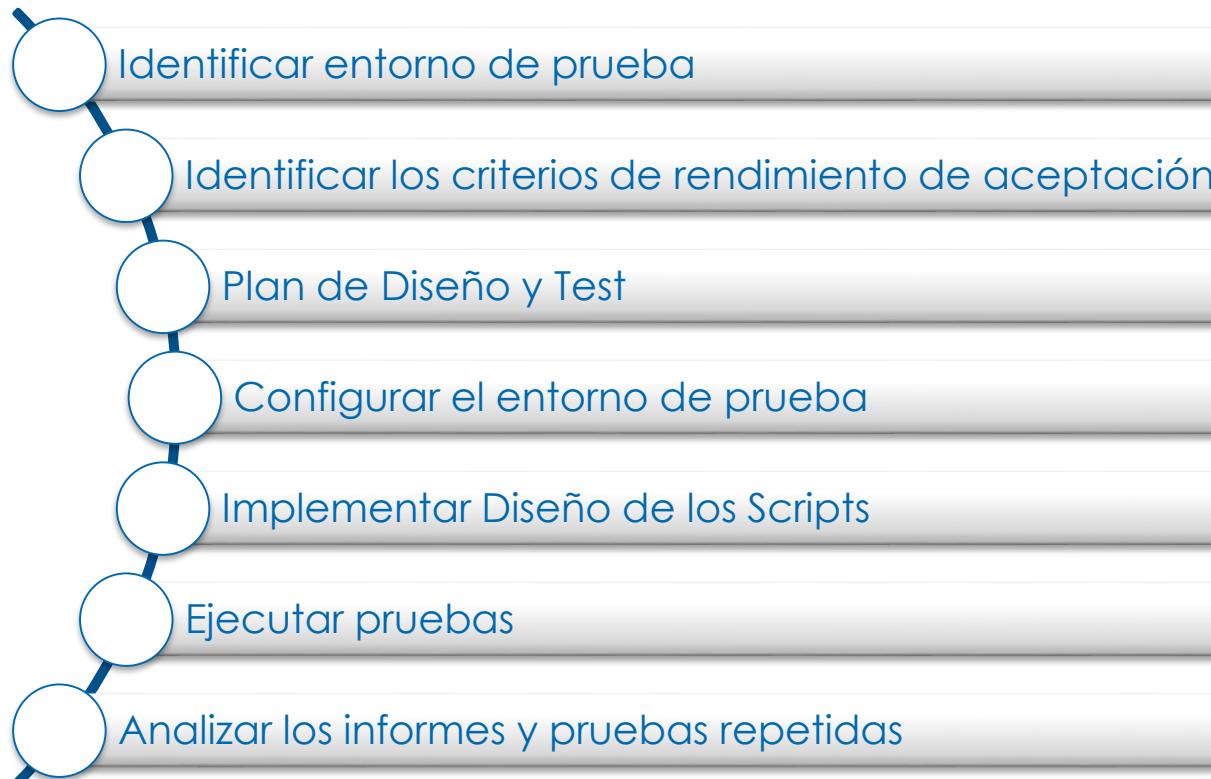
ORACLE Gold Partner

III. Pruebas a través del ciclo de vida software

03. Tipos de pruebas

B. Pruebas no funcionales (pruebas de desempeño o eficiencia) (5)

La estrategia básica para la aplicación de pruebas de rendimiento consiste en:



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



III. Pruebas a través del ciclo de vida software

03. Tipos de pruebas

B. Pruebas no funcionales (pruebas de seguridad) (6)

Las Pruebas de Seguridad son un proceso que permite validar que un sistema de información protege sus datos y funciona de acuerdo a los propósitos para los que fue diseñado.

Adicionalmente las pruebas de seguridad validan que la aplicación y la infraestructura que la soporta no evidencian vulnerabilidades que puedan ser aprovechadas por terceros para uso no deseado.

Las Pruebas de Seguridad se enfocan, estratégicamente, en uno o varios de los siguientes aspectos de la información:

- **Confidencialidad de la información:** Evitar que usuarios o sistemas no autorizados accedan a la información.
- **Integridad de la información:** Asegurar la exactitud y la completitud de la información, así como los métodos que se utilizan para su procesamiento.
- **Disponibilidad de la información:** Asegurar que la información esté disponible para usuarios y sistemas autorizados en el momento que lo requieran.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

III. Pruebas a través del ciclo de vida software

03. Tipos de pruebas

B. Pruebas no funcionales (pruebas de seguridad) (7)

La estrategia más básica para la aplicación de pruebas de seguridad consiste en:

- Identificar cada tipo de usuario o entidad, las funciones y datos a los que se debe acceder.
- Crear pruebas para cada tipo de usuario y verificar cada permiso, creando transacciones específicas para cada tipo de usuario.
- Modificar tipos de usuarios y volver a ejecutar las pruebas.
- Validar la implementación de protocolos de encriptación.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

III. Pruebas a través del ciclo de vida software

03. Tipos de pruebas

B. Pruebas no funcionales (pruebas de sistema) (8)

Pruebas de cumplimiento

- Cumplir normas y reglamentos (interno/externo)

Otros aspectos no funcionales de calidad:

- Portabilidad: adaptabilidad, reemplazabilidad, instalabilidad, coexistencia, cumplimiento de portabilidad
- Mantenibilidad: analizabilidad, modificabilidad, estabilidad, testabilidad, cumplimiento de mantenibilidad
- Fiabilidad: madurez, tolerancia a fallos, recuperabilidad, cumplimiento de fiabilidad



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

III. Pruebas a través del ciclo de vida software

03. Tipos de pruebas

C. Pruebas de la estructura software/Arquitectura (pruebas estructurales) (1)

Objetivo: Cobertura

- Análisis de la estructura de un objeto de prueba (enfoque: **caja blanca**)
- La finalidad de las pruebas es medir el grado en el cual la estructura del objeto de prueba ha sido cubierto por los casos de prueba

Ámbito de aplicación

- Las pruebas estructurales son posibles en todos los niveles de prueba, la **cobertura del código** se realiza normalmente de forma conjunta a las pruebas de componente y de integración mediante el uso de herramientas
- El diseño de pruebas estructurales se finaliza tras haber sido diseñadas las pruebas funcionales, con el propósito de obtener un alto grado de cobertura

Ejecución

- Se probará la estructura interna de un objeto de prueba (por ejemplo el flujo de control en el interior de un componente, el flujo a través de la estructura de un menú)

Objetivo: todos los elementos estructurales identificados deberán estar cubiertos por casos de prueba



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE
Gold Partner

III. Pruebas a través del ciclo de vida software

03. Tipos de pruebas

D. Pruebas asociadas al cambio (3)

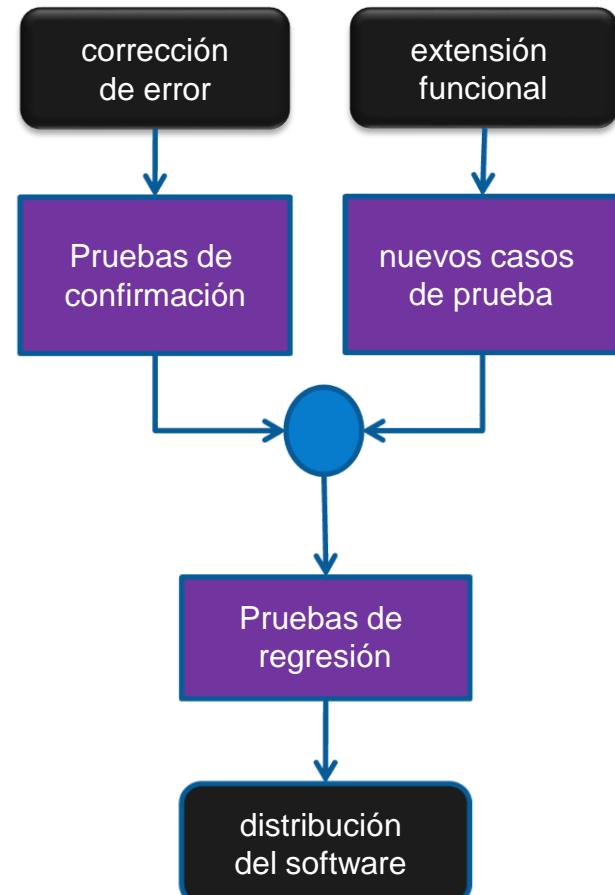
Objetivo: Probar los objetos después de cambios

Después de que un objeto de prueba o el entorno de su sistema ha recibido alguna **modificación** los resultados de pruebas asociadas al cambio resultan inválidos por lo cual posiblemente algunas **pruebas** deben ser **repetidas**

Dos razones para modificar el software

- **Corrección de errores**
- **Extensión funcional**

Debido a los efectos secundarios de la funcionalidad **extendida o nueva**, es necesario también repetir pruebas de áreas adyacentes



Microsoft Partner

Gold Cloud Platform
 Silver Cloud Productivity
 Silver Collaboration and Content
 Silver Application Development
 Silver Small Business



ISTQB
 International Software Testing Qualifications Board
 Silver Partner

ORACLE
 Gold Partner

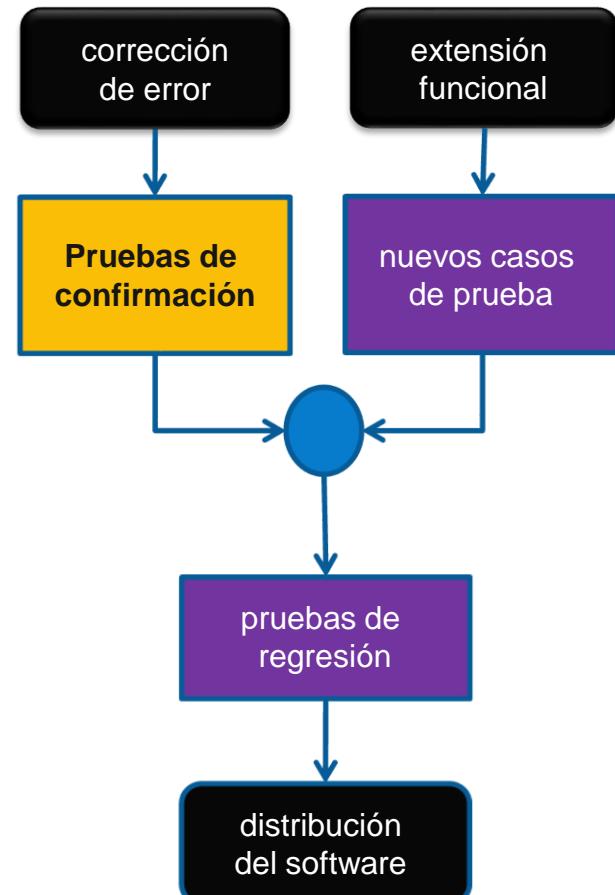
III. Pruebas a través del ciclo de vida software

03. Tipos de pruebas

D. Pruebas asociadas al cambio (3)

Pruebas de Confirmación:

- Pruebas que ejecutan aquellos casos de prueba que hubieran fallado la última vez que fueron ejecutados con el objetivo de verificar el éxito de acciones correctivas.



Microsoft Partner

Gold Cloud Platform
 Silver Cloud Productivity
 Silver Collaboration and Content
 Silver Application Development
 Silver Small Business

ISTQB
International Software Testing Qualifications Board

ORACLE
Gold Partner

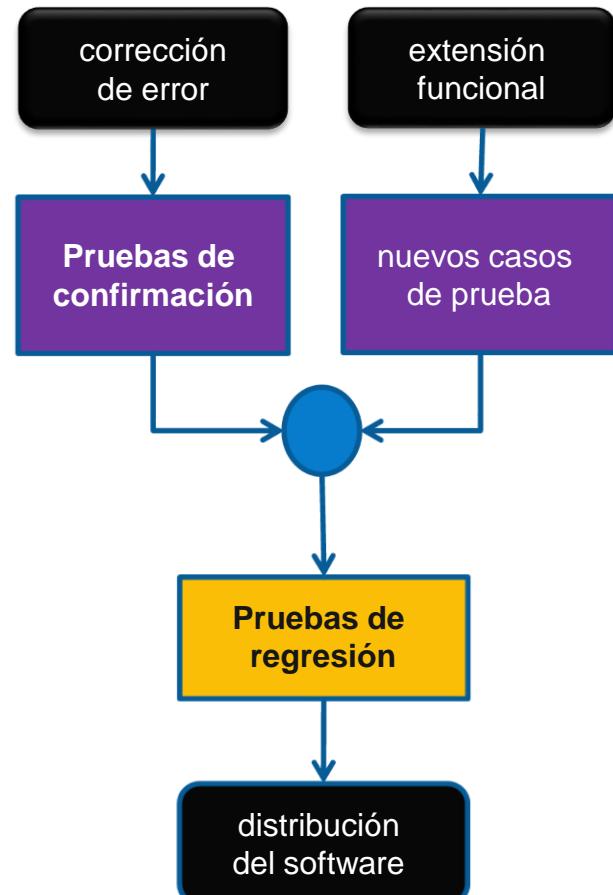
III. Pruebas a través del ciclo de vida software

03. Tipos de pruebas

D. Pruebas asociadas al cambio (3)

Pruebas de Regresión:

- Pruebas de un programa previamente probado que ha sufrido modificaciones, para asegurarse que no se han introducido o descubierto defectos en áreas del software que no han sido modificadas como resultado de los cambios realizados.
- Estas también se realizan cuando el software o su entorno ha sido modificado.



Microsoft Partner

Gold Cloud Platform
 Silver Cloud Productivity
 Silver Collaboration and Content
 Silver Application Development
 Silver Small Business

ISTQB
 International Software Testing Qualifications Board
 Silver Partner

ORACLE
 Gold Partner

III. Pruebas a través del ciclo de vida software

04. Otros tipos de pruebas

Pruebas posteriores a la aceptación del producto (1/4)

El **cliente** ha **aprobado** el producto y es puesto en producción

- El ciclo de desarrollo inicial, incluidas las **pruebas** asociadas, ha sido completado

El mismo software se encuentra al comienzo del ciclo de vida (de mantenimiento):

- Será utilizado por muchos años, será **ampliado**
- Es muy probable que aún contenga defectos, por lo tanto será modificado y **corregido**
- Necesitará adaptarse a nuevas condiciones y deberá integrarse a nuevos entornos
- Necesitará cambiar o extender los datos de **configuración**
- Será retirado, se extraerá del entorno de producción



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



III. Pruebas a través del ciclo de vida software

04. Otros tipos de pruebas

Pruebas posteriores a la aceptación del producto (2/4)

¡Cualquier nueva versión del producto, cada nueva actualización y cada cambio del software requiere pruebas adicionales!

Pruebas de mantenimiento

- Pruebas de los cambios en un sistema en operación o el impacto de un entorno modificado para un sistema en operación



III. Pruebas a través del ciclo de vida software

04. Otros tipos de pruebas

Pruebas posteriores a la aceptación del producto (3/4)

- El mantenimiento de software cubre dos campos diferenciados:
 - **Mantenimiento** tal como: corrección de errores o implementación de “**hot-fixes**”, que han sido parte de la versión inicial del software
 - **Distribuciones de software planificados**: adaptaciones como resultado de una modificación/cambio del entorno o nuevos requisitos del cliente
- Alcance de las pruebas de mantenimiento:
 - “Hot-fixes” y la corrección de defectos requiere la repetición de pruebas (“*re-test*”)
 - La ampliación de la funcionalidad requiere nuevos casos de prueba
 - La migración a otra plataforma requiere pruebas operativas (“*operational testing*”)
- **Adicionalmente**, son necesarias **pruebas de regresión** intensivas



III. Pruebas a través del ciclo de vida software

04. Otros tipos de pruebas

Pruebas posteriores a la aceptación del producto (4/4)

- El alcance de las pruebas depende del impacto del cambio
 - El **análisis de impacto** se utiliza para determinar las áreas afectadas con el objeto de decidir la cantidad de pruebas de regresión
 - Pueden ocurrir problemas si la **documentación** del software antiguo **falta** o es **incompleta**
-
- **Pruebas de Retirada** del software
 - Las pruebas, tras la retirada del software, pueden incluir:
 - Pruebas de migración de datos
 - Verificación del archivo de datos y programas
 - Pruebas en paralelo de sistemas nuevo y su antecedente



III. Pruebas a través del ciclo de vida software

04. Otros tipos de pruebas

Otros tipos de prueba

- **Pruebas de Humo (Smoke testing):** Subconjunto de todos los casos de prueba definidos que cubren la funcionalidad principal de un componente o sistema, con el objeto de asegurar que las funciones cruciales de un programa funcionan, pero sin preocuparse por los detalles finos.
- **Pruebas de Admisión:** Casos especiales de pruebas de humo cuyo objetivo es decidir si el componente o sistema está en condiciones de ser probado en detalle y proseguir con el proceso de pruebas.
- **Pruebas Exploratorias (Exploratory testing):** Técnica informal de diseño de pruebas donde quien prueba controla activamente el diseño de las pruebas a medida que las pruebas son realizadas y utiliza la información obtenida durante las pruebas para diseñar unas nuevas y mejores.
- **Pruebas de Mono (Monkey testing):** El usuario pone a prueba la aplicación o sistema (Normalmente sistemas que operan en tecnología táctil), proporcionando entradas al azar y comprobar el comportamiento buscando el bloqueo o falla del sistema.

III. Pruebas a través del ciclo de vida software

04. Otros tipos de pruebas

Otros tipos de prueba

Pruebas “Caja Gris” o pruebas traslúcidas

El concepto de testing de caja gris es simple, se basa en la realización de pruebas de caja negra basadas en casos de prueba realizados por personas que conocen el programa por dentro y en una prueba funcional observan también el comportamiento interno del sistema.

Este tipo de prueba es una mezcla de caja negra y caja blanca, considerándose como un tipo de testing en donde el probador sabe como funciona el programa, es decir, ve el código fuente pero trabaja sin poder modificarlo. Inclusive considera que sus casos de pruebas funcionales tienen la capacidad de recorrer el código.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



III. Pruebas a través del ciclo de vida software

04. Otros tipos de pruebas

Otros tipos de prueba

Pruebas “Caja Gris” o pruebas traslúcidas

Ventajas:

- Acercamiento para la solución de problemas estructurales y funcionales.
- Beneficios combinados de caja negra y caja blanca.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

III. Pruebas a través del ciclo de vida software

04. Otros tipos de pruebas

Otros tipos de prueba

Pruebas “Caja Gris” o pruebas traslúcidas

Desventajas:

- Requiere un esfuerzo adicional para probadores ya que debe tener un conocimiento técnico-funcional que se complementen.
- Método superficial ya que no tiene un objetivo de prueba definido (especificación o estructura).



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

III. Pruebas a través del ciclo de vida software

04. Otros tipos de pruebas

Otros tipos de prueba

Pruebas “Caja Gris” o pruebas traslúcidas



El Ying y el Yang de las Pruebas



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



Ingeniería de requerimiento

- Introducción a la ingeniería de requisitos
- Conceptos
- Aspectos para definirlos
- Estructuras
- Criterios de Calidad
- Validación de RQ
- Atributos de los RQ
- Herramienta de gestión de requisitos
- Trazabilidad de Requisitos



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

IV. Ingeniería de requerimiento

01. Introducción a la ingeniería de requisitos

Introducción

Una buena IR es importante dado que muchos errores surgen en las fases tempranas del ciclo de vida del desarrollo de un producto de software y sólo pueden ser rectificados posteriormente pero con altos costes.

Los síntomas típicos de una IR inapropiada son la ausencia o poca claridad de los requisitos. Algunos de los motivos habituales que conducen a una IR inadecuada son:

- La suposición errónea por parte de los implicados que muchas cosas se explican por sí mismas y no necesitan ser establecidas de forma explícita;
- La existencia de problemas de comunicación basados en diferencias de conocimiento y experiencia;
- La presión sobre el proyecto ejercida por parte del cliente para conseguir a corto plazo un sistema en producción.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE
Gold
Partner

IV. Ingeniería de requerimiento

02. Conceptos

Requerimiento

1. Una condición o capacidad que necesita un usuario para resolver un problema o alcanzar un objetivo.
2. Una condición o capacidad que debe cumplir un sistema o componente para satisfacer un contrato, estándar, especificaciones, u otros documentos impuestas formalmente.
3. Una representación documentada de una condición o capacidad para cumplir una necesidad.[IEEE Std 610.12 de 1990]

Alternativamente, también damos una definición más moderna:

- a. Una necesidad percibida por un stakeholder.
- b. Una capacidad o la propiedad que un sistema tendrá.
- c. Una representación de una necesidad, la capacidad o la propiedad documentada.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE
Gold Partner

IV. Ingeniería de requerimiento

02. Conceptos

Requerimiento Funcional

Es un requerimiento relativo al resultado de un comportamiento deseado por parte de una función de un sistema, componente o servicio.

Expresan la naturaleza del funcionamiento del sistema (cómo interacciona el sistema con su entorno y cuáles van a ser su estado y funcionamiento).

NOTA: Es conveniente indicar lo que **NO** hará el sistema al definir requerimientos.



IV. Ingeniería de requerimiento

02. Conceptos

Requerimiento No Funcional

Requisitos que especifica criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos, ya que éstos corresponden a los requisitos funcionales.

En tal sentido este tipo de requerimiento se refiere a todo lo que no describe la información a guardar, ni funciones a realizar, pero sí a atributos tales como seguridad, eficiencia, usabilidad, mantenibilidad y portabilidad (atributos no funcionales). Estos han de especificarse cuantitativamente, siempre que sea posible para que se pueda verificar su cumplimiento.



IV. Ingeniería de requerimiento

02. Conceptos



International
Requirements
Engineering
Board

Diferencias

Los requisitos funcionales definen
qué debe hacer un sistema.

Los requisitos no funcionales definen
cómo debe ser el sistema.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



IV. Ingeniería de requerimiento

03. Aspectos para definirlos

Aspectos importantes para definir requerimientos

La fuente de los requisitos de un sistema y, por lo tanto, su justificación, se encuentra en el contexto del sistema planificado.

La fuente consiste en el conjunto de todos los aspectos del contexto que han originado o influido en la definición de los requisitos. Entre los aspectos potenciales residentes en el contexto del sistema están:

- Personas (implicados o grupos de implicados)
- Sistemas en producción (sistemas técnicos, software y hardware)
- Procesos (procesos técnicos o físicos, procesos de negocio)
- Eventos (técnicos o físicos)
- Documentos (por ejemplo, leyes, estándares, documentación del sistema)



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE
Gold Partner

IV. Ingeniería de requerimiento

04. Estructura

Estructura de un documento de Requerimiento

Las estructuras de referencia para documentos de requisitos proponen estructuras de contenido completas y flexibles que han sido contrastadas en la práctica. El estándar ISO/IEC/IEEE 29148:2011, entre otros, describe estructuras de referencia comunes para documentos de requisitos.

En la práctica, el uso de estructuras de referencia para documentos de requisitos aporta una gran cantidad de resultados positivos.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

IV. Ingeniería de requerimiento

04. Estructura

Importancia del Uso de los Documentos de Requisitos

Los documentos de requisitos son la base para un gran número de tareas durante la vida de un proyecto, tales como:

- Planificación
- Diseño de arquitectura
- Implementación
- Pruebas
- Gestión del cambio
- Uso y mantenimiento del sistema
- Gestión de contratos



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

IV. Ingeniería de requerimiento

05. Criterios de Calidad

Criterios de Calidad para los Documentos de Requisitos (1/2)

Para poder servir como base de las actividades posteriores en el proceso de desarrollo y pruebas un documento de requisitos debe cumplir ciertos criterios de calidad. En particular, entre estos criterios se incluyen:

Ausencia de ambigüedad y consistencia



Estructura clara



Capacidad de ser modificado y capacidad de ser ampliado



Compleitud



Trazabilidad



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business

 ISTQB
International Software
Testing Qualifications Board
Silver Partner

 ORACLE
Gold
Partner

IV. Ingeniería de requerimiento

05. Criterios de Calidad

Criterios de Calidad para los Documentos de Requisitos (2/2)

Ausencia de ambigüedad y consistencia

- Todos los involucrados en la definición y revisión de un documento de requerimiento deben llegar a una misma y consistente interpretación del mismo.

Estructura clara

- El documento debe demostrar que ningún requerimiento entra en conflicto con otro, es fácil de demostrar la relación de asociación, secuencia y dependencia entre ellos. El lenguaje usado en su definición, no debe causar confusiones al lector.

Capacidad de ser modificado y capacidad de ser ampliado

- Los requerimientos son expresados de tal manera que cada ítem o característica que lo componen puede ser cambiado sin causar un gran impacto a los demás ítems.

Completitud

- Cada requerimiento debe describir de manera completa las funcionalidades que debe cumplir el sistema. Debe contener toda la información necesaria para que el desarrollador diseñe e implemente tal funcionalidad.

Trazabilidad

- Cada requerimiento debe poder ser seguido durante cada etapa del proyecto durante su ciclo de vida.



Microsoft Partner

Gold Cloud Platform
 Silver Cloud Productivity
 Silver Collaboration and Content
 Silver Application Development
 Silver Small Business



IV. Ingeniería de requerimiento

05. Criterios de Calidad

Criterios de Calidad Para los Requisitos (1/4)

Además de los criterios de calidad de los documentos como un todo, cada uno de los requisitos individuales debe cumplir con ciertos criterios de calidad, en particular:



IV. Ingeniería de requerimiento

05. Criterios de Calidad

Criterios de Calidad Para los Requisitos (2/4)

Consensuado: Todas las características que componen un requerimiento deben poseer el consentimiento activo de cada uno de los involucrados, teniendo en cuenta que esto no implica un consentimiento activo de cada uno, sino más bien una aceptación en el sentido de no-negación.

No ambiguo: Cada requerimiento es exacto (Preciso) y no vago (Claro); hay una sola interpretación; el significado de cada ítem del requerimiento es entendido; la especificación es fácil de entender. Un requerimiento no es ambiguo cuando tiene una sola interpretación.

Necesario: Un requerimiento es necesario si su omisión provoca una deficiencia en el sistema a construir, y además su capacidad, características físicas o factor de calidad no pueden ser reemplazados por otras capacidades del producto o del proceso.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE
Gold Partner

IV. Ingeniería de requerimiento

05. Criterios de Calidad

Criterios de Calidad Para los Requisitos (3/4)

Consistente: Ningún ítems de algún requerimiento entra en conflicto con otro de la especificación.

Verificable: Durante el desarrollo del programa y el proceso de prueba, es posible determinar si los ítem de un requerimiento han quedado satisfecho.

Realizable: Debe ser posible implementar cada requerimiento de acuerdo a las capacidades, limitaciones del sistema, el medio que lo rodea y los recursos que se poseen.

Trazable: Un requerimiento es trazable si se pueden identificar todas las partes del producto existente relacionadas con ese requerimiento y sus items.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE
Gold Partner

IV. Ingeniería de requerimiento

05. Criterios de Calidad

Criterios de Calidad Para los Requisitos (4/4)

Completo: Todos los ítems o características necesarias para la especificación del requerimiento está incluído. Un requerimiento está completo si no necesita ampliar detalles en su redacción, es decir, si se proporciona la información suficiente para su comprensión.

Comprensible: Un requerimiento es comprensible cuando es fácil de leer y entender. Su redacción debe ser simple y clara para aquellos que vayan a consultarla en un futuro.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

IV. Ingeniería de requerimiento

05. Criterios de Calidad



Criterios de Calidad Para los Requisitos

Además de estos criterios de calidad, existen dos reglas básicas de estilo para los requisitos expresados en lenguaje natural que ayudan a su legibilidad:

- Frases y párrafos cortos.
- Sólo un requisito por frase.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



IV. Ingeniería de requerimiento

06. Validación de RQ

Validación (1/3)

El principal objetivo de la validación de requisitos es determinar si éstos cumplen los criterios de calidad que han sido fijados previamente (ver UE 4.6) con el objetivo de detectar y corregir posibles anomalías tan pronto como sea posible. Puesto que los documentos de requisitos son la base para todas las actividades de desarrollo posteriores, cualquier tipo de error no detectado en los requisitos afecta todas las actividades subsiguientes de tal manera que el esfuerzo de corregir un error no detectado en un requisito se eleva de forma significativa en el curso de un desarrollo. Esto se debe a que no basta con corregir el error en el requisito, sino que deben reconstruirse todos aquellos artefactos basados en el mismo, por ejemplo el diseño de arquitectura, la implementación, los casos de prueba.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE
Gold Partner

IV. Ingeniería de requerimiento

06. Validación de RQ

Validación (2/3)

La validación de requisitos se basa en una serie de principios. Estos principios aseguran que durante la validación se puedan identificar tantos errores en los requisitos como sea posible. Los seis principios que rigen la validación de requisitos son:

- Intervención de los implicados adecuados
- Separación de los procesos de diagnóstico y corrección de errores
- Validación desde distintos puntos de vista
- Cambio adecuado del tipo de documentación
- Construcción de artefactos de desarrollo basados en los requisitos
- Validación reiterada



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE
Gold Partner

IV. Ingeniería de requerimiento

06. Validación de RQ

Validación (3/3)

Existen varias técnicas para la validación sistemática de requisitos que pueden combinarse, al menos parcialmente, con el objetivo de verificar la adecuación de los requisitos con respecto a los criterios de validación definidos de la forma más completa posible. Entre estas técnicas podemos mencionar:

- Comentarios (opinión de experto)
- Inspecciones
- Revisiones guiadas

También pueden utilizarse técnicas como:

- Lectura basada en la perspectiva
- Validación mediante prototipos
- Uso de listas de comprobación



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

IV. Ingeniería de requerimiento

07. Atributos de los RQ

Asignación de Atributos a los Requisitos (1/3)

Para gestionar los requisitos durante todo el ciclo de vida del sistema, es necesario recoger información sobre los mismos de la forma más estructurada posible mediante atributos. La definición de la estructura de atributos para los requisitos se lleva a cabo mediante un esquema de atributos, que puede definirse en forma tabular o creando un modelo de información. Algunos atributos habituales son:

- Identificador
- Nombre
- Descripción
- Fuente
- Nivel de Riesgo
- Prioridad

La obligatoriedad también puede registrarse como información adicional del requisito mediante un atributo



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



IV. Ingeniería de requerimiento

07. Atributos de los RQ

Asignación de Atributos a los Requisitos (2/3)

Estructura básica:

ID:	RQ_F_00001	Nombre:	El sistema debe permitir al usuario iniciar sesión.
Descripción:		El sistema debe permitir al usuario realizar el inicio de sesión con su correo electrónico y contraseña que definió al momento de registrarse.	
Nivel de Riesgo:	Medio	Prioridad:	Alta
ID:	RQ_NF_00047	Nombre:	Navegador web requerido debe ser Google Chrome
Descripción:		El sistema debe ser compatible y debe estar optimizado para su uso con el navegador Google Chrome versión 45.0.2454.101 m	
Nivel de Riesgo:	Medio	Prioridad:	Alta



Microsoft Partner

Gold Cloud Platform
 Silver Cloud Productivity
 Silver Collaboration and Content
 Silver Application Development
 Silver Small Business



IV. Ingeniería de requerimiento

07. Atributos de los RQ

Asignación de Atributos a los Requisitos (3/3)

Frecuentemente, los esquemas de atributos se definen y adaptan según las necesidades de un proyecto específico en base a condiciones específicas. Entre ellas se encuentran:

- Propiedades específicas del proyecto
- Restricciones de la organización
- Reglas de dominio
- Restricciones del proceso de desarrollo



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

IV. Ingeniería de requerimiento

08. Documentación de Requisitos Basada en Modelos

Uso de Modelos

El uso de modelos facilita la comprensión selectiva de la información de los hechos y sus relaciones, así como su registro de forma más rápida y documentación de forma no ambigua.

Uno de los modelos más eficientes y tradicionales para modelar requisitos son los casos de uso.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



Gold
Partner

IV. Ingeniería de requerimiento

08. Documentación de Requisitos Basada en Modelos

Modelos de Casos de Uso (1/9)

Los casos de uso ayudan a examinar y documentar un sistema planificado o ya existente desde la perspectiva de los usuarios. La aproximación de casos de uso se basa en dos técnicas de documentación complementarias:

- Diagramas de casos de uso
- Especificaciones de casos de uso

Los diagramas de casos de uso son modelos simples para documentar la funcionalidad del sistema desde el punto de vista de los usuarios, y documentar las interrelaciones de las funciones de un sistema y las relaciones entre estas funciones y el contexto del sistema. Los elementos de modelado más habituales de los diagramas de casos de uso son:

- Actores (personas u otros sistemas) en el contexto del sistema
- La frontera del sistema
- Casos de uso
- Diferentes tipos de relaciones entre los anteriores elementos



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



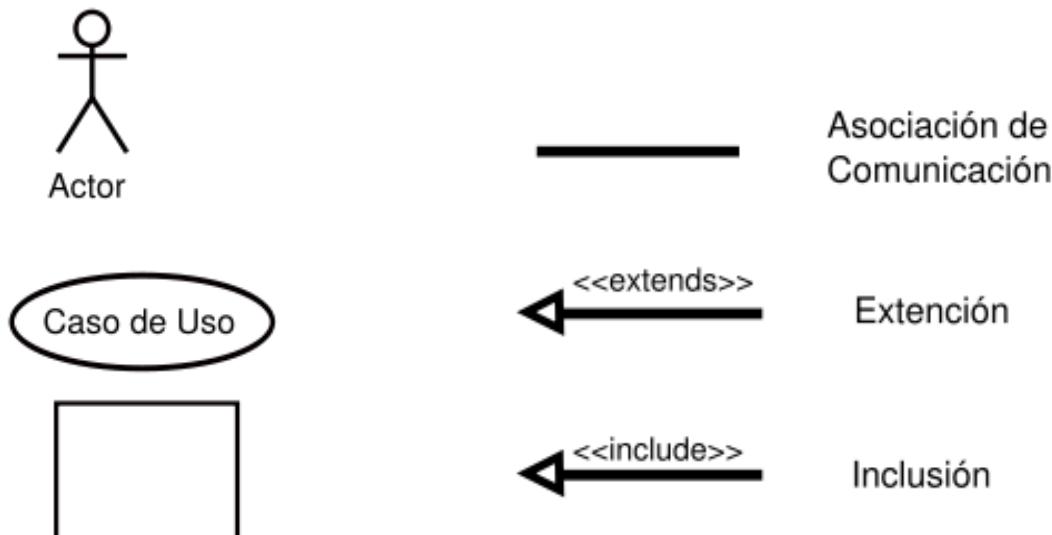
ORACLE
Gold Partner

IV. Ingeniería de requerimiento

08. Documentación de Requisitos Basada en Modelos

Modelos de Casos de Uso (2/9)

- Actores (personas u otros sistemas) en el contexto del sistema
- La frontera o límites del sistema
- Casos de uso
- Diferentes tipos de relaciones entre los anteriores elementos



Microsoft Partner

Gold Cloud Platform
 Silver Cloud Productivity
 Silver Collaboration and Content
 Silver Application Development
 Silver Small Business



Límite de un sistema

ISTQB
 Silver Partner
 International Software
 Testing Qualifications Board

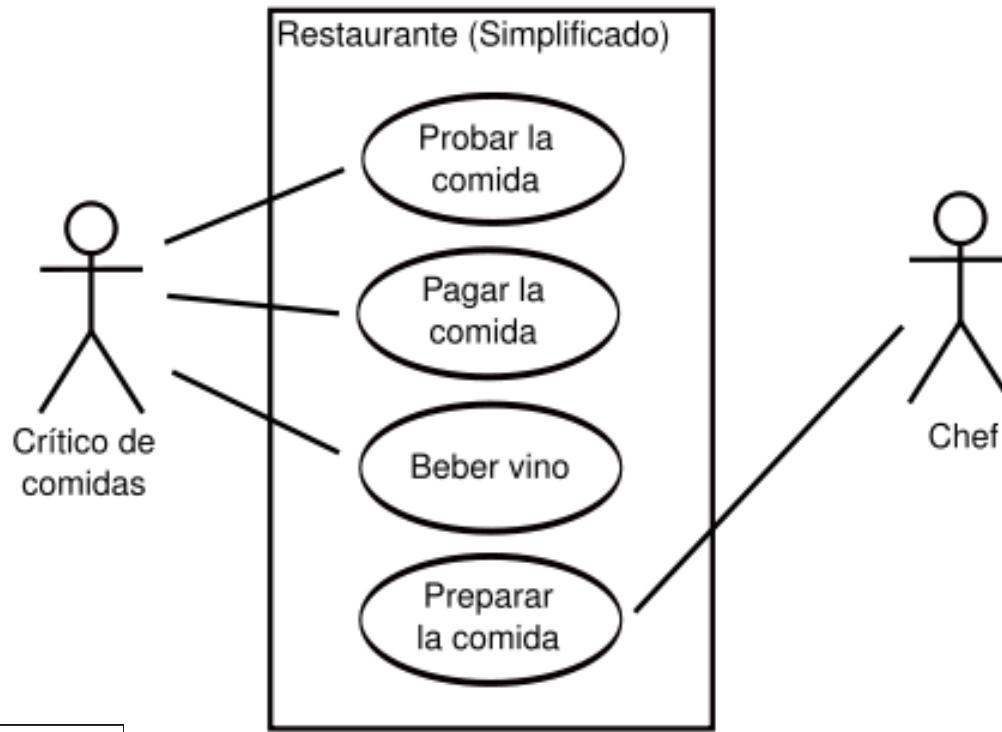
ORACLE
 Gold
 Partner

IV. Ingeniería de requerimiento

08. Documentación de Requisitos Basada en Modelos

Modelos de Casos de Uso (3/9)

Diagrama simple:



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



IV. Ingeniería de requerimiento

08. Documentación de Requisitos Basada en Modelos

Modelos de Casos de Uso (4/9)

Diagrama con casos a incluir y casos a extender

Este tipo de diagrama representa gráficamente una relación entre casos de uso, con el nombre correspondiente al tipo de relación de la que se trate: ya sea <<include>> o <<extend>>.

Estas, son relaciones que usamos para ligar gráficamente dos casos de uso, cuyos flujos de eventos están unidos, normalmente en una sola sesión del usuario. En otras palabras, un caso de uso que está ligado, por medio de una de estas dos relaciones, a otro caso de uso; también podría leerse o ejecutarse como un sólo caso de uso en lugar de dos.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

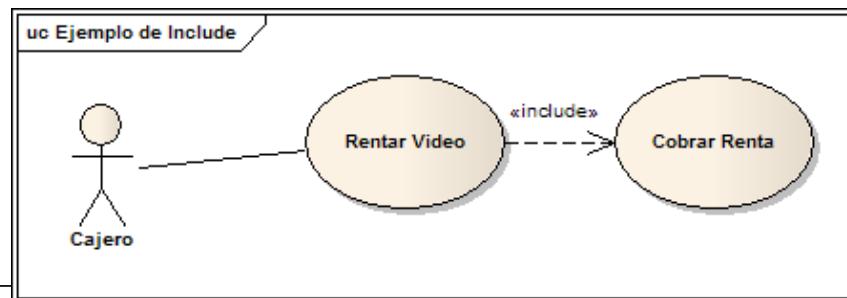
IV. Ingeniería de requerimiento

08. Documentación de Requisitos Basada en Modelos

Modelos de Casos de Uso (5/9)

Diagrama con casos a incluir y casos a extender

Include. En términos muy simples, cuando relacionamos dos casos de uso con un “include”, estamos diciendo que el primero (el caso de uso base) incluye al segundo (el caso de uso incluido). Es decir, el segundo es parte esencial del primero. Sin el segundo, el primero no podría funcionar bien; pues no podría cumplir su objetivo. Para una venta en caja, la venta no puede considerarse completa si no se realiza el proceso para cobrarla en ese momento. El caso de uso “Cobrar Renta” está incluido en el caso de uso “Rentar Video”, o lo que es lo mismo “Rentar Video” incluye (<<include>>) “Cobrar Renta”.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

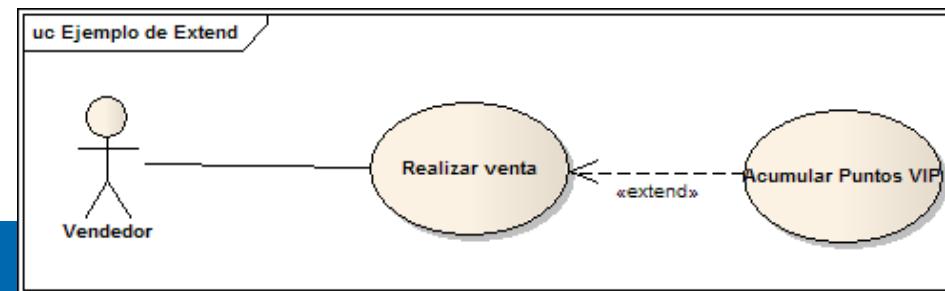
IV. Ingeniería de requerimiento

08. Documentación de Requisitos Basada en Modelos

Modelos de Casos de Uso (6/9)

Diagrama con casos a incluir y casos a extender

Extend. La polémica al querer seleccionar una de las dos relaciones es que en el “extend” también podemos ver, desde la perspectiva del usuario, a los dos flujos como si fueran uno sólo. Y en ciertos escenarios el caso de uso base no podría cumplir su objetivo si no se ejecutara la extensión. Pero, una de las diferencias básicas es que en el caso del “extend” hay situaciones en que el caso de uso de extensión no es indispensable que ocurra, y cuando lo hace ofrece un valor extra (extiende) al objetivo original del caso de uso base. En cambio en el “include” es necesario que ocurra el caso incluido, tan sólo para satisfacer el objetivo del caso de uso base. Ejemplo: Puedes “Realizar Venta” sin “Acumular Puntos de Cliente VIP”, cuando no eres un cliente VIP. Pero, si eres un cliente VIP sí acumularás puntos. Por lo tanto, “Acumular Puntos” es una extensión de “Realizar Venta” y sólo se ejecuta para cierto tipo de ventas, no para todas.



Microsoft Partner

Gold Cloud Platform
 Silver Cloud Productivity
 Silver Collaboration and Content
 Silver Application Development
 Silver Small Business



ORACLE
 Gold
 Partner

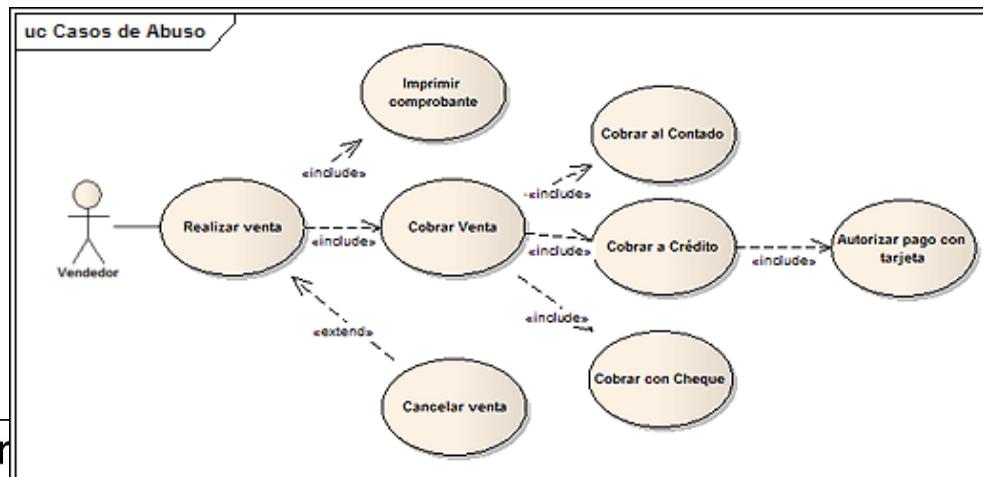
IV. Ingeniería de requerimiento

08. Documentación de Requisitos Basada en Modelos

Modelos de Casos de Uso (7/9)

Casos de Abuso

Uno de los riesgos que existe cuando la gente sabe que tiene estas relaciones como un elemento a utilizar en sus modelos de casos de uso, consiste en su abuso. Mucha gente, y sobre todo la que arrastra prácticas de métodos estructurados, la suele utilizar en exceso. No es raro ver modelos de casos de uso que llegan a tener decenas de inclusiones y extensiones, incluso las inclusiones y extensiones se vuelven a extender a varios niveles, generando una maraña de casos de uso que no ofrecen valor al ser mostrados explícitamente.



IV. Ingeniería de requerimiento

08. Documentación de Requisitos Basada en Modelos

Modelos de Casos de Uso (8/9)

Especificaciones de casos de uso: Define y complementa al diagrama de caso de uso.
Una especificación puede contener los siguientes elementos

- ID
- NOMBRE
- CREADO POR
- ULTIMA ACTUALIZACIÓN POR
- FECHA DE CREACIÓN
- FECHA DE ULTIMA ACTUALIZACIÓN
- ACTORES
- DESCRIPCIÓN O PASO A PASO
- PRE-CONDICIÓN
- POST-CONDICIÓN
- FLUJO NORMAL
- FLUJOS ALTERNATIVOS
- RELACIONES
- FRECUENCIA DE USO
- REGLAS DE NEGOCIO
- ENTIDADES
- NOTAS Y ASUNTO



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



IV. Ingeniería de requerimiento

08. Documentación de Requisitos Basada en Modelos

Modelos de Casos de Uso (9/9)

Elementos básicos mínimos recomendados para una especificaciones de casos de uso.

- ID
- NOMBRE
- ACTORES
- DESCRIPCIÓN O PASO A PASO
- PRE-CONDICIÓN
- POST-CONDICIÓN
- FLUJO NORMAL
- FLUJOS ALTERNATIVOS
- REGLAS DE NEGOCIO



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

IV. Ingeniería de requerimiento

09. Herramientas

Herramientas de gestión de requisitos

Las herramientas de gestión de requisitos son herramientas desarrolladas específicamente para la IR. Estas herramientas deberían ofrecer las características siguientes:

- Gestionar distintos tipos de información
- Gestionar las relaciones lógicas entre la información
- Identificar artefactos de forma unívoca
- Hacer que la información sea accesible de forma flexible y segura, por ejemplo a través del control de acceso
- Soportar distintas vistas de los datos
- Organizar la información, por ejemplo, mediante la asignación de atributos o jerarquías
- Generar informes sobre la información recopilada
- Generar documentos a partir de la información

Las herramientas ofimáticas estándar ofrecen todas estas funcionalidades pero de manera limitada. Las herramientas especializadas de IR refinan estas funcionalidades, por ejemplo, mediante gestión de la trazabilidad.

IV. Ingeniería de requerimiento

10. Trazabilidad de requisitos

Trazabilidad (1/5)

¿Qué es la trazabilidad?

Capacidad de identificar elementos relacionados en la documentación y el software, tales como requisitos con las pruebas asociadas.

En un proyecto con altos niveles de calidad todos los artefactos construidos pueden ser trazables uno con los otros “Trazabilidad bidireccional”.



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

IV. Ingeniería de requerimiento

10. Trazabilidad de requisitos

Trazabilidad (2/5)

Durante la gestión de los requisitos y pruebas como parte de una buena gestión se debe registrar, organizar y mantener información de trazabilidad.

Los beneficios que ofrece la trazabilidad de requisitos son:

- Simplificación de la verificabilidad
- Permite la identificación de características del sistema que no son necesarias
- Permite la identificación de requisitos que no son necesarios
- Apoyo al análisis de impacto
- Apoyo a la reutilización
- Apoyo a la identificación de responsabilidades
- Apoyo al mantenimiento y administración del sistema



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE Gold Partner

IV. Ingeniería de requerimiento

10. Trazabilidad de requisitos

Trazabilidad (3/5)

Con respecto a las relaciones de trazabilidad, deben distinguirse tres clases de relaciones de trazabilidad:

- Trazabilidad pre-especificación de requisitos
- Trazabilidad pos-especificación de requisitos
- Trazabilidad entre requisitos

Sólo se debería mantener aquella información de traza para la que exista un uso claro. La trazabilidad de los RQ se puede representar de diversas maneras. Algunas formas típicas son:

- Referencias basadas en texto e hipervínculos
- Matrices de trazabilidad
- Grafos de trazabilidad



Microsoft Partner

Gold Cloud Platform
Silver Cloud Productivity
Silver Collaboration and Content
Silver Application Development
Silver Small Business



ORACLE
Gold Partner

IV. Ingeniería de requerimiento

10. Trazabilidad de requisitos

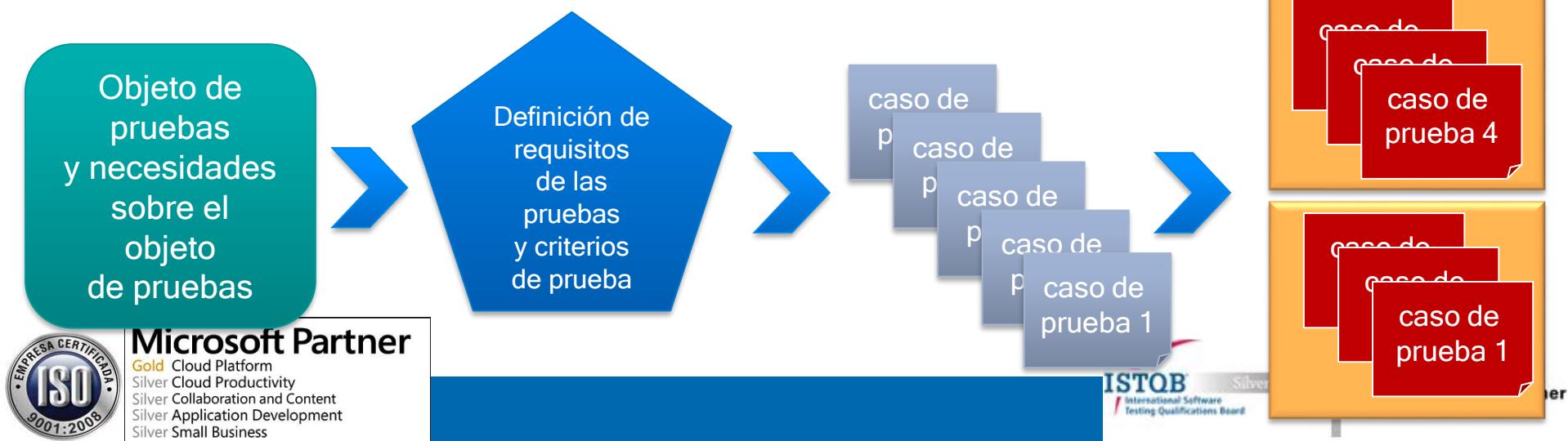
Trazabilidad (4/5)

Obtención de casos de prueba a partir de requisitos

- El diseño de casos de prueba debe ser un proceso controlado

Los casos de prueba pueden ser creados **formal o informalmente**, dependiendo de las características del proyecto y la madurez del proceso en uso

Escenarios de Pruebas



IV. Ingeniería de requerimiento

10. Trazabilidad de requisitos

Trazabilidad (5/5)

Obtención de casos de prueba a partir de requisitos

Las pruebas deben ser trazables:

¿Qué casos de prueba han sido incluidos en el catálogo de pruebas, basados en qué requisitos?

Las consecuencias de los cambios en los requisitos sobre las pruebas a realizar pueden ser identificadas directamente

La trazabilidad Ayuda a determinar la cobertura de requisitos

Permite establecer con seguridad resultados esperados para un caso o escenario de prueba.

Escenarios de Pruebas

