TEMA 4. Estrategias de prueba del Software (1/2)

Calidad del Software

Dr. José Luis Abellán Miguel

Grado en Ingeniería Informática

Dr. José Luis Abellán Miguel - Tlf: (+34) 968 278 657
Universidad Católica San Antonio de Murcia www.ucam.edu

Índice

- Introducción
- Enfoque Estratégico para la Prueba del SW
- Estrategia de Prueba para SW convencional

Bibliografía

- Pressman, R. ***Ingeniería del Software: Un enfoque práctico***. 7ª edición. Madrid: McGraw Hill, 2010. ISBN: 9701054733 (disponible en la biblioteca UCAM) → **Capítulo 17**

Introducción (1/3)

- Prueba de SW
  - Descubrir errores que se cometieron de manera inadvertida conforme se diseñó y construyó el SW
- **Estrategia de prueba de SW (EPSW):**
  - Guía que describe los pasos que deben realizarse como parte de la prueba, cuándo se planean y se llevan a cabo dichos pasos, y cuánto esfuerzo, tiempo y recursos se requerirán
  - Elementos de la EPSW:
    - Planificación de la prueba
    - Diseño de casos de prueba
    - Ejecución de la prueba
    - Recolección y evaluación de los resultados

Introducción (2/3)

- ¿Quién hace la EPSW?
  - El gerente de proyecto, los ingenieros de software y los especialistas en pruebas desarrollan una estrategia para probar el software.
- ¿Por qué es importante?
  - Estrategia sistemática para probar el SW:
    - No se desperdicia tiempo
    - Se emplea el esfuerzo necesario
    - Se detectan todos los errores cometidos

Introducción (3/3)

- ¿Cuáles son los pasos?
  - La prueba comienza “por lo pequeño” y avanza “hacia lo grande”
    1. Primeras etapas de prueba se enfocan sobre un solo componente o un pequeño grupo de componentes relacionados

       Descubrir errores en los datos y en la lógica de procesamiento
    2. Deben integrarse hasta que se construya el sistema completo

       Serie de pruebas de orden superior para descubrir errores en la satisfacción de los requerimientos del cliente.
- ¿Cuál es el producto final?
  - **<u>Documento especificación de pruebas</u>**
    - Descripción de la EPSW:

      Estrategia global y procedimiento con pasos de prueba específicos y los tipos de pruebas que se realizarán: Casos de prueba y sus tareas
    - Se revisa <u>ANTES</u> de realizar las pruebas

Enfoque Estratégico para la prueba del SW (1/7)

- Durante el proceso de software, debe definirse una EPSW
- Consejos para una EPSW efectiva:
  - Realizar revisiones técnicas efectivas para eliminar errores antes de comenzar la prueba.
  - La prueba comienza en los componentes y opera "hacia afuera" (integración de todo el sistema de cómputo).
  - Seleccionar la técnica de prueba más adecuada al producto SW
  - Las pruebas las realiza el desarrollador del software y (para proyectos grandes) un grupo de prueba independiente (GPI).
  - Prueba y depuración son actividades diferentes, pero la depuración debe incluirse en cualquier estrategia de prueba.

Enfoque Estratégico para la prueba del SW (2/7)

- Durante el proceso de software, debe definirse una EPSW
- Consejos para una EPSW efectiva:
  - Realizar revisiones técnicas efectivas para eliminar errores antes de comenzar la prueba.
  - La prueba comienza en los componentes y opera "hacia afuera" (integración de todo el sistema de cómputo).
    - **Pruebas de bajo nivel**: verificar un segmento de código
    - **Pruebas de alto nivel**: validan las principales funciones del sistema vs. especificaciones del cliente.
  - Seleccionar la técnica de prueba más adecuada al producto SW
  - Las pruebas las realiza el desarrollador del software y (para proyectos grandes) un grupo de prueba independiente (GPI).
  - Prueba y depuración son actividades diferentes, pero la depuración debe incluirse en cualquier estrategia de prueba.

Enfoque Estratégico para la prueba del SW (3/7)

- Verificación:
  - Conjunto de tareas que garantizan que el software implementa correctamente una función específica
    - "¿Construimos el producto correctamente?"

- Validación:
  - Conjunto diferente de tareas que aseguran que el software que se construye sigue los requerimientos del cliente.
    - "¿Construimos el producto correcto?"

Enfoque Estratégico para la prueba del SW (4/7)

- Las pruebas representan el último bastión desde donde puede valorarse la calidad y, de manera más pragmática, descubrirse errores.
- Las pruebas no deben verse como una red de seguridad.
  - "No se puede probar la calidad. Si no está ahí antes de comenzar las pruebas, no estará cuando termine de probar".
- La calidad se incorpora en el software a lo largo de todo el proceso de ingeniería del software.
  - La adecuada aplicación de métodos y herramientas, revisiones técnicas efectivas, y gestión y medición sólidas conducen a la calidad que se confirma durante las pruebas.

Enfoque Estratégico para la prueba del SW (5/7)

Organización de las pruebas del SW (1/3)

- Conflicto inherente de intereses que ocurre conforme comienzan las pruebas
  - A las personas que construyen el software se les pide probarlo
    - Mucho interés en demostrar que el programa está libre de errores, que funciona de acuerdo con los requerimientos del cliente y que se completará a tiempo y dentro del presupuesto
    - El constructor actuará con cuidado, y diseñará y ejecutará pruebas que demostrarán que el programa funciona, en lugar de descubrir errores

Enfoque Estratégico para la prueba del SW (5/7)

Organización de las pruebas del SW (2/3)

- Interpretaciones de lo anterior:
  - El desarrollador de software no debe hacer pruebas en absoluto
  - El software debe “ponerse tras una pared” que lo separe de los extraños que lo probarán sin misericordia
  - Las pruebas deben involucrarse con el proyecto sólo cuando los pasos de las pruebas estén por comenzar

Enfoque Estratégico para la prueba del SW (5/7)

Organización de las pruebas del SW (2/3)

- Interpretaciones de lo anterior:
  - El desarrollador de software no debe hacer pruebas en absoluto
  - El software debe “ponerse tras una pared” que lo separe de los extraños que lo probarán sin misericordia
  - Las pruebas deben involucrarse con el proyecto sólo cuando los pasos de las pruebas estén por comenzar

LAS AFIRMACIONES ANTERIORES SON FALSAS!!

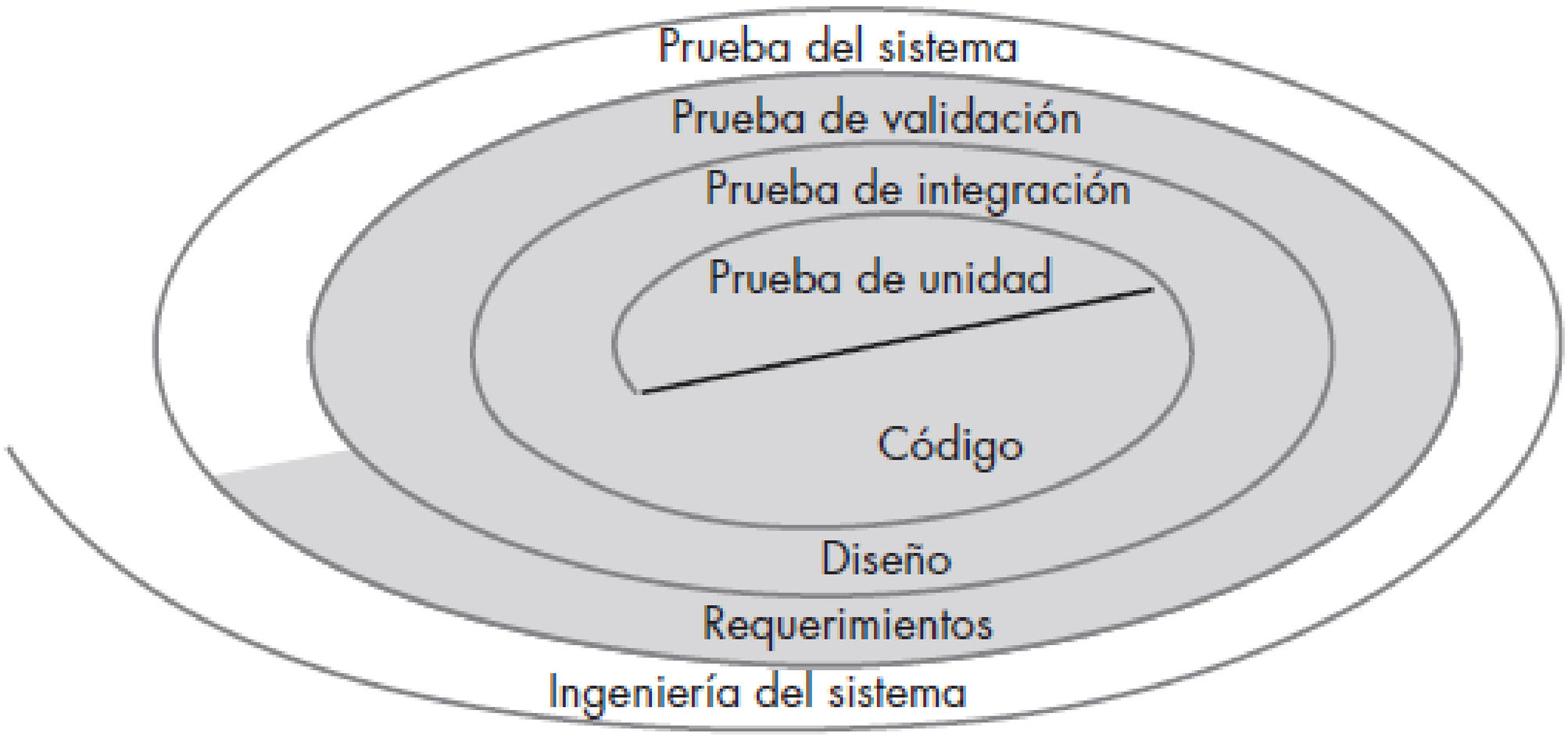
Enfoque Estratégico para la prueba del SW (5/7)

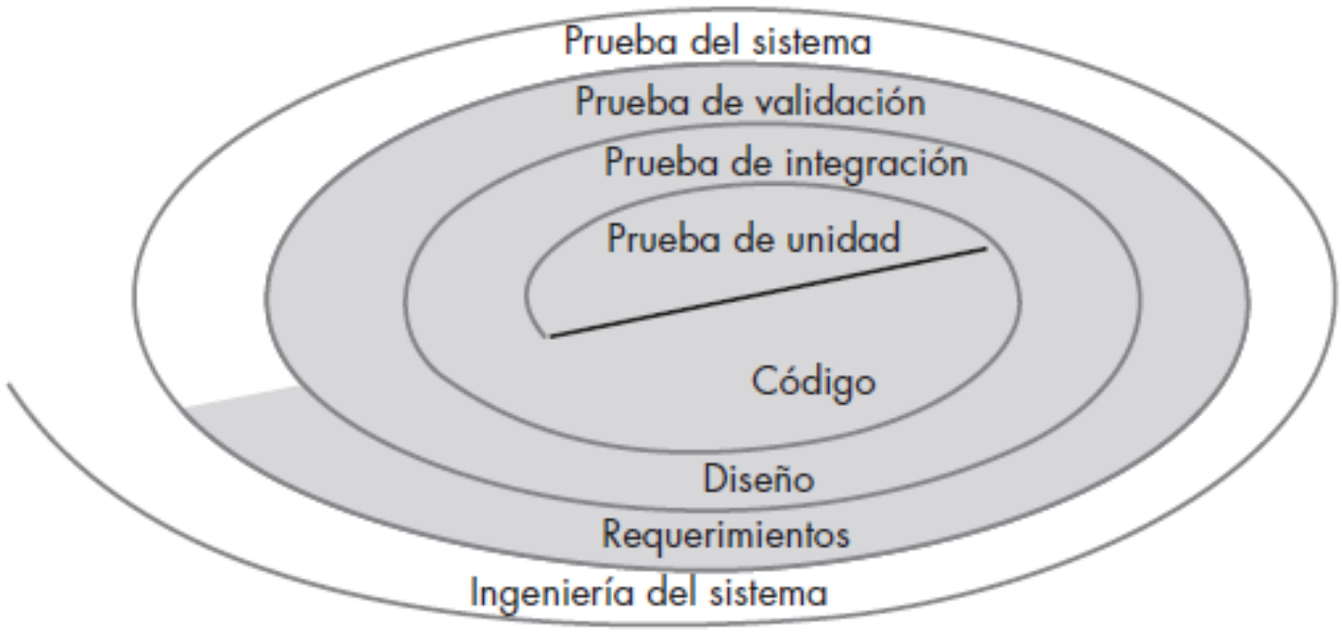
Organización de las pruebas del SW (3/3)

- Buena organización de las pruebas del SW:
  - El desarrollador de software siempre es responsable de probar las unidades individuales (componentes) del programa y de asegurarse de que cada una desempeña la función o muestra el comportamiento para el cual se diseñó
  - El desarrollador también realiza pruebas de integración, una etapa en las pruebas que conduce a la construcción (y prueba) de la arquitectura completa del software
  - Sólo después de que la arquitectura de software está completa se involucra un **grupo de prueba independiente (GPI)**.
    - Las pruebas independientes resuelven el conflicto de intereses que de otro modo puede estar presente
      → Al GPI se le paga por encontrar errores
    - El desarrollador y el GPI trabajan conjuntamente: el desarrollador corrige los errores que se descubran.

Enfoque Estratégico para la prueba del SW (6/7)

Visión general de la EPSW (1/2)

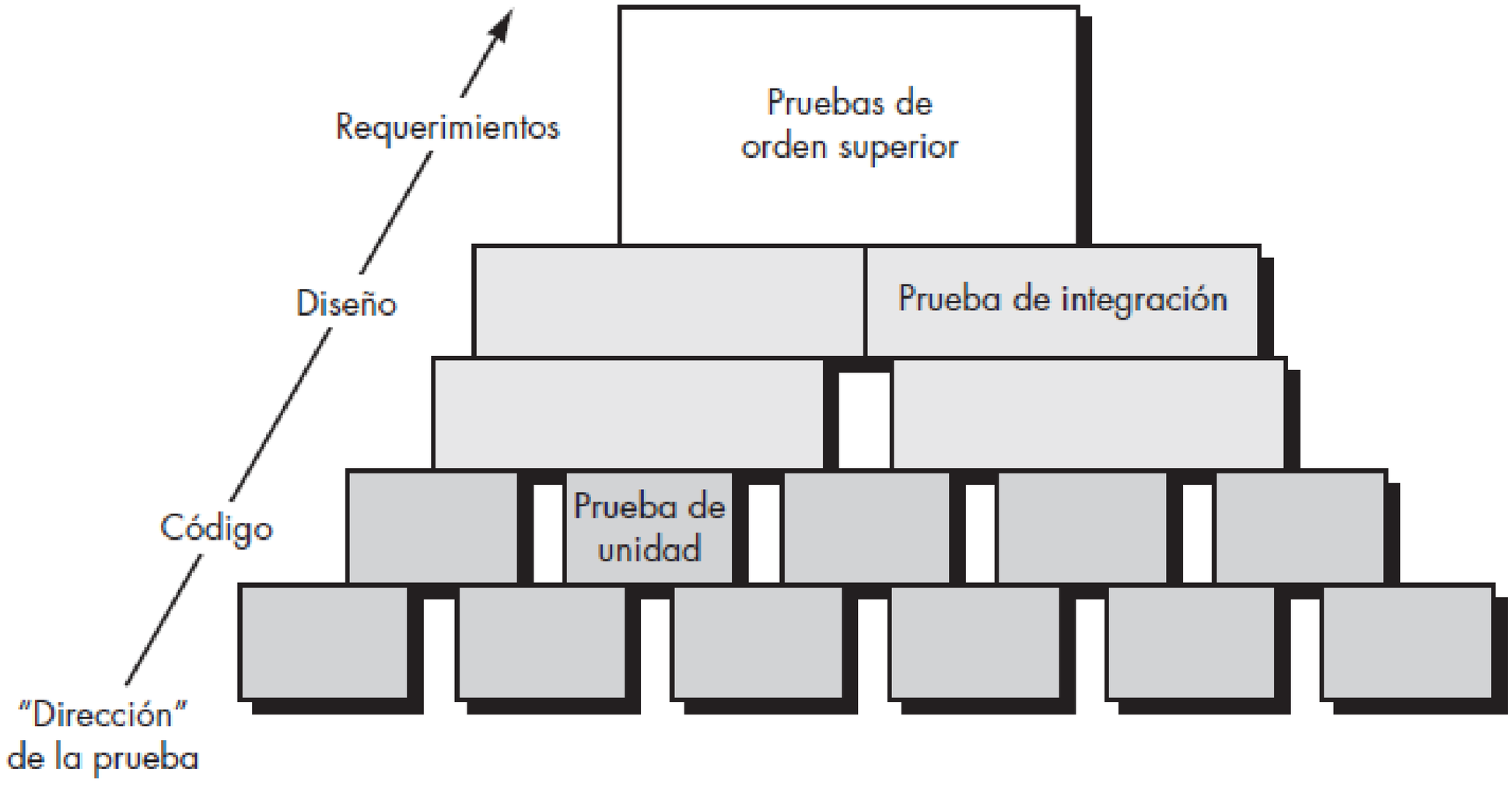




- *Prueba de unidad* se concentra en cada unidad (por ejemplo, componente, clase o un objeto de contenido de una *webapp*) del software: código fuente
- *Prueba de integración*: el enfoque se centra en el diseño y la construcción de la arquitectura del software
- *Prueba de validación*, donde los requerimientos establecidos como parte de su modelado se validan confrontándose con el software que se construyó
- *Prueba del sistema*, donde el software y otros elementos del sistema se prueban como un todo

Enfoque Estratégico para la prueba del SW (6/7)

Visión general de la EPSW (2/2)



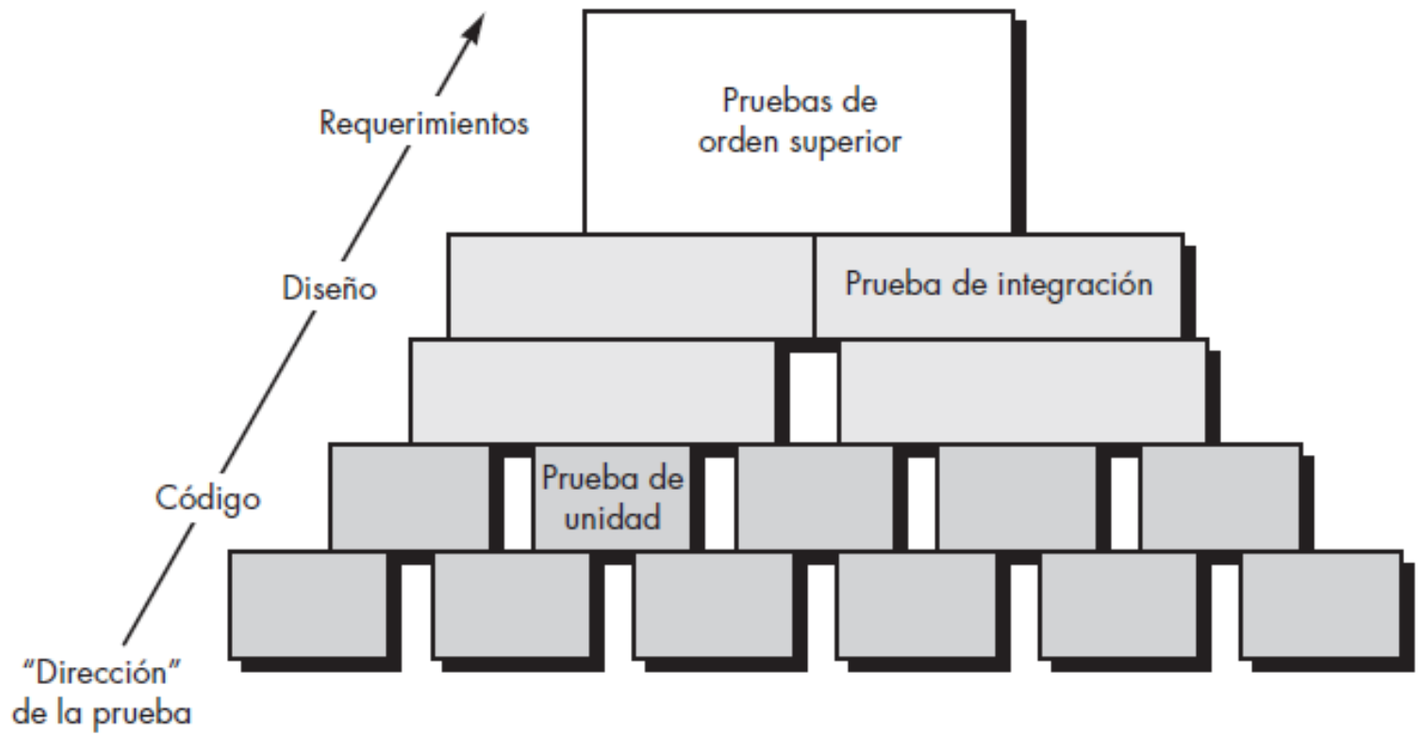
Enfoque Estratégico para la prueba del SW (7/7)

- *Especifican los requerimientos del producto en forma cuantificable mucho antes de comenzar con las pruebas*
- *Establecen de manera explícita los objetivos de las pruebas*
- *Entienden a los usuarios del software y desarrollan un perfil para cada categoría de usuario*
- *Desarrollan un plan de prueba que enfatice "pruebas de ciclo rápido".*
- *Construyen software "robusto" que esté diseñado para probarse a sí mismo*
- *Usan revisiones técnicas efectivas como filtro previo a las pruebas*
- *Realizan revisiones técnicas para valorar la estrategia de prueba y los casos de prueba*
- *Desarrollan un enfoque de mejora continuo para el proceso de prueba (métricas)*

EPSW Convencional

A. Esperarse hasta que el sistema esté completamente construido y luego realizar las pruebas sobre el sistema total, con la esperanza de encontrar errores
   - Software defectuoso
B. Realizarse pruebas diariamente
   - Muy efectivo pero poco atractivo a los desarrolladores de SW

❑ **Visión incremental:**
- Prueba de Unidad
- Prueba de Integración



EPSW Convencional (1/2)

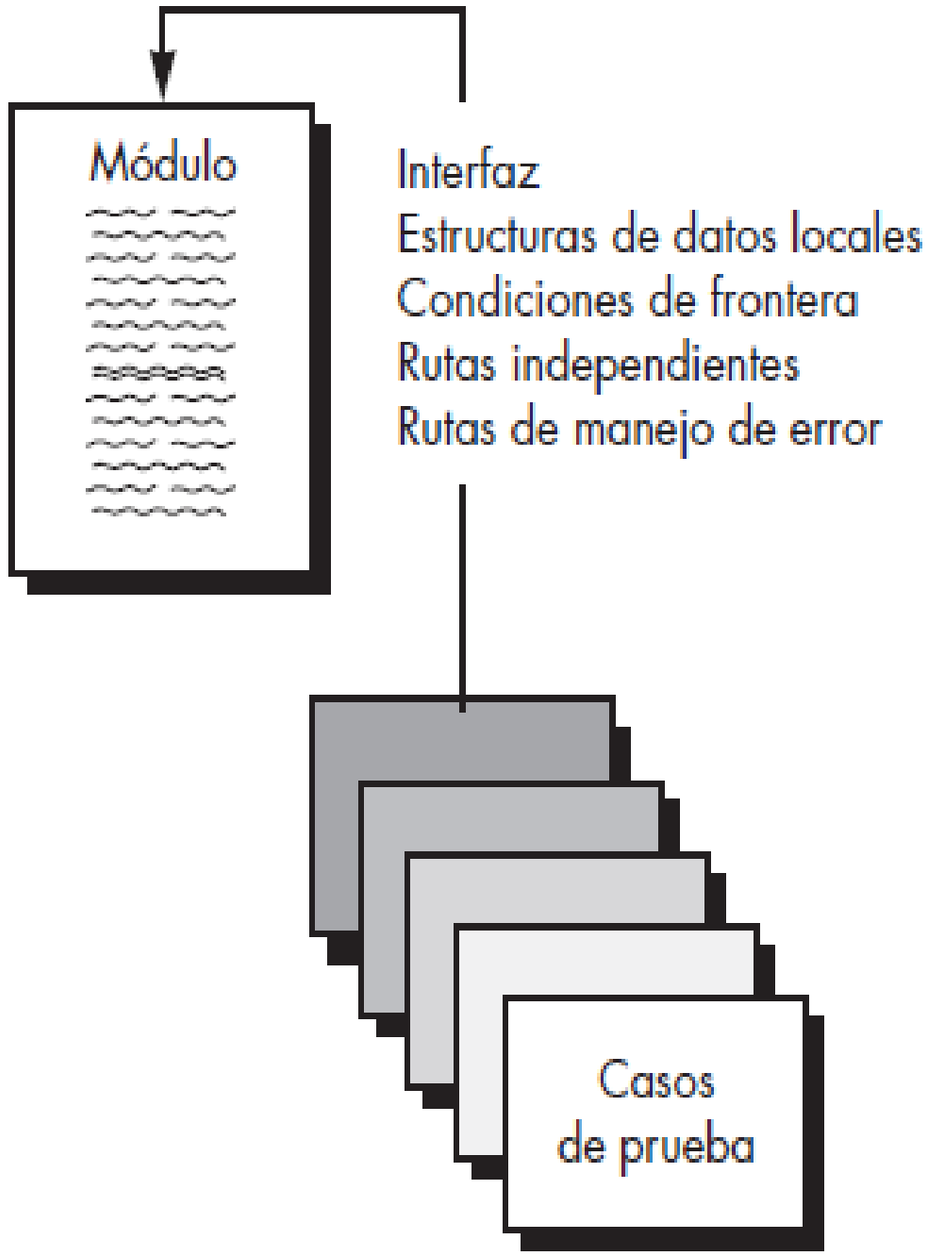
Prueba de Unidad (1/3)

- Esfuerzos de verificación en la unidad más pequeña del diseño de software (componente o módulo SW)
  - Descripción del diseño de componente como guía, las rutas de control importantes se prueban para descubrir errores dentro de la frontera del módulo
- Se centran en la lógica de procesamiento interno y de las estructuras de datos dentro de las fronteras de un componente
- Este tipo de pruebas puede realizarse en paralelo para múltiples componentes

EPSW Convencional (1/2)

Prueba de Unidad (2/3)

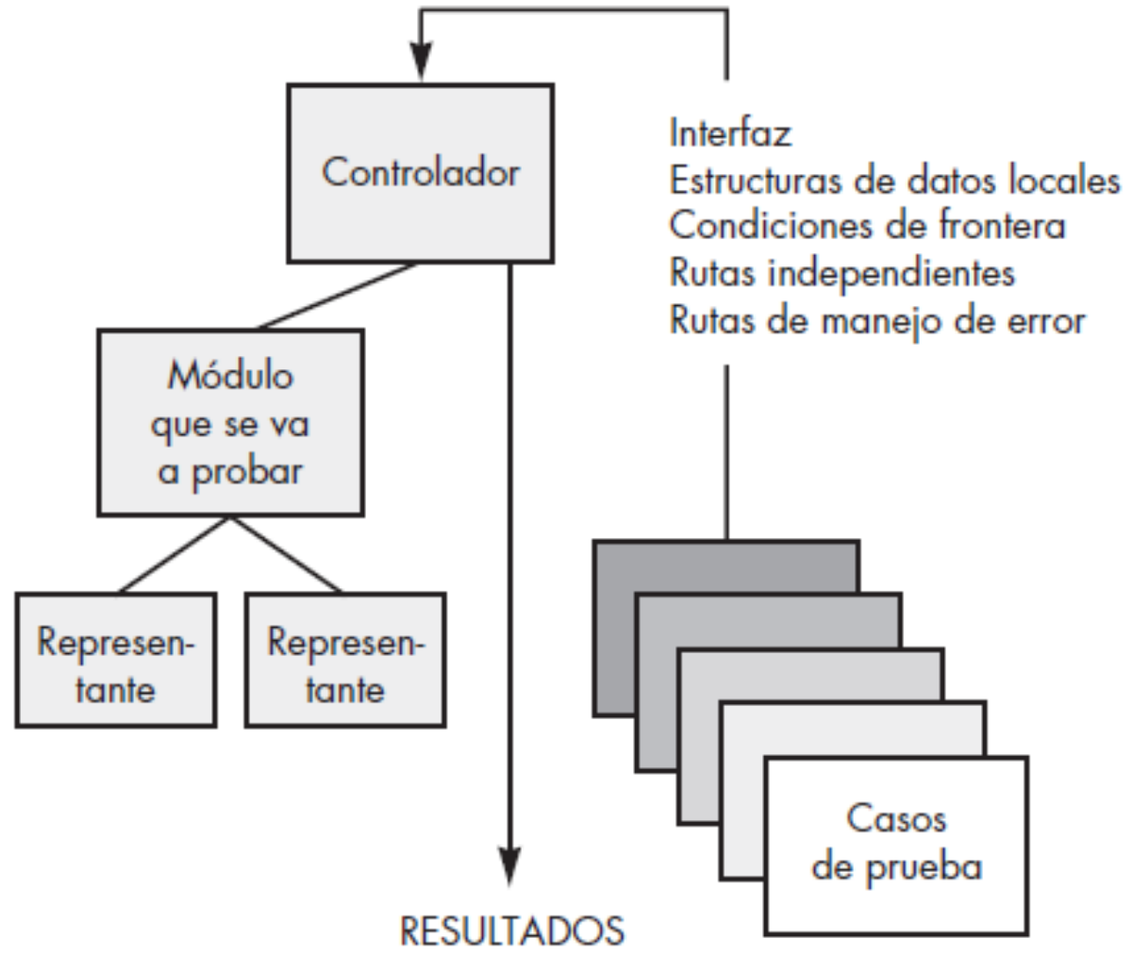
1. Interfaz del módulo
   - Garantizar que la información fluya de manera adecuada hacia y desde la unidad de software que se está probando
2. Estructuras de datos locales
   - Asegurar que los datos almacenados temporalmente mantienen su integridad
3. Condiciones frontera
   - Asegurar que el módulo opera adecuadamente en las fronteras para limitar o restringir el procesamiento (n-ésimo elemento en un array de n elementos)
4. Rutas independientes en las estructuras de control y rutas de manejo de error
   - Todas los caminos se pueden ejecutar



EPSW Convencional (1/2)

Prueba de Unidad (3/3)

- Cada caso de prueba debe acoplarse con un conjunto de resultados esperados
- Un componente no es un programa independiente
  - Software **controlador**
    - "Programa principal" que acepta datos de caso de prueba, pasa tales datos al componente (que va a ponerse a prueba) e imprime resultados relevantes
  - Software **representante (stub)**
    - Sustituir módulos que están subordinados al (invocados por el) componente que se va a probar



→ Realizar mínima manipulación de datos, imprimir verificación de entradas y regresar el control al módulo sobre el que se realiza la prueba

EPSW Convencional (2/2)

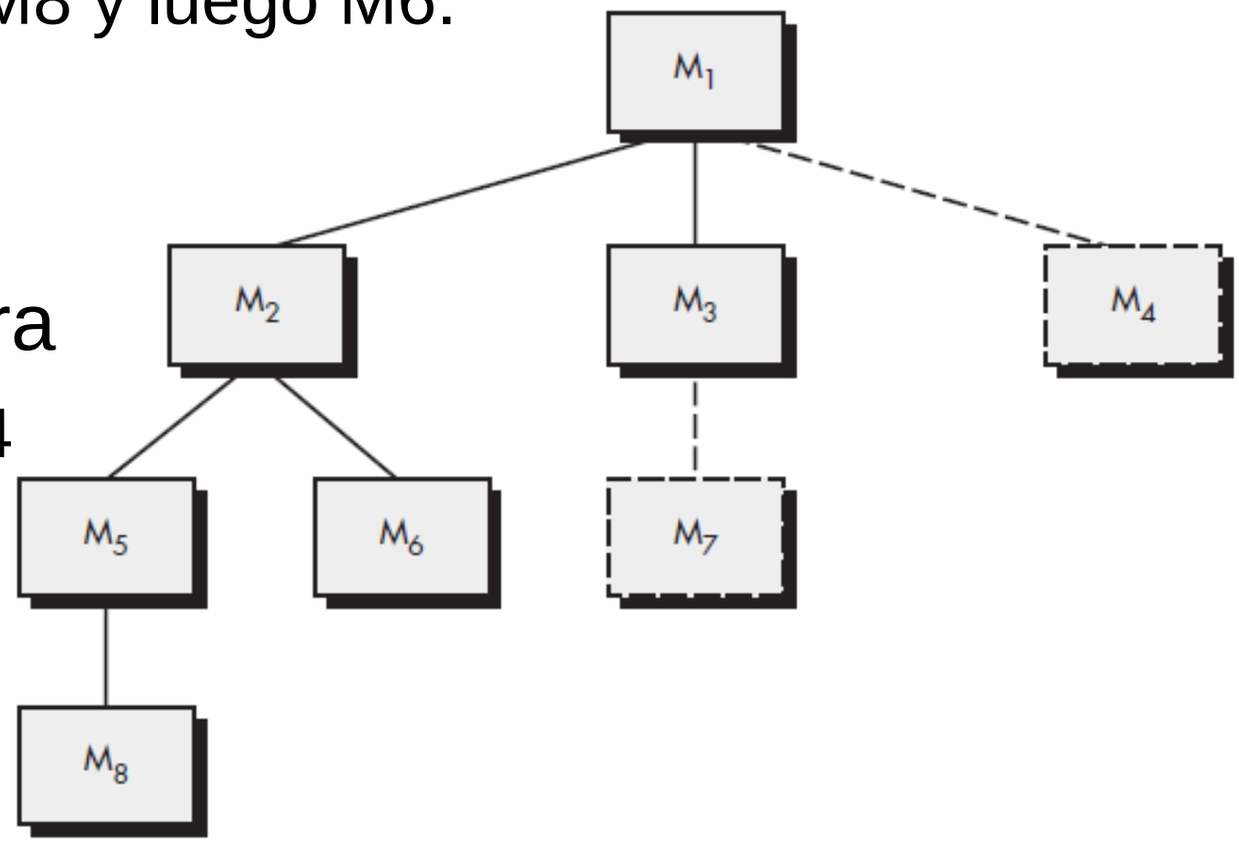
Prueba de Integración (1/12)

- Si todos los módulos se han probado satisfactoriamente pueden existir problemas en su integración
  - Efectos adversos entre componentes, estructuras de datos perdidas en la conexión de componentes, combinación de funciones entre módulos no ofrece el comportamiento esperado, etc.
- Enfoques de prueba de integración:
  - **Big bang:** Todos los componentes se combinan por adelantado y todo el programa se prueba como un todo → ***Problemas??***
  - **Incremental**: El programa se construye y prueba en pequeños incrementos → ***Ventajas??***
    - Descendente
    - Ascendente

EPSW Convencional (2/2)

Prueba de Integración (2/12) → **Enfoque Incremental descendente**

- Los módulos se integran al moverse hacia abajo a través de la jerarquía de control
  - Recorrido primero en profundidad
    - Ruta izquierda: M1, M2, M5, M8 y luego M6.
    - Ruta central: M1, M3 y M7
    - Ruta derecha: M1, M4
  - Recorrido primero en anchura
    1. Nivel de control: M2, M3 y M4
    2. Nivel: M5, M6 y M7
    3. Nivel: M8.



EPSW Convencional (2/2)

Prueba de Integración (3/12) → **Enfoque Incremental descendente**

- Pasos en el proceso de integración incremental descendente
  1. El módulo de control principal se usa como un controlador de prueba y los representantes (*stubs*) se sustituyen con todos los componentes directamente subordinados al módulo de control principal
  2. Dependiendo del enfoque de integración seleccionado (es decir, primero en profundidad o anchura), los representantes subordinados se sustituyen uno a la vez con componentes reales.
  3. Las pruebas se llevan a cabo conforme se integra cada componente
  4. Al completar cada conjunto de pruebas, otro representante se sustituye con el componente real
  5. Las pruebas de regresión pueden realizarse para asegurar que no se introdujeron nuevos errores
  6. Volver al paso 2 hasta que se construye toda la estructura del programa

EPSW Convencional (2/2)

Prueba de Integración (4/12) → **Enfoque Incremental descendente**

- Problema cuando se requiere procesamiento en niveles bajos en la jerarquía a fin de probar de manera adecuada los niveles superiores
  - Stubs: ningún dato significativo puede fluir hacia arriba en la estructura del programa
  - Opciones:
    - Demorar las pruebas hasta que los representantes se sustituyan con módulos reales → dificultades para determinar la causa de los errores
    - Desarrollar resguardos que realicen funciones limitadas que simulen al módulo real → sobrecarga significativa conforme los representantes se vuelven cada vez más complejos
    - Integrar el software desde el fondo de la jerarquía y hacia arriba

      →Integración ascendente

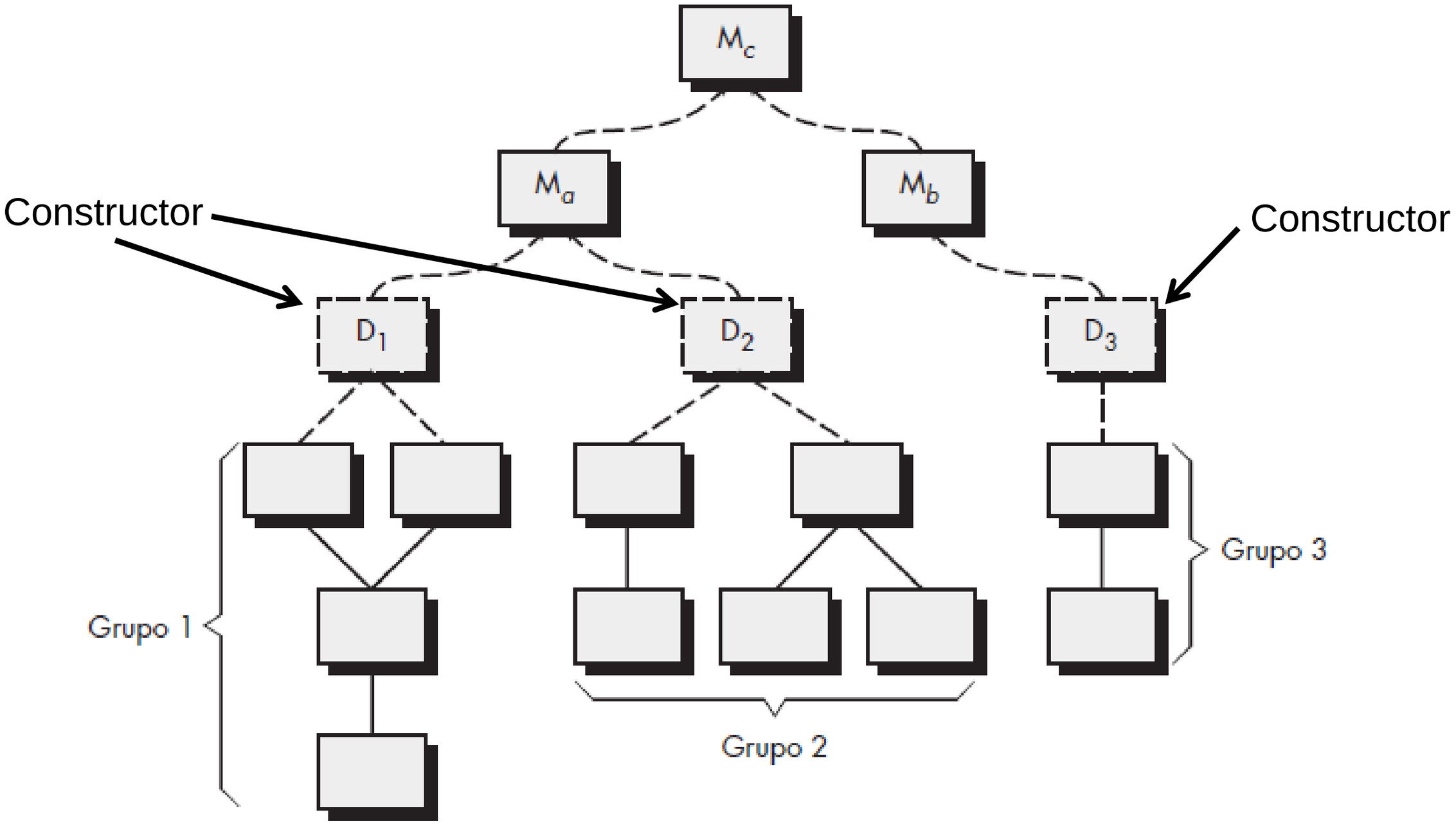
EPSW Convencional (2/2)

Prueba de Integración (5/12) → **Enfoque Incremental ASCENDENTE**

- Comienza con prueba de *módulos atómicos* (componentes en los niveles inferiores dentro de la estructura del programa) → No hacen falta stubs
- Pasos en la prueba de integración incremental ascendente:
  1. Los componentes en el nivel inferior se combinan en grupos (en ocasiones llamados ***construcciones*** o *builds)* que realizan una subfunción de software específica
  2. Se escribe un *controlador* (un programa de control para pruebas) a fin de coordinar la entrada y salida de casos de prueba.
  3. Se prueba el grupo
  4. Los controladores se remueven y los grupos se combinan moviéndolos hacia arriba en la estructura del programa.

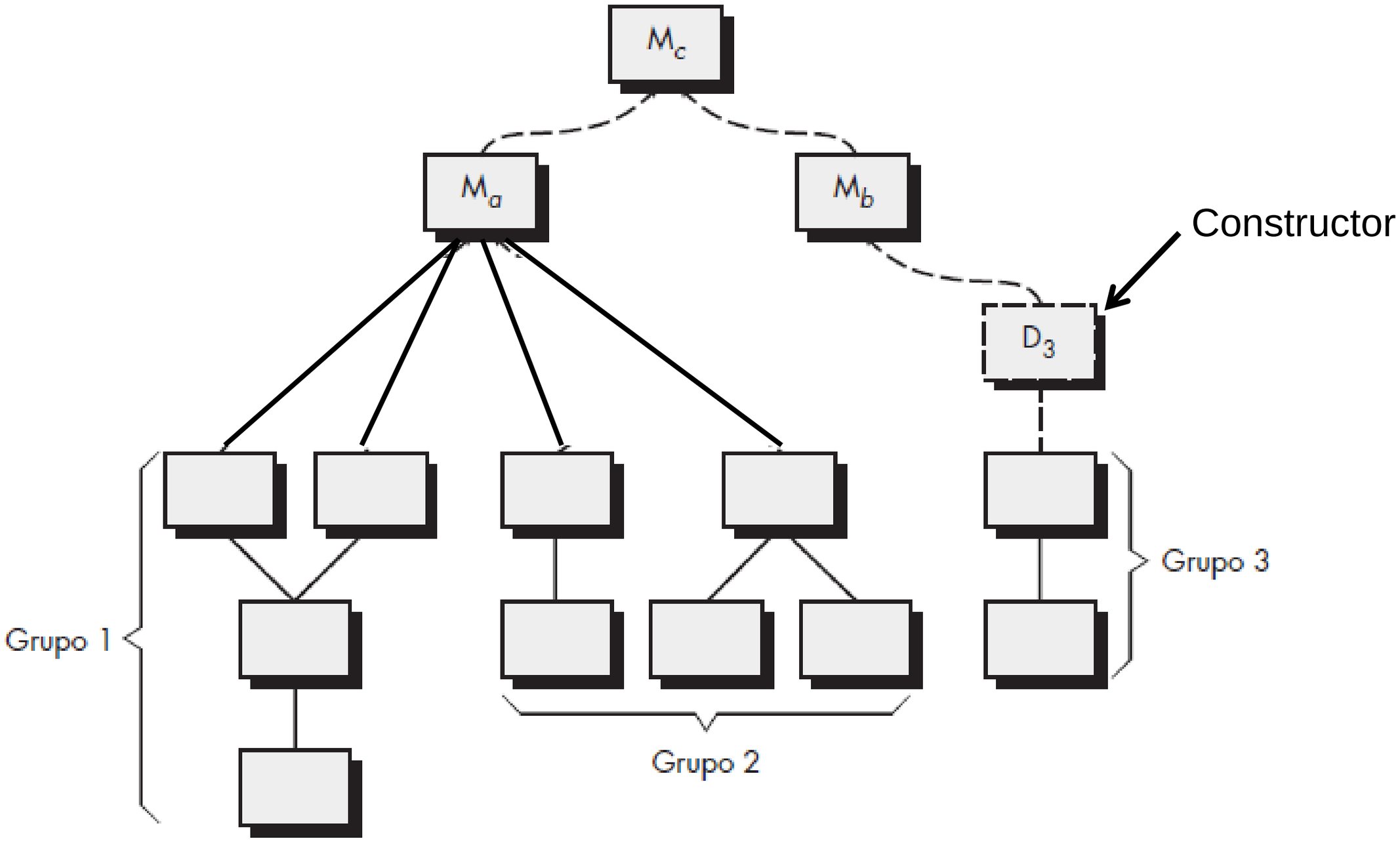
EPSW Convencional (2/2)

Prueba de Integración (6/12) → **Enfoque Incremental ASCENDENTE**



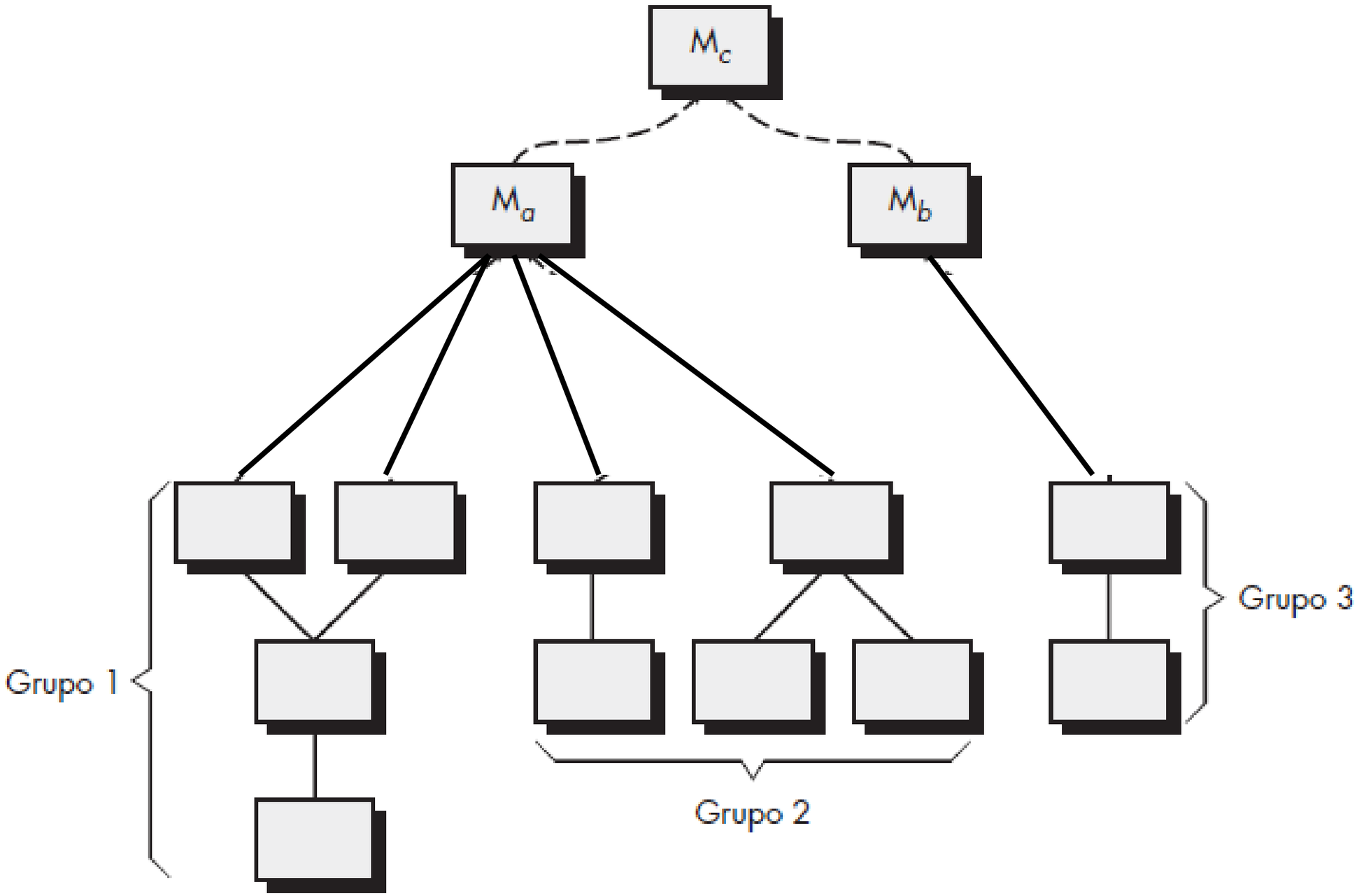
EPSW Convencional (2/2)

Prueba de Integración (6/12) → **Enfoque Incremental ASCENDENTE**



EPSW Convencional (2/2)

Prueba de Integración (6/12) → **Enfoque Incremental ASCENDENTE**



EPSW Convencional (2/2)

Prueba de Integración (7/12) → **Pruebas de Regresión**

- En las pruebas de integración, cada vez que se agrega un nuevo módulo el SW cambia:
  - Nuevas rutas de flujo de datos, ocurren nuevas operaciones de entrada/salida y se invoca nueva lógica de control
  - Causar problemas con las funciones que anteriormente trabajaban sin fallas
- **Prueba de regresión:**
  - Nueva ejecución de algún subconjunto de pruebas que ya se realizaron a fin de asegurar que los cambios no propagaron efectos colaterales no deseados

EPSW Convencional (2/2)

Prueba de Integración (8/12) → **Pruebas de Regresión**

- Ayudan a garantizar que los cambios (debidos a pruebas o por otras razones) no introducen comportamiento no planeado o errores adicionales
- Se pueden realizar manualmente
  - También mediante *herramientas de captura/reproducción* permiten al ingeniero de software capturar casos de prueba y resultados para una posterior reproducción y comparación
- ***Suite de prueba de regresión*** (pruebas a ejecutar):
  - muestra representativa de pruebas que ejercitará todas las funciones de software
  - Pruebas adicionales que se enfocan en las funciones del software que probablemente resulten afectadas por el cambio
  - Pruebas que se enfocan en los componentes del software que cambiaron

Cuidado con el número de pruebas de regresión y el tamaño de la suite!!

EPSW Convencional (2/2)

Prueba de Integración (9/12) → **Pruebas de Humo**

- Enfoque de prueba de integración que se aplica a proyectos críticos en el tiempo
  - Estrategia de integración constante. El software se reconstruye (con el agregado de nuevos componentes) y se prueba cada día.
  - No tiene que ser exhaustiva pero debe poder exponer los problemas principales.
- Actividades:
  - Los componentes de software traducidos en código se integran en una *construcción*. Una construcción incluye todos los archivos de datos, bibliotecas, módulos reutilizables y componentes sometidos a ingeniería que se requieren para implementar una o más funciones del producto
  - Se diseña una serie de pruebas para exponer los errores que evitarán a la construcción realizar adecuadamente su función. La intención debe ser descubrir errores “paralizantes” que tengan la mayor probabilidad de retrasar el proyecto
  - La construcción se integra con otras construcciones, y todo el producto (en su forma actual) se somete a prueba de humo diariamente. El enfoque de integración puede ser descendente o ascendente

EPSW Convencional (2/2)

Prueba de Integración (10/12) → **Pruebas de Humo**

- Beneficios de aplicar pruebas de humo
  - Se minimiza el riesgo de integración:
    - Se realizan diariamente → las incompatibilidades y otros errores paralizantes pueden descubrirse tempranamente
  - La calidad del producto final mejora
    - Descubre errores funcionales así como errores de diseño tempranos
  - El diagnóstico y la corrección de errores se simplifican
    - Los errores descubiertos durante la prueba de humo se asocien con "nuevos incrementos de software" → la parte agregada es probable que sea el error
  - El progreso es más fácil de valorar
    - Con cada día que transcurre, más software se integra y se demuestra que funciona → Incrementa la moral del equipo y brinda a los gerentes un buen indicio de que se está progresando

EPSW Convencional (2/2)

Prueba de Integración (11/12) → **Productos de trabajo**

❑ Tipos y fases de pruebas

- *Integridad de interfaz*. Las interfaces internas y externas se prueban conforme cada módulo (o grupo) se incorpora en la estructura.
- *Validez funcional*. Se realizan pruebas diseñadas para descubrir errores funcionales ocultos.
- *Contenido de la información*. Se realizan pruebas diseñadas para descubrir errores ocultos asociados con las estructuras de datos locales o globales.
- *Rendimiento*. Se realizan pruebas diseñadas para verificar los límites del rendimiento establecidos durante el diseño del software

EPSW Convencional (2/2)

Prueba de Integración (12/12) → **Productos de trabajo**

- **Documento especificación de pruebas**
  - Plan de prueba para integración del software
    - Calendario para la integración
    - Desarrollo de software de sobrecarga del sistema (stubs y controladores)
    - Entorno y recursos de la prueba
  - Procedimiento de prueba (descripción de las pruebas específicas a realizar)
    - Se señala el orden de la integración y las pruebas correspondientes en cada paso de ésta.
    - Lista de todos los casos de prueba (anotados para referencia posterior) y los resultados esperados
  - *Reporte de prueba (anexo)*
    - Historia de resultados, problemas o peculiaridades de prueba reales