Automatización y gestión de pruebas del software

Macario Polo Usaola
Grupo Alarcos
Departamento de Tecnologías y Sistemas de Información
Universidad de Castilla-La Mancha

Macario.Polo@uclm.es

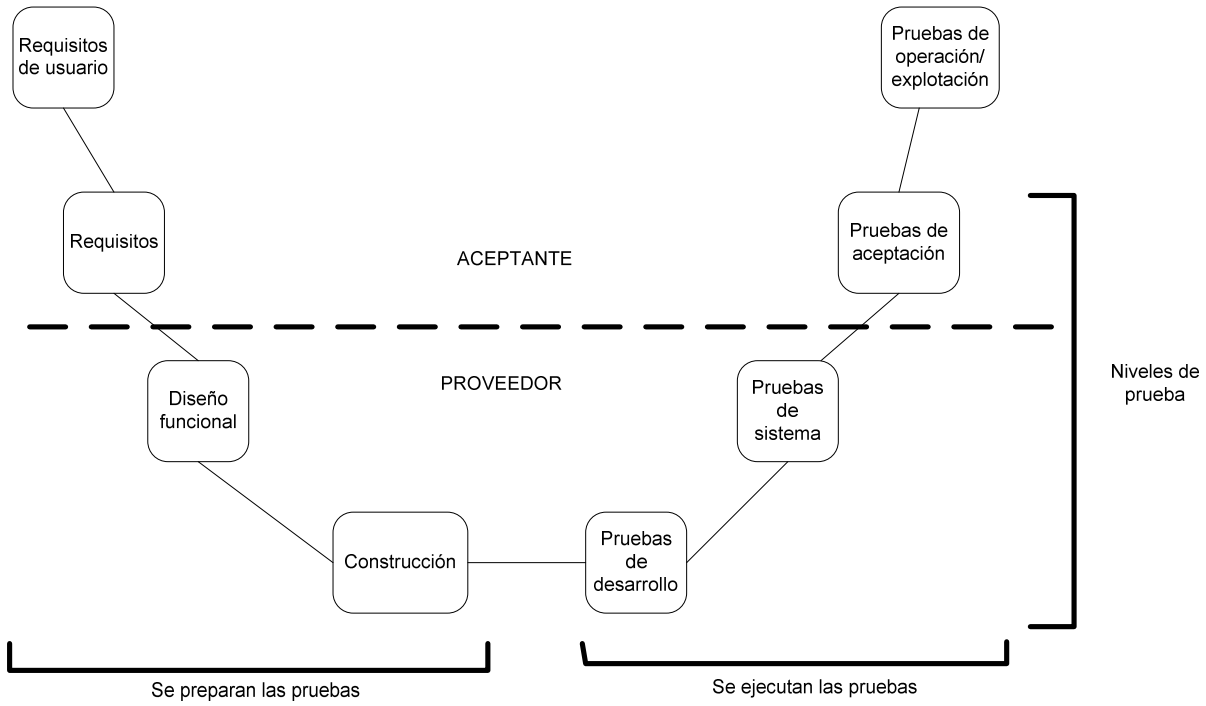
Contenidos

- Introducción
- Generación de casos de prueba
- Metodologías/modelos de proceso
  - MTPF
  - TMAP

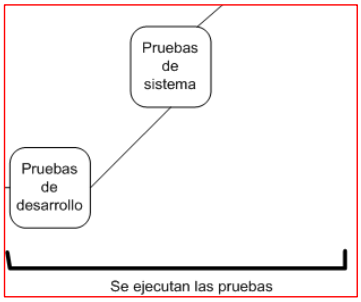
Objetivo de las pruebas

- Encontrar errores en el SUT (*System Under Test*)
- Para ello, intentamos ejecutar el SUT de “todas” las maneras posibles.
- De alguna manera, tenemos que poder medir “cuánto sistema” se ha ejecutado
- Conceptos importantes:
  - Criterios de cobertura
  - Valores de prueba (*test data)*
  - Ejecución de servicios
  - Combinación de valores
  - Existen, además, diferentes “niveles de prueba”

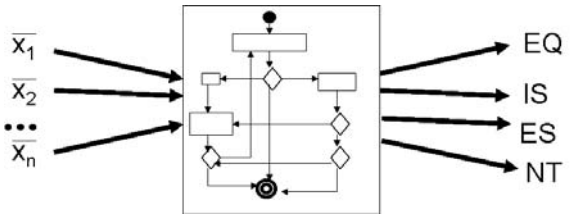
Niveles de prueba



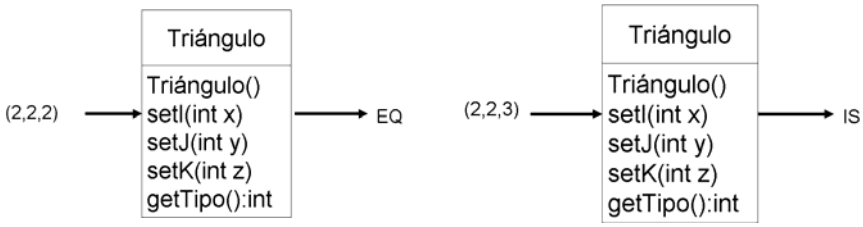
Pruebas de desarrollo y de sistema



- Caja blanca



- Caja negra

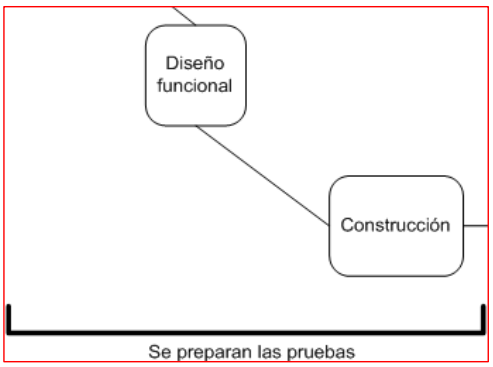


Criterios de cobertura

- Sentencias
- Decisiones
- Condiciones
- Todos los usos
- Caminos
- Funciones
- Llamadas
- Bucles
- Mutantes
- …

- Casos de uso
- Escenarios
- Carga máxima
- Valores de prueba
- …

Diseño de pruebas: valores de prueba o valores “interesantes”



- Clases de equivalencia
- Valores límite
- Conjetura de errores
- Experiencia del *tester*

Se proponen valores y, luego, se mezclan o combinan de alguna manera

Líneas de automatización

- Tareas de generación de casos
  - A partir del código
  - A partir del conocimiento del programa
  - A partir de una especificación formal
- Tareas mecánicas
  - Ejecución y control
  - Facilidades de reejecución
- Tareas administrativas
  - Registro de especificaciones
  - Obtención de informes de resultados

Generación de casos

- Los valores de prueba se combinan para obtener buenos casos de prueba
- Estrategias de combinación:
  - Todas las combinaciones (*all combinations)*
  - Each choice
  - Pairwise
  - n-wise
  - Random
  - Antirandom

Estrategias de combinación (I)

- Supongamos {0, 1, 2, 3, 4, 5, 6} como valores de prueba para los tres lados del triángulo
- *All combinations*: 7x7x7=343 casos de prueba:
  - (0,0,0), … (1,1,1), …(1,2,4), … (2,2,2), …

Estrategias de combinación (II)

- *Each choice*: Se satisface cuando cada valor interesante de cada parámetro se incluye, al menos, en un caso de prueba.

  - (0, 1, 2)→ con este, en los próximos casos no sería necesario probar con el 0 en la primera posición, ni con el 1 en la segunda, ni con el 2 en la tercera
  - (1, 0, 2)
  - (2, 3, 4)
  - (2, 4, 3)

  •(3, 5, 0)
  •(4, 2, 1)
  •(3, 0, 5)→ este caso vuelve a emplear el 0 en la segunda posición; sin embargo, es necesario utilizarlo para probar con el 5 en la tercera posición
  •(5, 0, 1)

Estrategias de combinación (III)

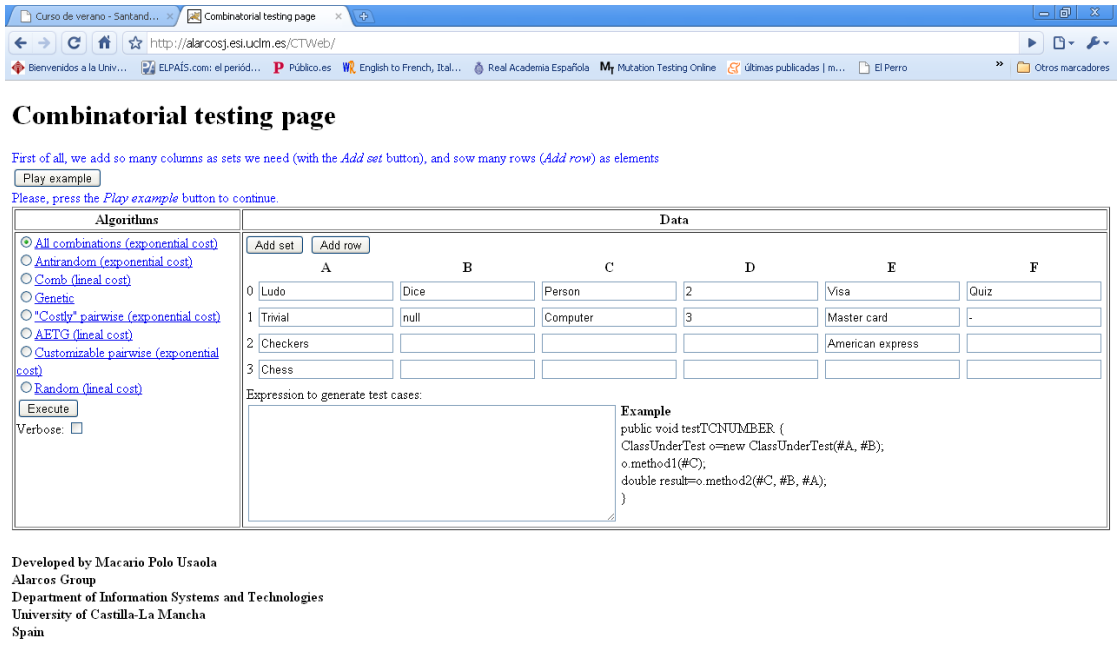
- *Pairwise* (cobertura de pares de valores). Se utiliza porque muchos errores aparecen gracias a la interacción de dos valores de parámetros. El criterio requiere que cada posible par de valores interesantes de cualesquiera dos parámetros sea incluido en algún caso de prueba.

Estrategias de combinación (y IV)

- *Random*: se generan casos de prueba aleatoriamente. Puede utilizarse algún tipo de distribución de probabilidad.
- *Antirandom*: se elige un caso de prueba al azar y se incluye en el *suite*; se añade el que sea más diferente; se añade el que sea más diferente a los dos ya elegidos; luego, el que sea más diferente a los tres; luego…

Demostración

- Página de testing combinatorio:
  - http://alarcosj.esi.uclm.es/CTWeb



Redundancia de casos de prueba (I)

- En el ejemplo del triángulo, probablemente (3,3,3) sea redundante respecto de (2,2,2)
- La generación y ejecución de pruebas tiene un coste, por lo que los casos redundantes son molestos

Redundancia de casos de prueba (II)

- Un caso de prueba puede ser redundante respecto de un criterio de cobertura, pero no respecto de otro:

```
if (i+j<=k || j+k<=i || i+k<=j) {
    tipo=Triangulo.NO_TRIANGULO;
    return tipo;
} else {
    tipo=Triangulo.ESCALENO;
    return tipo;
}
```

(1,2,3) y (2,4,2) son redundantes respecto de *sentencias*, pero no respecto de *condiciones*

Redundancia de casos de prueba (y III)

- Los casos de prueba redundantes se eliminan bien durante la ejecución, bien después de ejecutarlos todos, con un algoritmo voraz.
- Así nos quedamos con un conjunto equivalente y reducido, para las pruebas de regresión

	tc1	tc2	tc3	tc4	tc5	tc6
m1	X	X				
m2	X	X			X	
m3		X				X
m4			X			X
m5			X			X
m6			X			X
m7						X

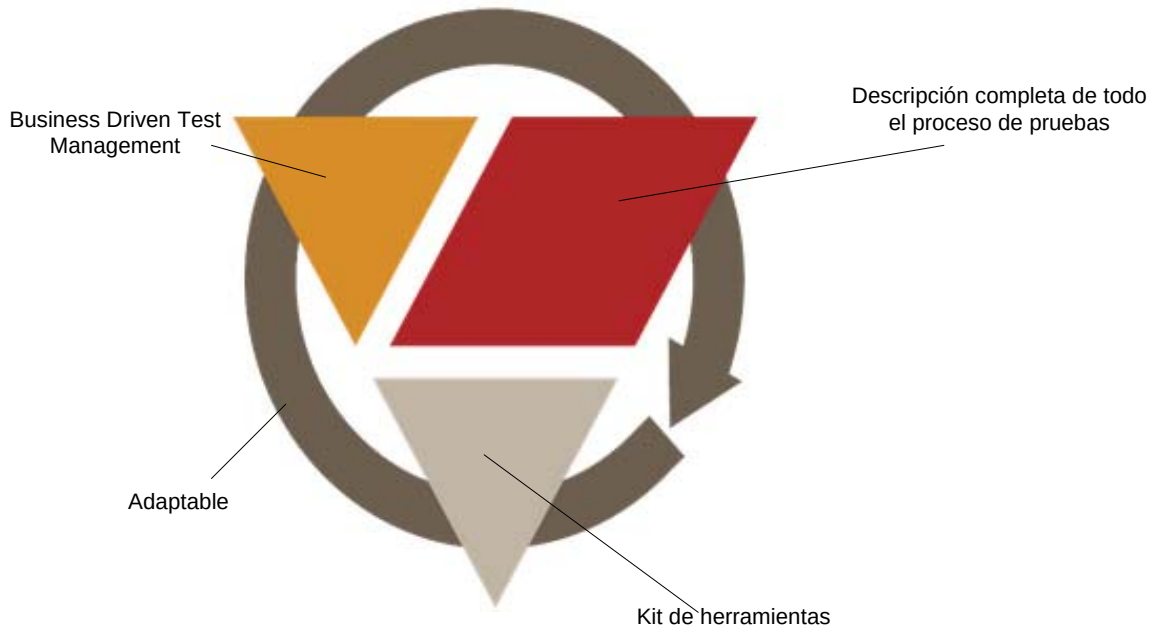
El *Minimal Test Practice Framework* (I)

- Proceso de mejora para PYMEs.
- Se estructura en cinco tipos de prácticas y tres niveles de madurez.
- El objetivo del MTPF es pasar de una estrategias de pruebas “ad hoc” a un proceso de pruebas claro y estructurado.

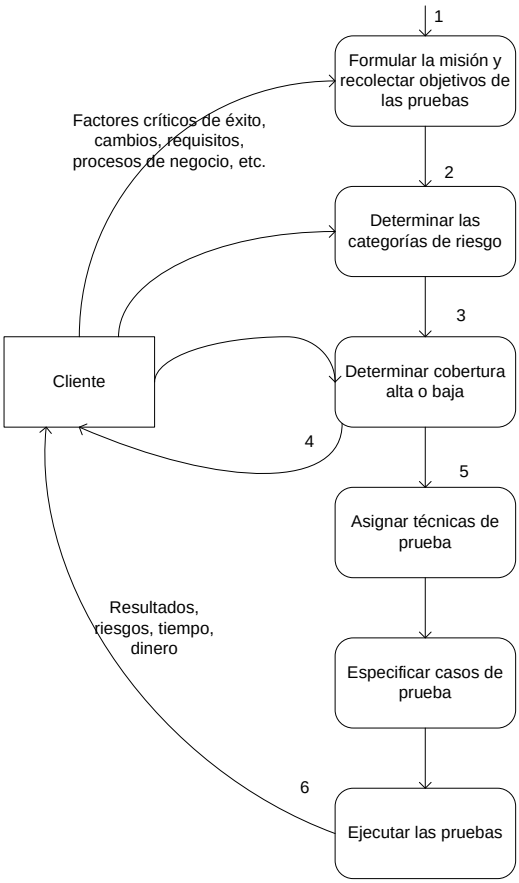
El *Minimal Test Practice Framework* (y II)

Fase 3 (~30 personas)	**Mantenimiento del sistema** Mantener el sistema para que se adapte a la evolución de la compañía	**Definición de equipos** Creación de un equipo de pruebas independiente del de desarrollo. Los miembros de este equipo pueden especializarse (usabilidad, seguridad, etc.).	**Inspecciones** Sustitución progresiva los walkthroughs por inspecciones que, al requerir preparación previa, resultan más eficientes. Se definirán los roles de cada inspector.	**Gestión de riesgos** Se creará y mantendrá una base de datos de problemas y experiencias que ayude a predecir las áreas de mayor riesgo de los proyectos antes de que acontezcan.	**Coordinación del aseguramiento de la calidad** Establecimiento de rutinas de aseguramiento de la calidad, de forma que se asegure que el software no será entregado antes de que alcance el nivel mínimo predeterminado.
Fase 2 (~20 personas)	**Creación del sistema** Introducción de sistema de recogida y almacenamiento de problemas de acuerdo a estándares de Fase 1. Definición de procedimientos para recolectar, documentar, almacenar y reutilizar conocimiento de cada proyecto.	**Definición de roles** Asignar los roles de responsable de pruebas y de ingenieros de pruebas (*testers*). Las responsabilidades de los *testers* son: gestión de walkthroughs, gestión de, desarrollo de casos de prueba; gestión del sistema de recogida de problemas; gestión de experiencia.	**Walkthroughs** Realización de walkthroughs antes de que el software esté preparado para su ejecución, idealmente en la fase de diseño. El equipo de walkthroughs se compone de desarrolladores y diseñadores, al que puede asistir un *tester*.	**Casos de prueba** Se construyen casos de prueba para comprobar que se prueban las situaciones y acciones más comunes. A la creación de casos de prueba se le asignarán varios *testers*, ya que es una de las actividades que requiere más tiempo. Se crearán varios casos para cada escenario.	
Fase 1 (~10 personas)	**Definición de estándares de registro** Definición de terminología, lenguaje, procedimientos, etc.: • Campos que, juntos, describan el problema • Que sigan el flujo del *tester* • Fácil de entender • Que identifique cada problema unívocamente • Que permita agrupar problemas	**Definir responsabilidades** Las responsabilidades son: desarrollo de planes de pruebas para cada proyecto; gestión del entorno de pruebas; gestión del sistema de recogida de problemas; actualización de checklists; evaluación de prácticas; control de necesidades para la siguiente fase	**Uso de checklists** Creación de checklists, o revisión de las que pudieran estar siendo utilizadas.	**Gestión básica** El entorno de pruebas debe estar disponible siempre que se requiera. Las actividades básicas son: organizar el entorno de pruebas para cada proyecto, mantenerlo actualizado y documentar su forma de uso.	**Plan de pruebas** El plan de pruebas permite recoger en un solo documento todo lo relacionado con las pruebas de cada proyecto (p.ej., el IEEE Std. for Soft. Test Documentation). Es importante indicar los diferentes hitos en el plan de pruebas.
Fases / Categorías	Registro sistemático de defectos	Definición de roles y organización	Verificación y validación	Gestión de las pruebas	Planificación

Metodología TMAP



BDTM



- El elemento clave es determinar qué vamos a probar y con qué profundidad

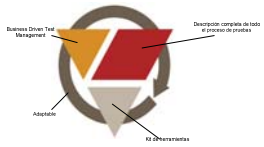
Business Driven Test Management

BDTM

Business Driven Test Management

- Da control del proceso de pruebas al cliente
- Utiliza el lenguaje del cliente
- El esfuerzo de pruebas está relacionado con los riesgos del producto
- Proporciona la cobertura adecuada en cada lugar
- Da visibilidad a los resultados de las pruebas

Kit de herramientas



Kit de herramientas

- El kit de herramientas incluye:
  - Técnicas (cómo *testear*)
  - Infraestructura (dónde y con qué *testear*)
  - Organización (quién *testea*)

Kit.Técnicas

- Análisis de riesgos del producto
- Estimación de esfuerzos
- Gestión de defectos
- Métricas
- Diseño de pruebas
- Revisión

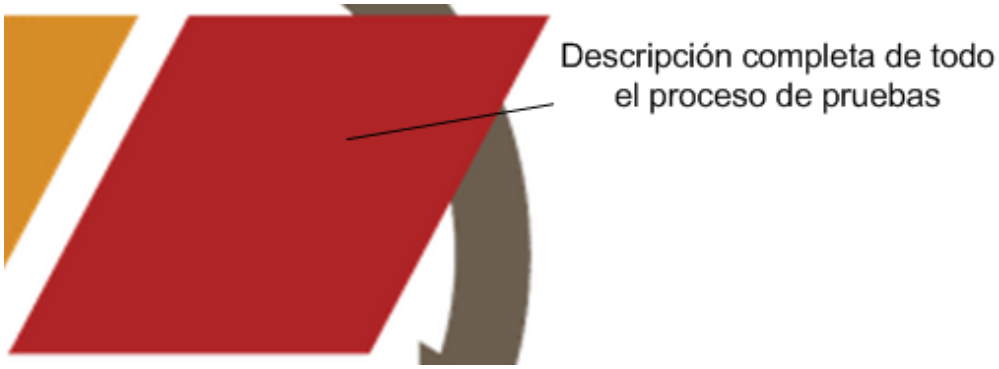
Kit.Infraestructura

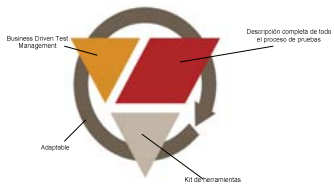
- Entorno de pruebas
  - Entorno físico “DTAP” (desarrollo; test; aceptación; producción)
  - Elementos auxiliares: BB.DD.; SS.OO.; interfaces; datos; aplicación
  - Requisitos del entorno: seguridad, estándares
- Herramientas de test
- Lugares de trabajo

Kit.Organización

- Política de pruebas
- Organización permanente de pruebas
- Organización de pruebas en proyectos
- Profesionales de pruebas/Roles

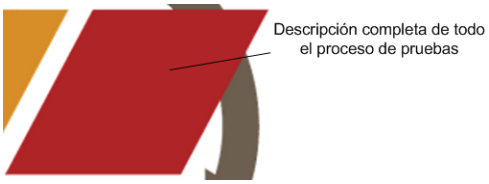
Descripción completa de todo el proceso de pruebas





Descripción completa de todo el proceso de pruebas

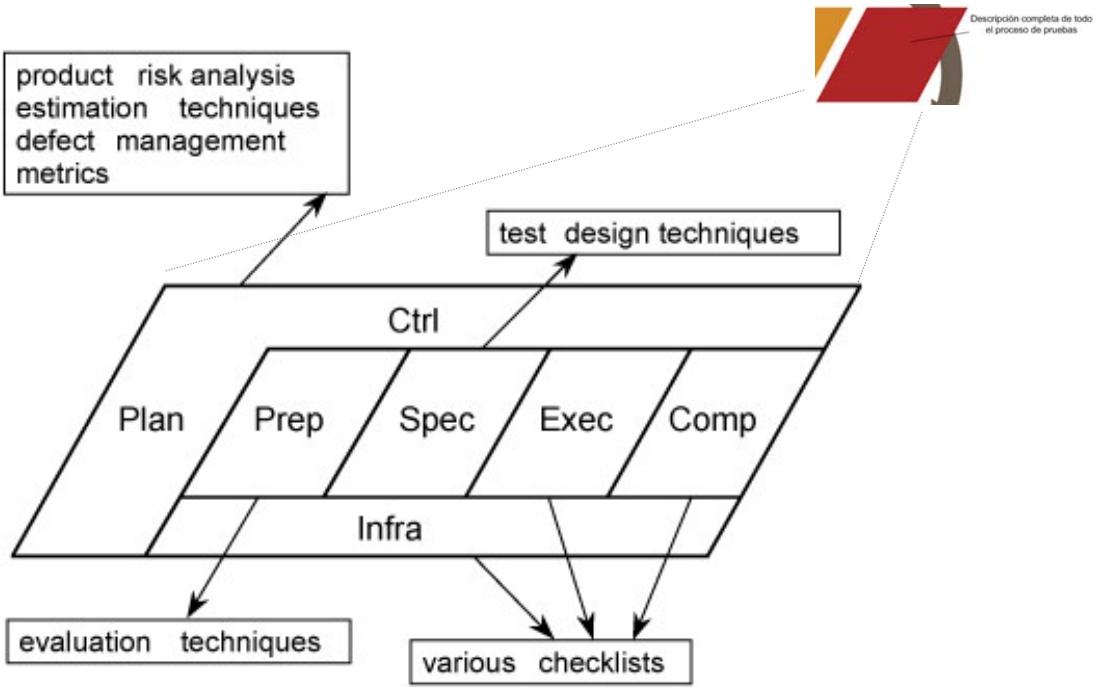
- Fases de TMAP:
  - Máster plan de pruebas, gestión total del proceso de pruebas
  - Pruebas de aceptación y sistema
  - Pruebas de desarrollo
  - Procesos de soporte



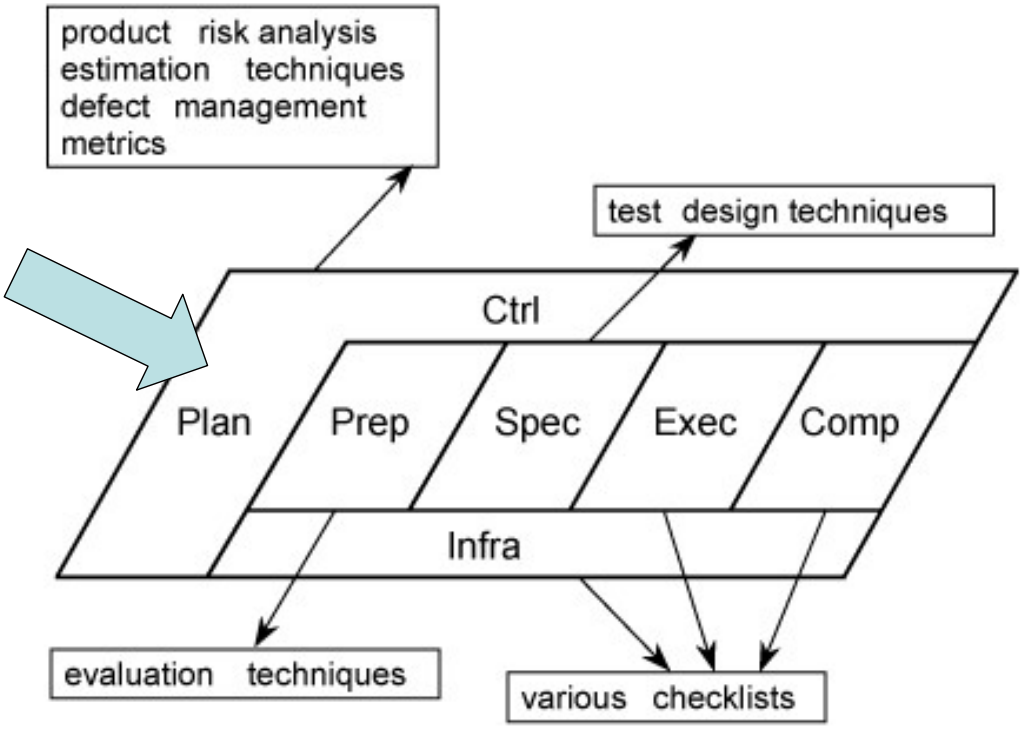
Descripción completa de todo el proceso de pruebas

- Las fases anteriores:
  - Son aplicables a todos los niveles de pruebas
  - Permiten coordinar y sincronizar los diferentes niveles de pruebas
  - Transmiten tareas y responsabilidades entre grupos involucrados
  - Dividen el proceso de test en fases, actividades y productos

Máster plan de pruebas, gestión total del proceso de pruebas



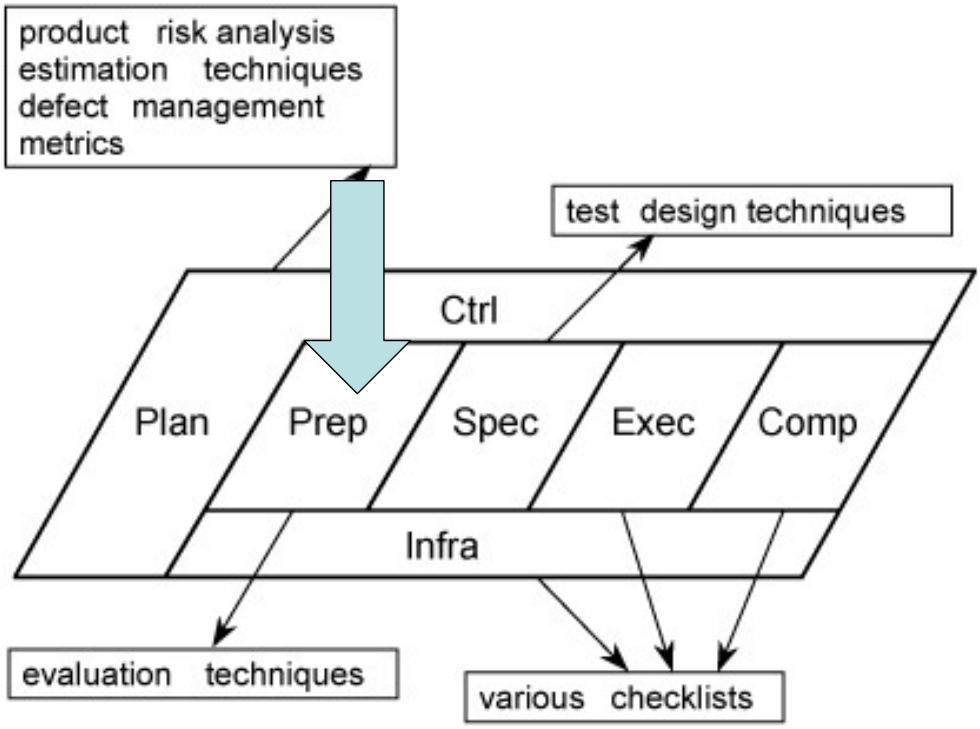
Fase de Planificación



Planificación.Actividades

1. Establishing the assignment (Establecer la misión)
2. Understanding the assignment (Entender la misión)
3. Determining the test basis (Determinar la **base de test**)
4. Analysing the product risks (Analizar los riesgos del producto)
5. Determining the test strategy (Determinar la estrategia de pruebas)
6. Estimating the effort (Estimar esfuerzo)
7. Determining the planning (Planificar el proyecto)
8. Allocating test units and test techniques (Recolectar uds. y técnicas)
9. Defining the test products (Definir los productos de test)
10. Defining the organisation (Definir la organización)
11. Defining the infrastructure (Definir la infraestructura)
12. Organising the management (Organizar la gestión)
13. Determining the test project risks and countermeasures (Determinar riesgos y contramedidas)
14. Feedback and consolidation of the plan (Realimentar y consolidar el plan)

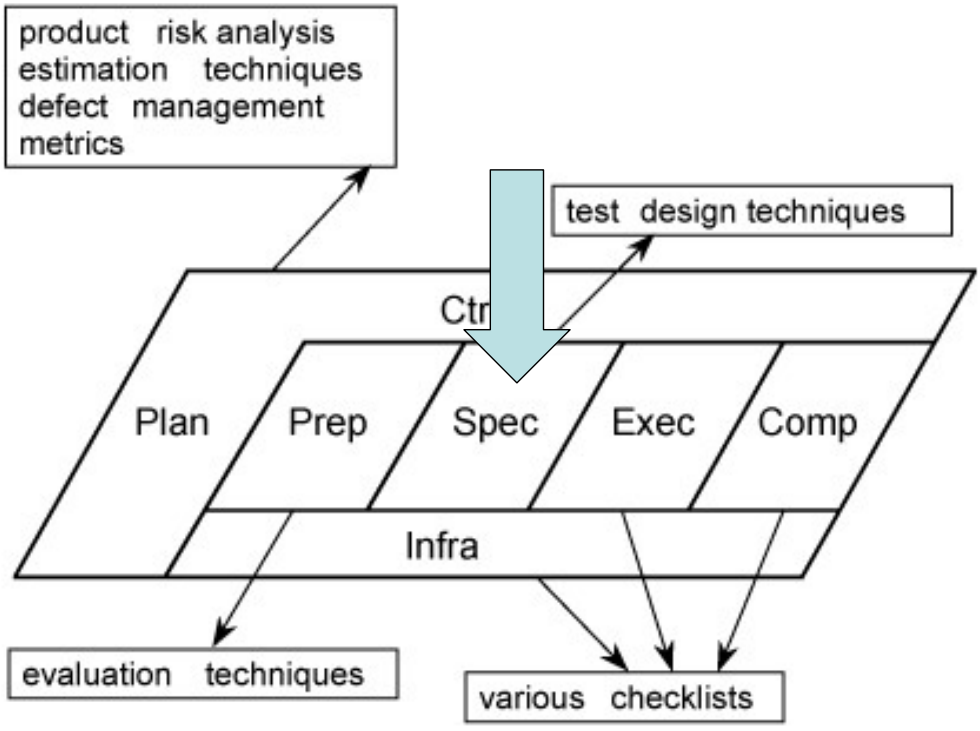
Fase de Preparación



Preparación.Actividades

1. Collection of the **test basis** (Recolectar la **base de test**)
2. Creating checklists (Crear listas de comprobación)
3. Assessing the test basis (Evaluar la base de test)
4. Creating the testability review report (Crear informe de ***testability***)

Fase de Especificación



Especificación.Actividades

1. Creating test specifications (Crear las especificaciones de prueba)
2. Defining **central starting point**(s) (Definir situaciones iniciales)
3. Specifying the **test object intake** (especificar la "**prueba de humo**")

Especificación.Crear las especificaciones de prueba

- Objetivo: definir los casos de prueba para cada unidad que se va a probar
- Método:
  - Identificar **situaciones de prueba**
  - Crear **casos de prueba lógicos**
  - Crear **casos de prueba físicos**
  - Determinar las **situaciones iniciales**
  - Definir el guión de pruebas (***test script***)
- Técnicas: diseño de pruebas
- Herramientas: gestión de defectos, de diseño de pruebas, de gestión de ***testware***, de automatización, de rendimiento, de carga…

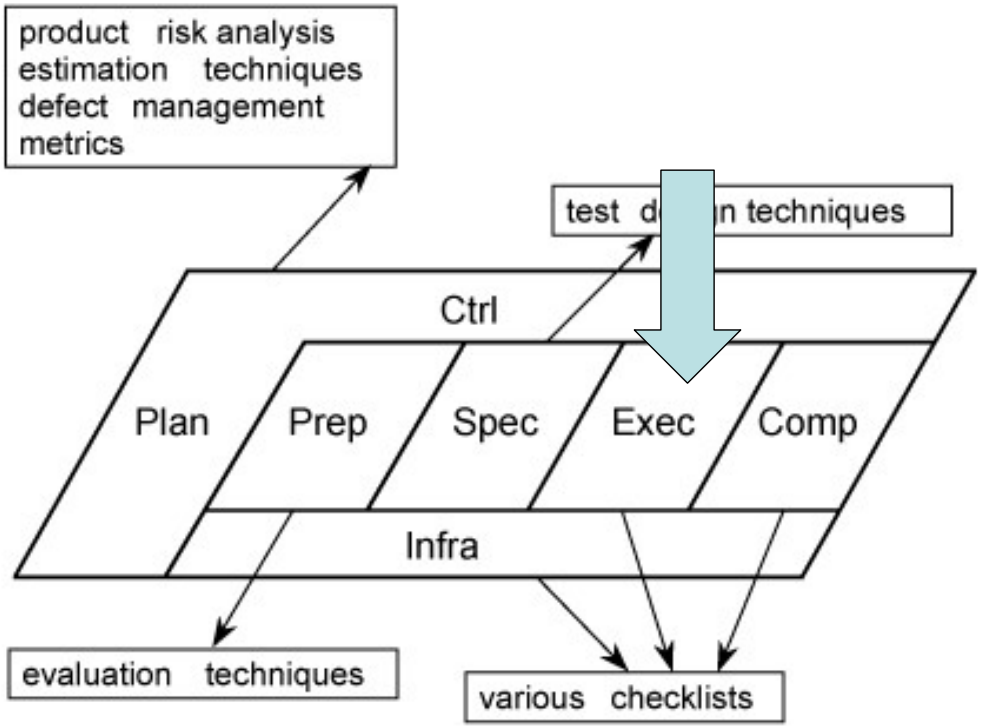
Especificación.Definir situaciones iniciales

- Objetivo: definir una o más situaciones iniciales del entorno de test para obtener o crear datos de prueba para las especificaciones de los casos de prueba
- Método:
  - Establecer situaciones iniciales
  - Identificar datos de prueba
  - Añadir datos de prueba
- Salida: descripción de situaciones iniciales
- Técnicas: no aplicable
- Herramientas: gestión de *testware*

Especificación.Especificación de la prueba de humo (*object intake)*

- Objetivo: preparar la validación del objeto de test, para asegurar que la ejecución de las pruebas puede empezar lo antes posible (o sea, que el sistema por lo menos parezca que funciona)
- Método:
  - Crear una lista de control para validar el objeto de prueba
  - Definir el plan de la “prueba de humo”
- Técnicas: no aplicable
- Herramientas: gestión de *testware*, de datos, de diseño de pruebas, de ejecución de pruebas

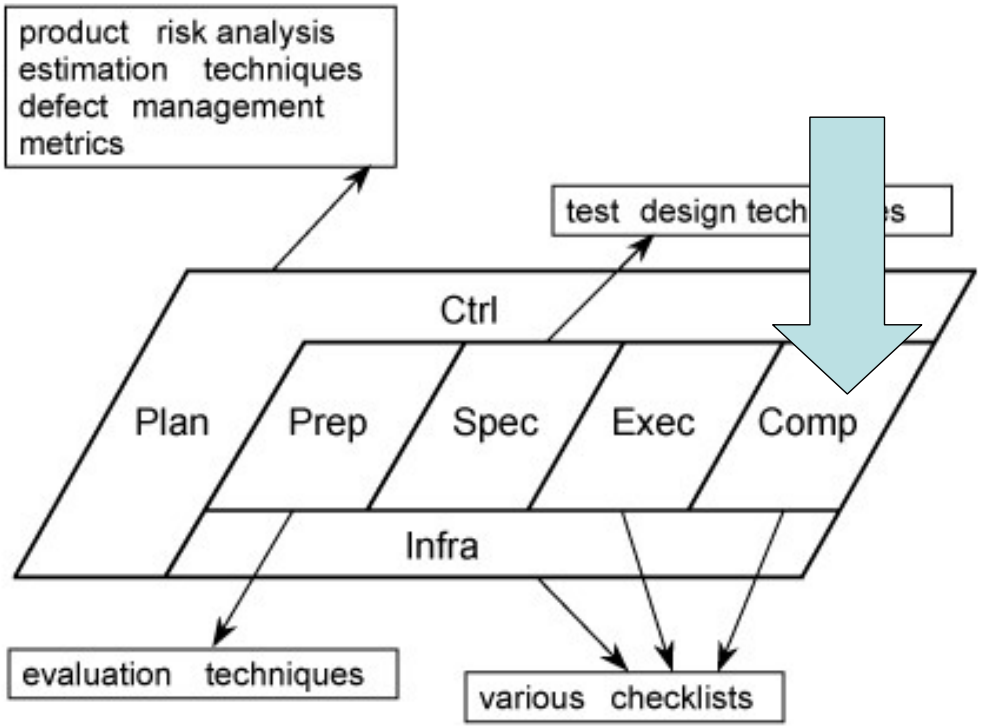
Fase de Ejecución



Ejecución.Actividades

1. **Intake** of the test object (Recepción del objeto de pruebas)
2. Preparing the starting points (Preparación de situaciones iniciales)
3. Executing the (re)tests (Ejecutar las pruebas)
4. Checking and assessing the test results (Comprobar y evaluar los resultados)

Fase de Finalización (Completion)



Finalización.Actividades

1. Evaluating the test process (Evaluación del proceso de pruebas)
2. Preserving the testware (Conservación del testware)

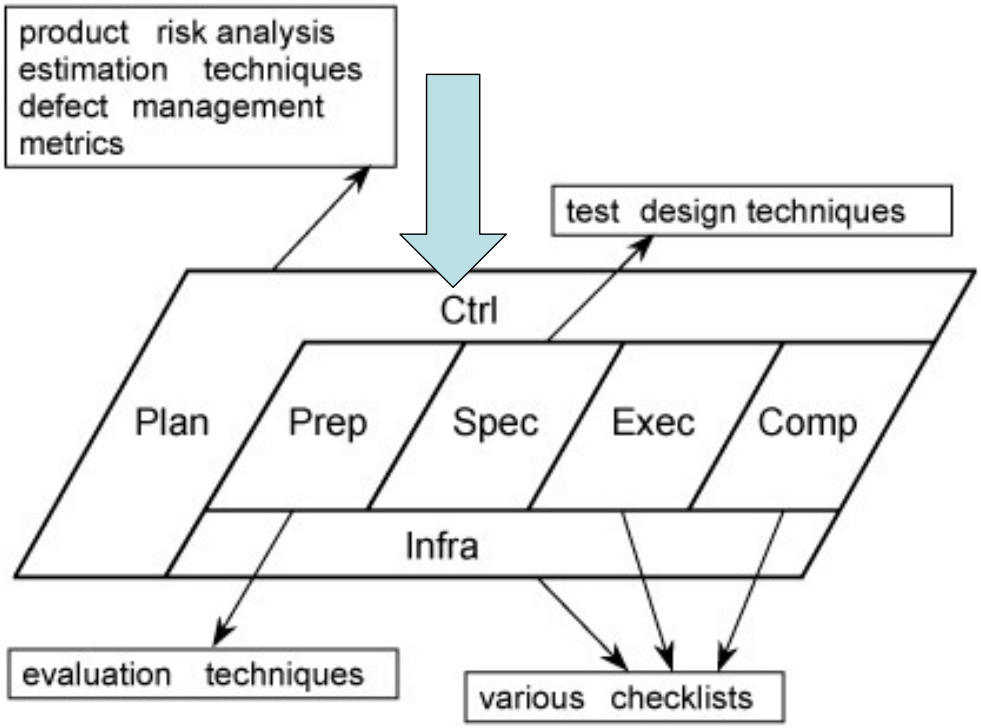
Finalización. Evaluación del proceso de pruebas

- Objetivo: aprender de la experiencia del proyecto e identificar las buenas prácticas que deben consolidarse y los puntos de mejora para el futuro
- Método:
  - Reunión de equipo
  - Realización de informe
- Se obtiene un “informe de evaluación del proceso”
- Técnicas: Lista de control “Evaluación del proceso de pruebas”

Finalización.Conservar el testware

- Objetivo: transferir el testware para que sea usable en el futuro
- Método:
  - Seleccionar el testware que se conservará
  - Refinar, completar y asegurar su accesibilidad
  - Transferir el testware
- Se obtienen:
  - Testware
  - Inventario de testware
- Técnicas: no aplicables

Fase de Control



Control.Actividades

1. Management (Gestión)
2. Monitoring (Seguimiento)
3. Reporting (Informes)
4. Adjusting (Reajustar)

Técnicas de especificación de pruebas

Qué es una Técnica de Especificación de Pruebas

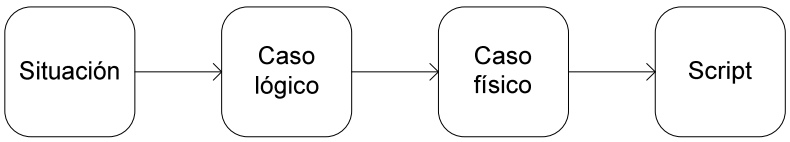
- Es un método estandarizado para derivar casos de prueba a partir de una determinada test basis, de manera que se alcance una cobertura específica

- ***Cobertura:*** ratio entre lo que puede ser probado y lo que se prueba realmente con los casos de prueba
- ***Caso de prueba***: se utiliza para examinar si el sistema exhibe el comportamiento deseado bajo unas circunstancias bien determinadas.

Técnica de Especificación de Pruebas

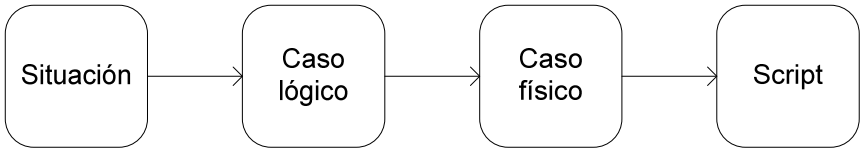
- Se utilizan en la Fase de Preparación (tarea 2, Crear especificaciones de prueba):
  - Identificar **situaciones de prueba**
  - Crear **casos de prueba lógicos**
  - Crear **casos de prueba físicos**
  - Determinar las **situaciones iniciales**
  - Definir el guión de pruebas (***test script***)
- Permiten:
  - Implantar la estrategia de pruebas
  - Detectar defectos de forma efectiva
  - Reproducir las pruebas
  - Disminuir la dependencia entre quien especifica y quien ejecuta
  - Controlar calidad
  - Transferir y mantener
  - Planificar y controlar

Conceptos fundamentales



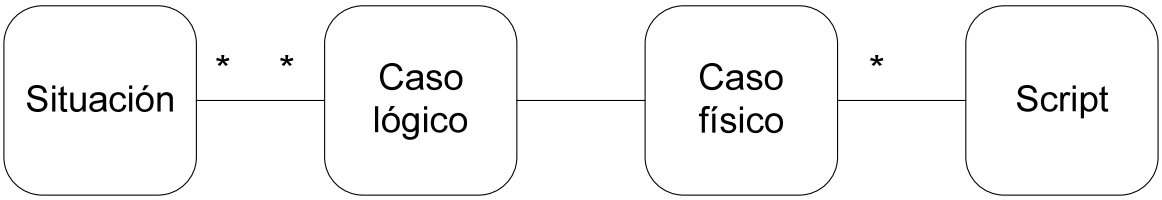
- Situación de prueba:
  - Situación aislada bajo la cual el objeto de test muestra un comportamiento específico que necesita ser probado
  - Cada situación ocurre por lo menos en un caso lógico
- Caso lógico:
  - Describe en términos lógicos las circunstancias en las que se examina el comportamiento del sistema. Se indican las situaciones de prueba que cubre este caso
  - O sea, un caso lógico cubre una o más situaciones de prueba

Conceptos fundamentales



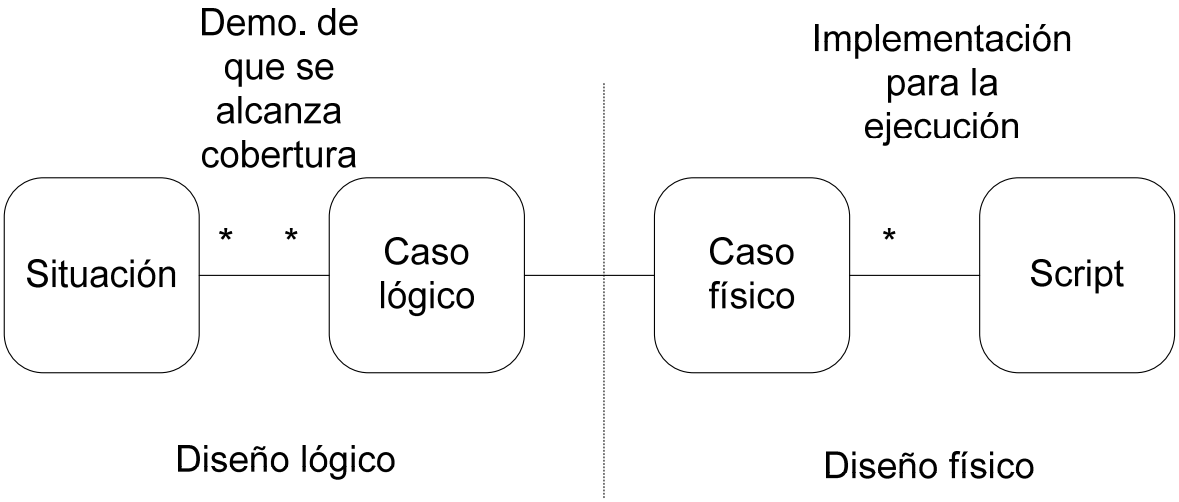
- Caso físico:
  - Elaboración concreta de un caso lógico, con la asignación de valores reales para las entradas y determinación exacta del entorno
  - O sea, que incluye: datos de entrada; configuración del entorno; acciones; resultados esperados
- Script:
  - Combina uno o más casos físicos

Relaciones



- Cada situación debe estar al menos en un caso
- Cada caso cubre una o más situaciones
- Cada caso lógico se traduce a un caso físico
- El script puede englobar varios casos físicos

Relaciones



- ***Cobertura:*** ratio entre lo que puede ser probado y lo que se prueba realmente con los casos de prueba

Ejemplo

- Una tienda virtual:
  - Un pedido de más de un libro
  - Un pedido de exactamente un libro

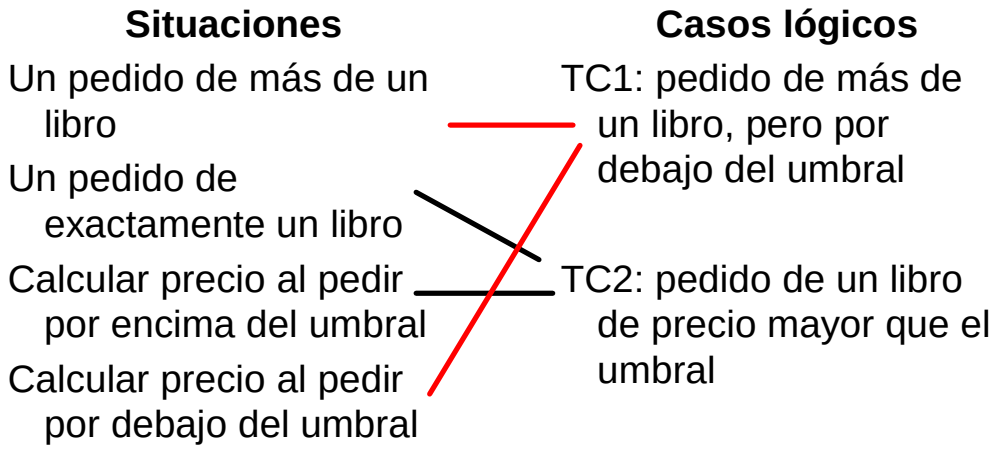
  Si además hay descuento cuando se supera un cierto umbral de precio total, tenemos:
  - Calcular precio al pedir por encima del umbral
  - Calcular precio al pedir por debajo del umbral

  Arriba tenemos cuatro **situaciones de prueba**.

Ejemplo

- A partir de esas cuatro situaciones tenemos, p.ej., dos casos lógicos:
  - TC1: pedido de más de un libro, pero por debajo del umbral
  - TC2: pedido de un libro de precio mayor que el umbral

Ejemplo



Ejemplo

- Casos físicos:
  - Umbral: 50 €
  - Descuento: 10%
  - Precios de tres libros: 18,50; 25,50; 65,00
  - TC1 (más de un libro, por debajo de umbral):
    - Pedir libros 1 y 2
    - No esperamos descuento
    - Precio del pedido: 44 €
  - TC2 (un libro, por encima);
    - Pedir libro 3
    - Esperamos 10% de descuento
    - Precio del pedido: 58,50 €

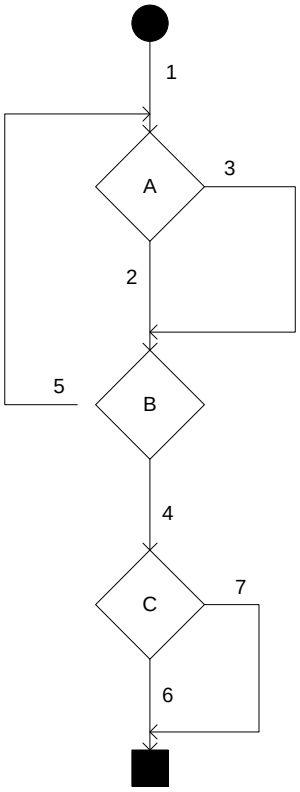
Sobre la cobertura

- ***Cobertura:*** ratio entre lo que puede ser probado y lo que se prueba realmente con los casos de prueba
- ***Tipo de cobertura:*** forma en la que se expresa la cobertura de las situaciones de prueba
- ***Ratio de cobertura:*** porcentaje de situaciones de prueba, según el *tipo de cobertura*, que cubre la prueba

Tipos de cobertura importantes

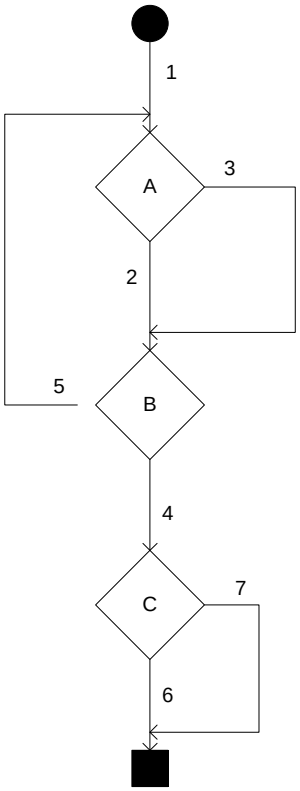
- Caminos
- Puntos de decisión
- Clases de equivalencia
- Pairwise
- Arrays ortogonales (otra forma de pairwise)
- Valores límite
- CRUD
- Perfiles operacionales (simulación del uso real del sistema)
- Perfiles de carga (simulación de carga real del sistema en cuanto a usuarios y transacciones)
- Caminos correctos y caminos de fallo
- Listas de comprobación

Cobertura de caminos



- Situaciones ≈ cada camino individual (1,2,3,4,5,6,7)
- Nivel de cobertura: pasar una vez por cada camino:
  - Casos lógicos: 1-2-5-3-4-6 y 1-3-4-7
- Nivel de cobertura: pares E/S de cada ***punto de decisión***:
  - A: (1,2), (1,3), (5,2), (5,3)
  - B: (2,4), (2,5), (3,4), (3,5)
  - C: (4,6), (4,7)

Cobertura de caminos



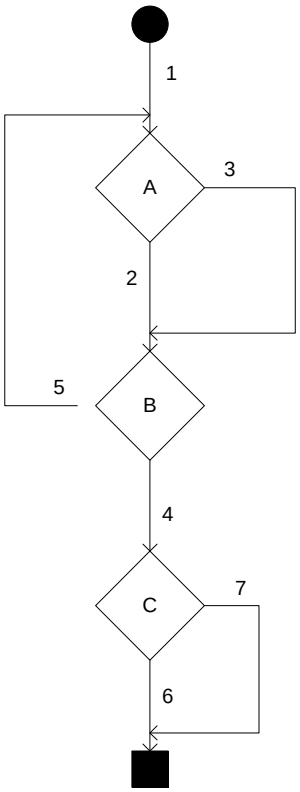
- A: (1,2), (1,3), (5,2), (5,3)
- B: (2,4), (2,5), (3,4), (3,5)
- C: (4,6), (4,7)
- Casos lógicos:
  - 1-2-4-6
  - 1-3-5-3-4-7
  - 1-2-5-2-4-6

A: (1,2), (1,3), (5,2), (5,3)
B: (2,4), (2,5), (3,4), (3,5)
C: (4,6), (4,7)

- Otra solución:
  - 1-2-5-3-4-6
  - 1-3-5-2-4-7

A: (1,2), (1,3), (5,2), (5,3)
B: (2,4), (2,5), (3,4), (3,5)
C: (4,6), (4,7)

Cobertura de caminos



- El nivel de profundidad (hemos visto 1 y 2), puede aumentarse a 3, 4, etc.
  - Para conseguirlo, se construirían las ternas, “cuaternas”, etc.
  - De forma general:

  Nivel(n+1)=Nivel(n)+ “1 paso más en el recorrido del proceso”
  - Ternas:
    - A: 1-2-4, 1-2-5, 1-3-4, 1-3-5, 5-2-4, 5-2-5, 5-3-4, 5-3-5
    - Etc.

Tabla-guía de técnicas de diseño (I)

Técnica	Tipo de cobertura	Test basis	Característica de calidad
Tablas de decisión	MCC	Condiciones individuales	Funcionalidad detallada
Combinaciones de datos	Clases equiv. con MCC o PW	Cualquiera	Funcionalidad detallada y general
Comparación elemental	MCDC	Pseudocódigo o espec. Estructuradas	Funcionalidad detallada
Conjetura de errores	—	Cualquiera	Varias
Testing exploratorio	Depende	Cualquiera	Varias
Ciclo de datos	CRUD y Decisiones	Matriz CRUD Reglas de integridad	Func. General, conectividad, adecuación

Tabla-guía de técnicas de diseño (y II)

Técnica	Tipo de cobertura	Test basis	Característica de calidad
Ciclo de procesos	Pares de caminos	Descripción estructurada de procesos	Adecuación
Vida real	Simulación estadística	Perfiles operacionales; perfiles de carga	Efectividad, conectividad, rendimiento
Prueba semántica	MCDC	Especificaciones de E/S; reglas de negocio	Funcionalidad, validación
Prueba sintáctica	Checklist	Especificaciones de E/S; descripciones de atributos	Funcionalidad, validación, facilidad de uso
De casos de uso	Checklist	Casos de uso	Adecuación, efectividad, facilidad de uso

Métricas (I)

- Horas de test por fase y actividad
- Entregables
- Progreso de defectos
- Productividad:
  - Horas de test/defecto
  - Nº de tests por hora
  - Nº de especificaciones por hora
  - Nº de defectos/script
  - Nº de test/defecto
  - Distribución de horas por fase de TMAP
  - Nº de test/PF
  - Nº de test/KLOC
  - Horas de test/PF o KLOC

Métricas (y II)

- Nº de horas/página de documentación
- Nº de defectos/página de documentación
- Nº de casos/página
- Nº de páginas/PF
- Porcentaje de defectos detectados

Contenidos

- Introducción
- Generación de casos de prueba
- Metodologías/modelos de proceso
  - MTPF
  - TMAP

Automatización y gestión de pruebas del software

Macario Polo Usaola
Grupo Alarcos
Departamento de Tecnologías y Sistemas de Información
Universidad de Castilla-La Mancha

Macario.Polo@uclm.es