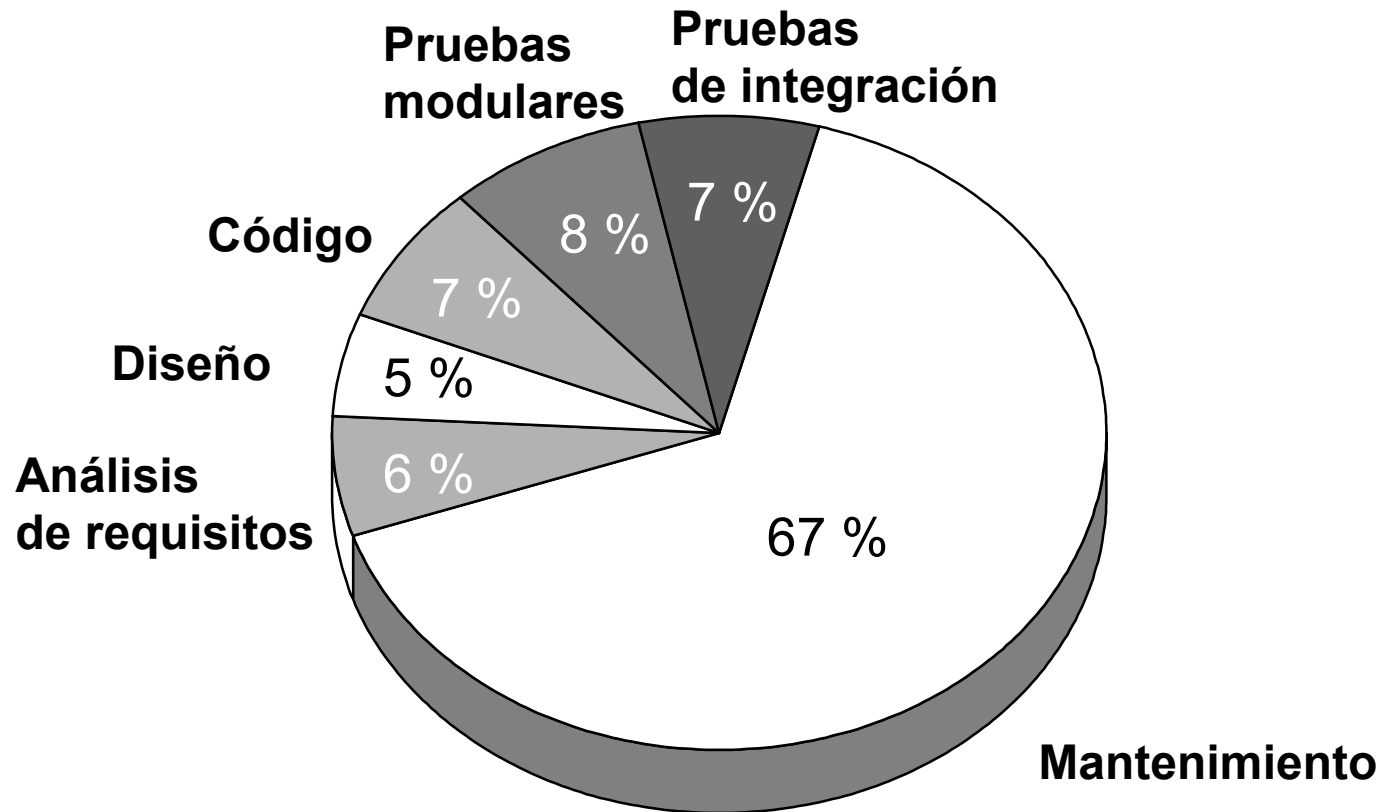





DISTRIBUCION DEL COSTE DEL CICLO DE VIDA



FACTORES

- ☐ *Inexistencia de métodos, técnicas y herramientas que puedan proporcionar una solución global al mantenimiento*
- ☐ *La complejidad de los sistemas se incrementa paulatinamente por la realización de continuas modificaciones*
- ☐ *La documentación del sistema es defectuosa o inexistente*
- ☐ *Se considera el mantenimiento como una actividad poco creativa, a diferencia del desarrollo*
- ☐ *Las actividades del mantenimiento se suelen realizar bajo presión de tiempo*
- ☐ *Poca participación del usuario durante el desarrollo del sistema*

ACTUACIONES COMUNES PARA MANTENER LA OPERATIVIDAD DEL SOFTWARE

-  *Corrección de defectos en el software*
-  *Creación de nuevas funcionalidades en el software por nuevos requisitos de usuario*
-  *Mejora de la funcionalidad y del rendimiento*

TIPOS DE MANTENIMIENTO

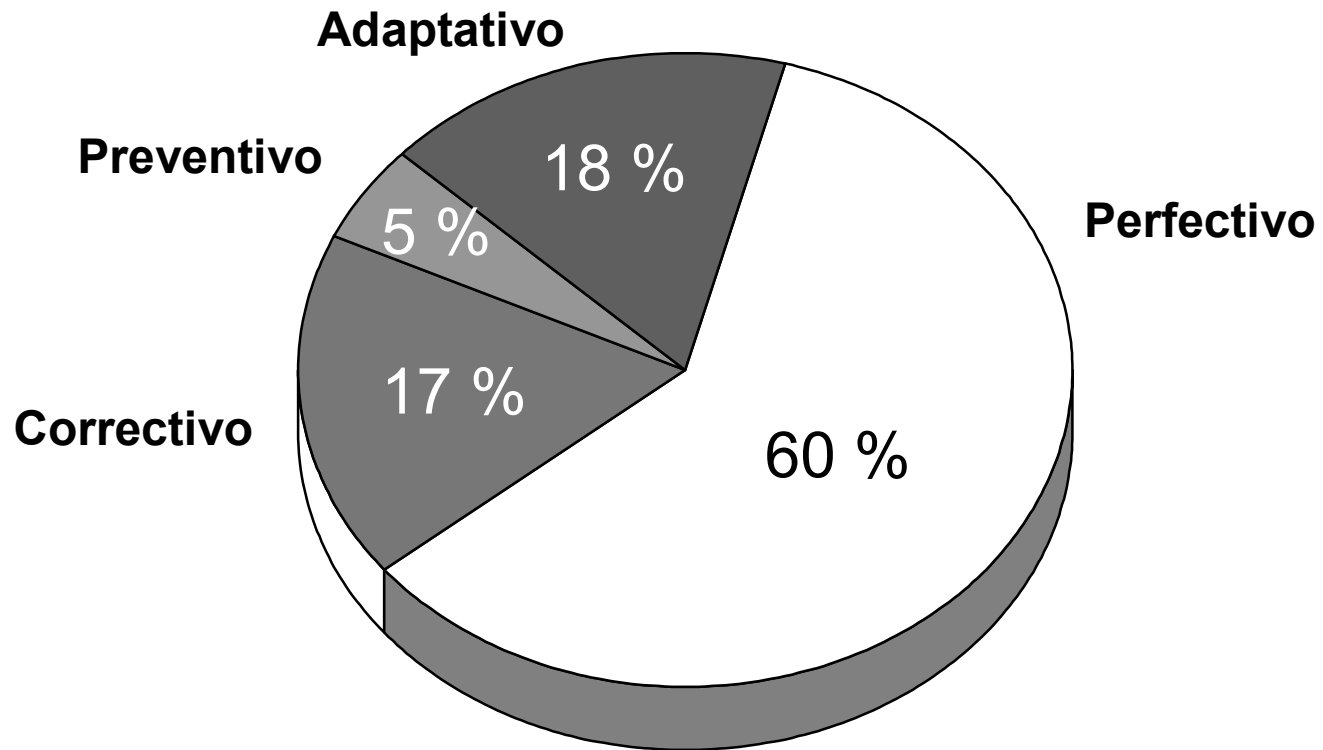
***Mantenimiento perfectivo:** conjunto de actividades para mejorar o añadir nuevas funcionalidades requeridas por el usuario*

***Mantenimiento adaptativo:** es el conjunto de actividades para adaptar el sistema a los cambios (hardware o software) en su entorno tecnológico*

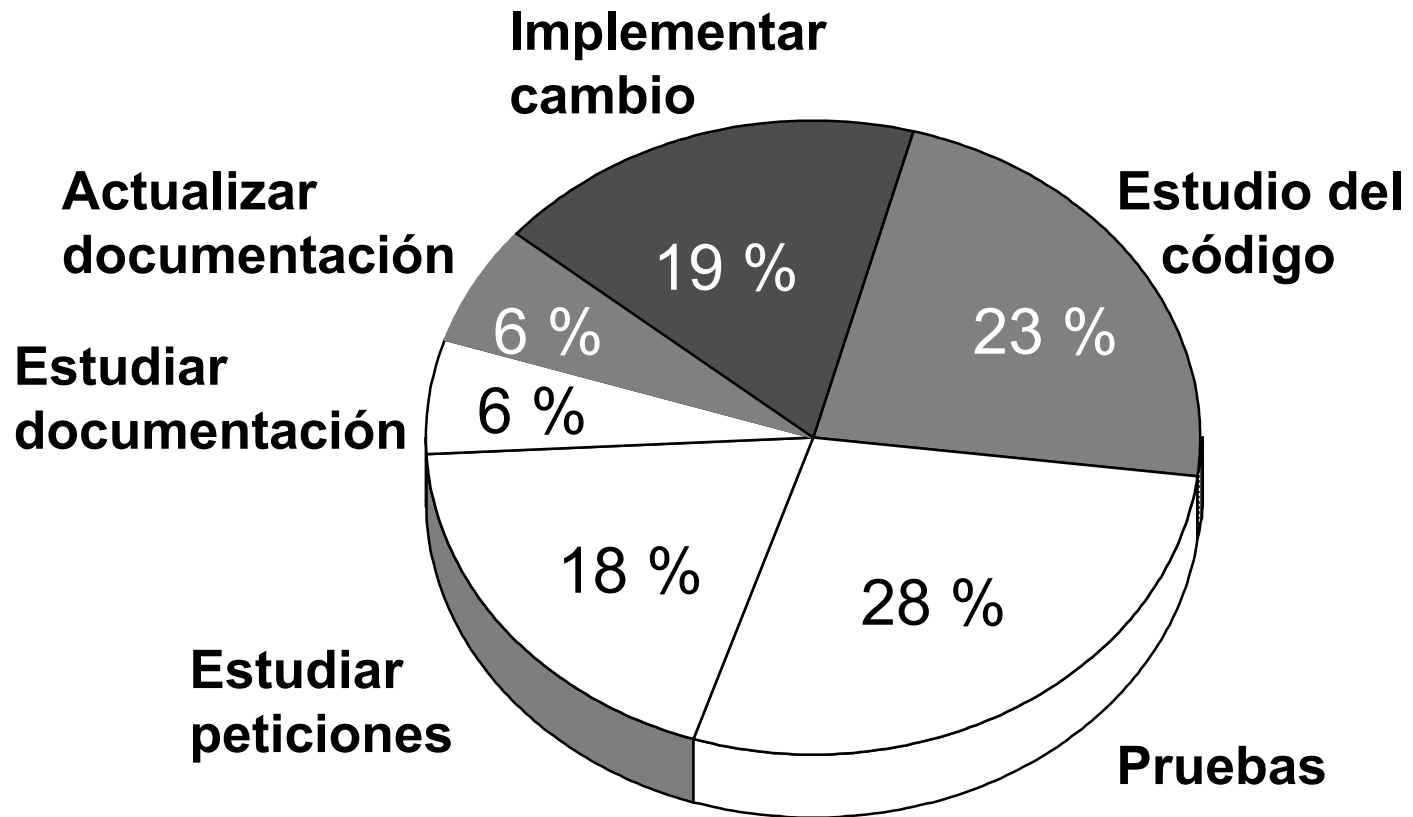
***Mantenimiento correctivo:** es el conjunto de actividades dedicadas a corregir defectos en el hardware o en el software detectados por los usuarios durante la explotación del sistema*

***Mantenimiento preventivo:** conjunto de actividades para facilitar el mantenimiento futuro del sistema*

TIPOS DE MANTENIMIENTO Y COSTE RELATIVO



DISTRIBUCION DEL TIEMPO EN TAREAS DE MANTENIMIENTO



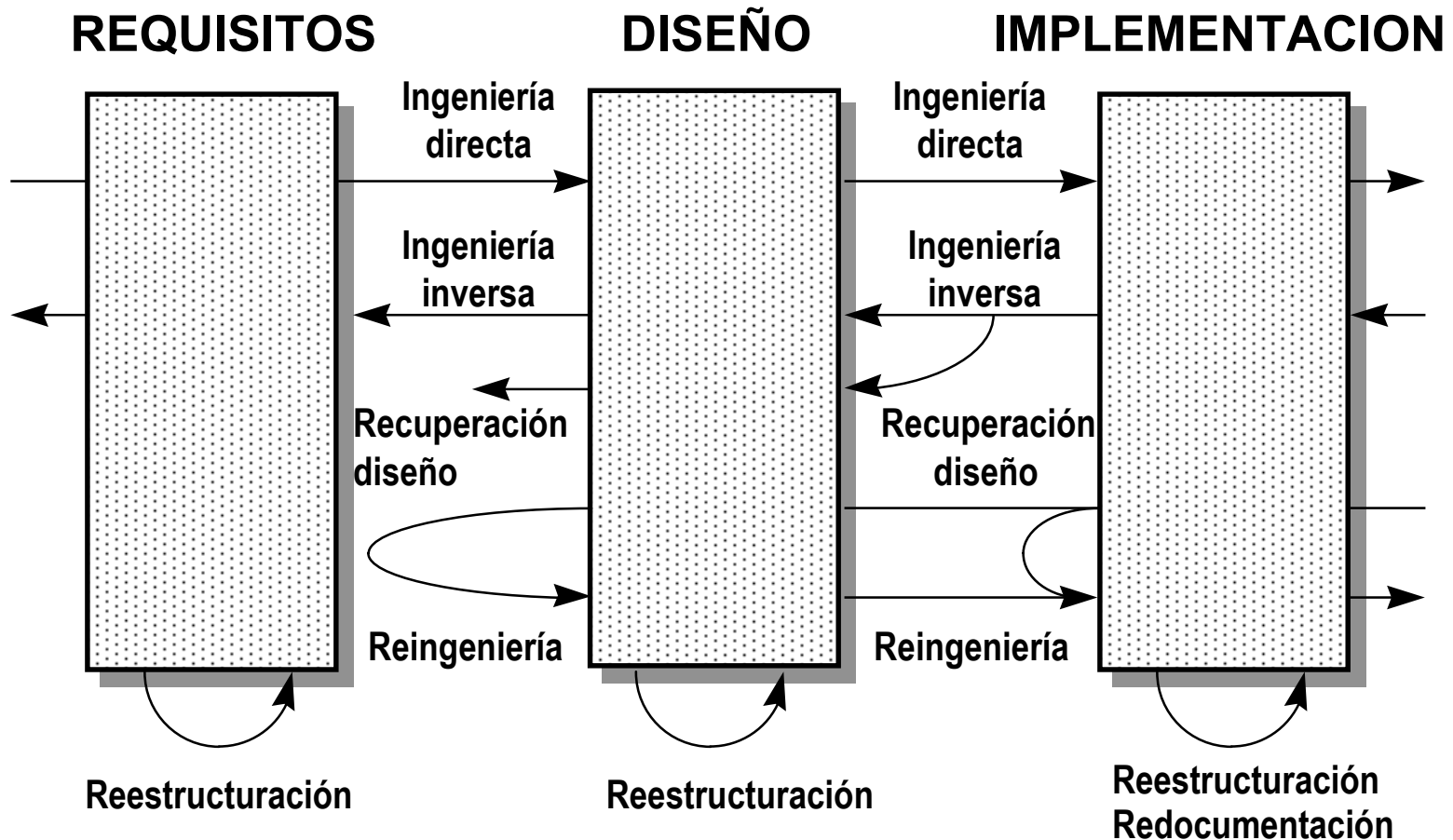
LA REINGENIERÍA DEL SOFTWARE

TECNOLOGÍA DE LA REINGENIERÍA	
MEJORA DEL SOFTWARE	Reestructuración. Redocumentación, Anotación, actualización de documentación. Ingeniería para reutilización. Remodularización. Reingeniería de procesos de negocio (BPR) Reingeniería de datos. Análisis de facilidad de mantenimiento, análisis económico.
COMPRENSIÓN DEL SOFTWARE	Visualización Análisis, mediciones. Ingeniería inversa, recuperación de diseño.
CAPTURA, CONSERVACIÓN Y EXTENSIÓN DEL CONOCIMIENTO SOBRE EL SOFTWARE	Descomposición. Ingeniería Inversa y recuperación de diseño. Recuperación de objetos. Comprensión de programas. Transformaciones y bases de conocimiento.

LA IMPORTANCIA DE LA REINGENIERÍA DEL SOFTWARE

- ☺ *Puede reducir los riesgos evolutivos de una organización*
- ☺ *Puede ayudar a las organizaciones a recuperar sus inversiones en software*
- ☺ *Puede hacer el software más fácilmente modificable*
- ☺ *Amplía las capacidades de las herramientas CASE*
- ☺ *Es un catalizador para la automatización del mantenimiento del software*
- ☺ *Puede actuar como catalizador para la aplicación de técnicas de inteligencia artificial (IA) para resolver problemas de reingeniería*

RELACIONES ENTRE LOS TERMINOS ASOCIADOS CON LA REINGENIERIA



INGENIERIA DIRECTA

«Corresponde al desarrollo de software tradicional»

REESTRUCTURACION

«Es la transformación de una forma de representación a otra en el mismo nivel de abstracción relativo, mientras se mantenga el comportamiento externo del sistema (funcionalidad y semántica)»

«Es la modificación del software para hacerlo más fácil de entender y cambiar»

INGENIERIA INVERSA

«Es el proceso de análisis de un sistema para identificar sus componentes e interrelaciones y crear representaciones del sistema en otra forma o a un nivel más alto de abstracción»

AREAS EN LA INGENIERÍA INVERSA

***Redocumentación:** «es la creación o revisión de una representación equivalente semánticamente dentro del mismo nivel de abstracción relativo»*

***Recuperación de diseño:** «es un subconjunto de la ingeniería inversa, en el cual, aparte de las observaciones del sistema, se añaden conocimientos sobre su dominio de aplicación, información externa, y procesos deductivos con el objeto de identificar abstracciones significativas a un mayor nivel»*

REDISEÑO

«Consiste en consolidar y modificar los modelos obtenidos, añadiendo nuevas funciones requeridas por los usuarios»

REINGENIERÍA DEL SOFTWARE

«Es el examen y alteración de un sistema para reconstruirlo de una nueva forma y la subsiguiente implementación de esta nueva forma»

OTRAS TECNOLOGIAS

*La **remodularización** consiste en cambiar la estructura modular de un sistema de forma que se obtenga una nueva estructura siguiendo los principios del diseño estructurado*

***Análisis de la facilidad de mantenimiento.** Normalmente la mayor parte del mantenimiento se centra relativamente en unos pocos módulos del sistema*

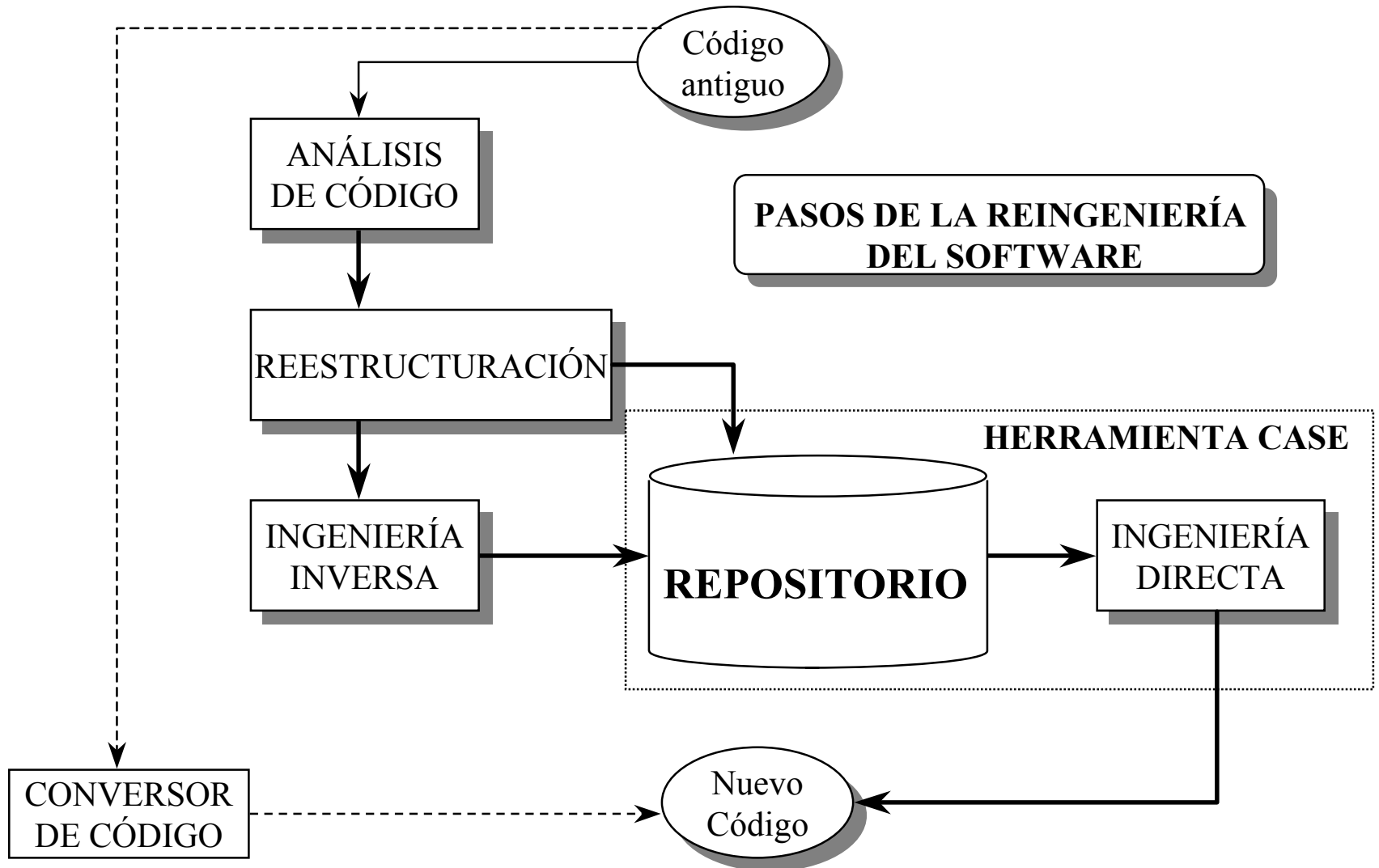
***Visualización.** El proceso más antiguo para la comprensión del software*

***Análisis y mediciones.** Son importantes tecnologías que estudian ciertas propiedades de los programas*

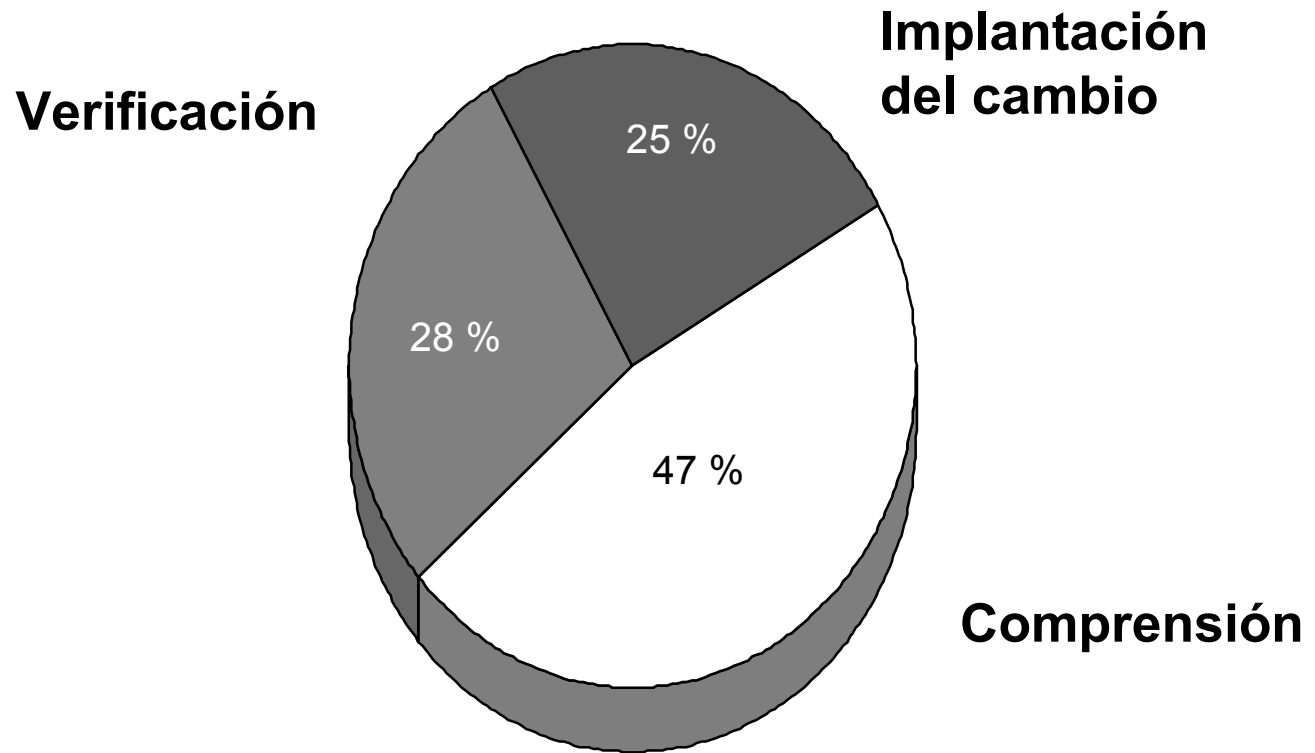
PROCESO DE REINGENIERIA DEL SOFTWARE

- ☐ *Mejorar su facilidad de mantenimiento futuro*
- ☐ *Facilitar su migración, que el proceso de traducir un programa de un lenguaje a otro, moverlo de un entorno operativo a otro o actualizar su tecnología*
- ☐ *Aumentar su esperanza de vida*
- ☐ *Capturar sus componentes en un repositorio que puede ser gestionado por herramientas CASE*
- ☐ *Incrementar la productividad de mantenimiento*

PASOS DE LA REINGENIERIA DEL SOFTWARE



ANALISIS DE CÓDIGO FUENTE



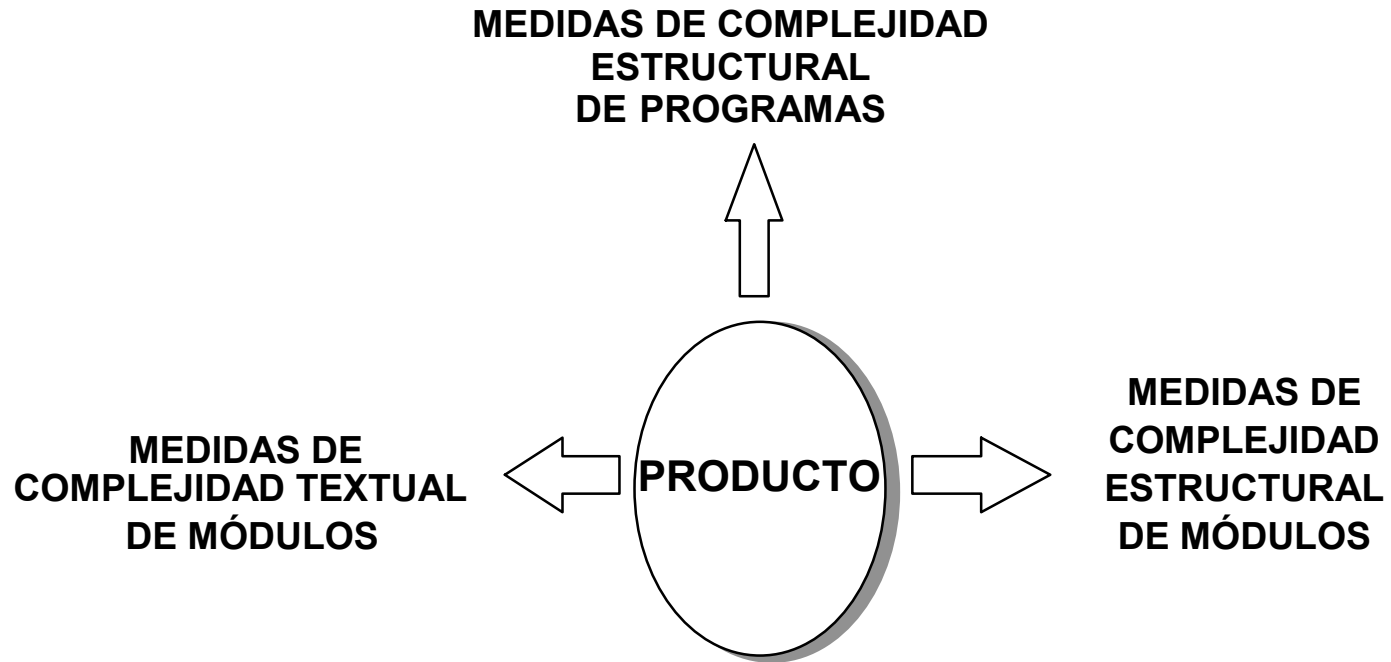
ANALISIS DE CÓDIGO FUENTE

Análisis estático

Consiste en una evaluación que estudia la estructura del código sin ejercitar o ejecutar dicho código.

- ॐ Auditoría de código: revisión del código para identificar errores de sintaxis y para comprobar el seguimiento de los estándares de codificación*
- ॐ Métricas de producto: permiten obtener un conjunto de métricas sobre distintos atributos del software*
- ॐ Análisis de flujo: identifica el flujo de control y de datos para determinar dónde están los errores*

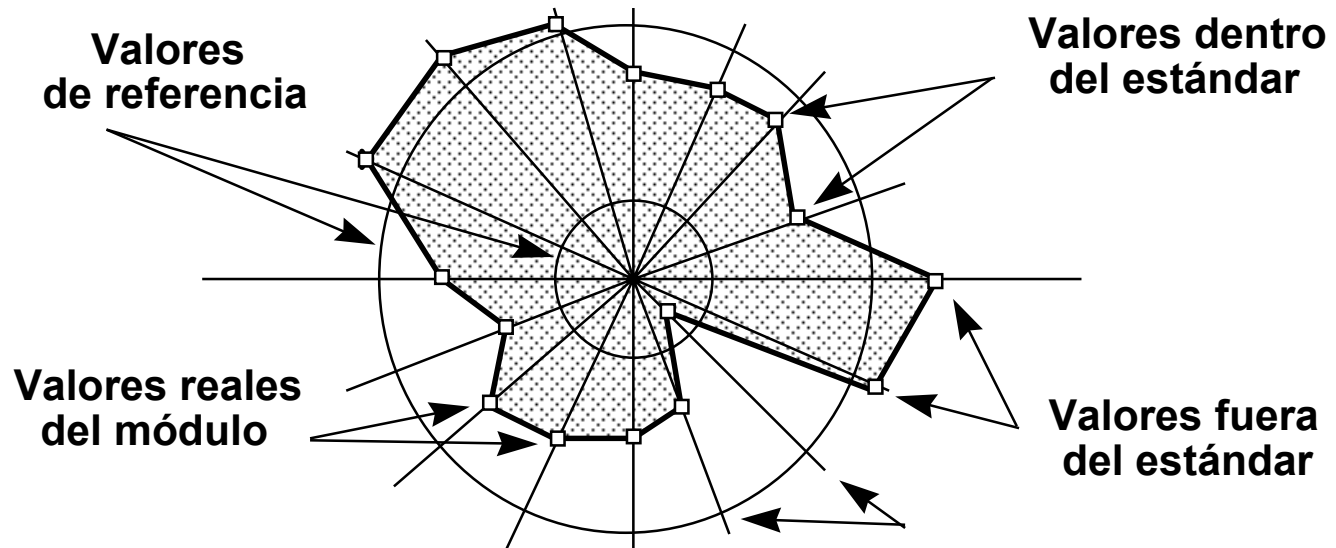
TIPOS DE MÉTRICAS DE PRODUCTO



ALGUNAS METRICAS DE PRODUCTO

Métricas de la complejidad estructural de los programas
Número de caminos Accesibilidad de un módulo Facilidad de prueba de un módulo Complejidad jerárquica Complejidad estructural Facilidad de prueba del sistema Entropía del grafo de llamadas Impureza del grafo de llamadas Complejidad del flujo de información
Métricas de la complejidad estructural de los módulos
Número ciclomático de McCabe Densidad de control Número de nodos pendientes Número máximo de grado Número máximo de niveles
Métricas de la complejidad textual de los módulos
Métricas de Halstead

REPRESENTACION DE UN DIAGRAMA DE KIVIAT

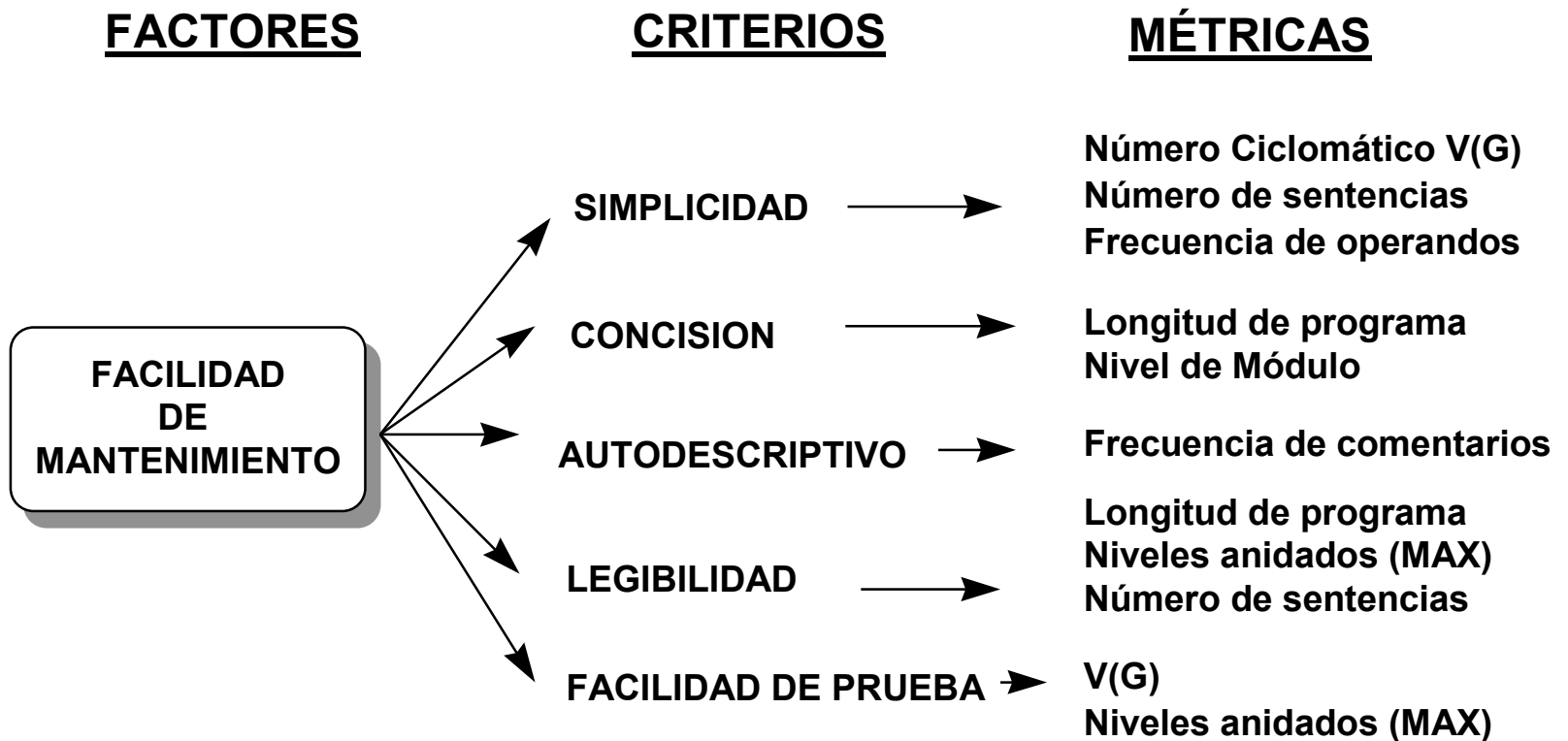


ANALISIS DE CÓDIGO FUENTE

Análisis dinámico o pruebas

Es un proceso por el que se detectan defectos ejecutando el código al comparar los resultados obtenidos con los esperados

UNA SELECCIÓN DE MÉTRICAS PARA MEDIR EL FACTOR «FACILIDAD DE MANTENIMIENTO»

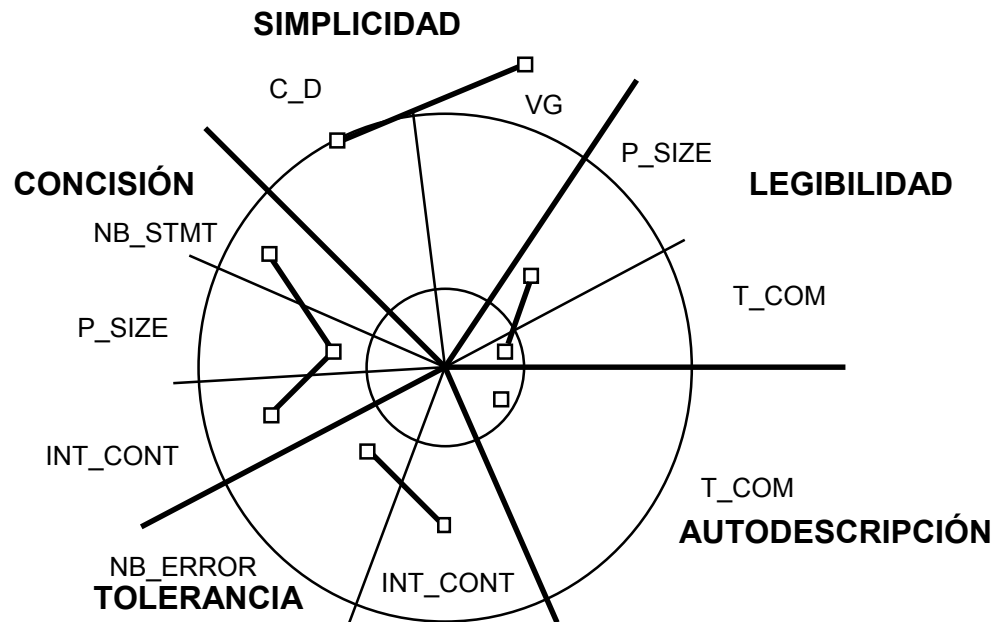


EJEMPLO DE DEFINICION DE MÉTRICAS Y SU ASIGNACION A LOS CRITERIOS DE CALIDAD

Definición de métricas de módulo		
Longitud de programa	:	LENG
Número de sentencias	:	NSTM
Frecuencia comentarios	:	COMF
Frecuencia operandos	:	OPEF
Número ciclomático	:	V(G)
Niveles anidados (máx)	:	NEST
Nivel de módulo	:	MLEV
Selección de límites		
LENG	1	100
NSTM	1	50
COMF	0.15	1.0
OPEF	1.0	3.0
V(G)	1	20
NEST	0	4
MLEV	0.03	1.00
Definición de criterios de calidad		
FACILIDAD DE PRUEBA:	V(G), NEST	
SIMPLICIDAD	: V(G), NSTM, OPEF	
CONCISION	: LENG, MLEV	
LEGIBILIDAD	: LENG, NEST, NSTM	
AUTODESCRIPTIVO	: COMF	

GRAFICO DE CRITERIOS

CRITERIO	CLASE
LEGIBILIDAD	: CRÍTICO
SIMPLICIDAD	: CRÍTICO
CONCISIÓN	: ACEPTADO
TOLERANCIA	: ACEPTADO
AUTODESCR.	: RECHAZADO



REESTRUCTURACION

- ✍ *Nivel de análisis: se transforman los modelos de análisis en otros más comprensibles*
- ✍ *Nivel de diseño: se transforman unos modelos de diseño en otros*
- ✍ *Nivel de implementación: las representaciones obtenidas pueden enfocarse tanto a datos como a procesos*

REESTRUCTURACION DE DATOS

- ✍ *Un mismo dato puede nombrarse de distintas formas en un sistema (sinonimia).*
- ✍ *Un mismo dato puede definirse varias veces de forma diferente en un sistema*

VENTAJAS

- ▣ *Mejorar la comprensión de los sistema*
- ▣ *Incrementar la productividad del personal encargado del mantenimiento*
- ▣ *Mejorar la documentación sobre los datos y forzar estándares sobre su utilización*
- ▣ *Preparar el sistema antes de derivar modelos de diseño mediante técnicas de ingeniería inversa*
- ▣ *Facilitar la reutilización de los datos para otras aplicaciones*

HERRAMIENTA DE REESTRUCTURACION DE DATOS

ENTRADA

- Código fuente
- JCL
- Módulos
- Copias
- Descripciones de datos

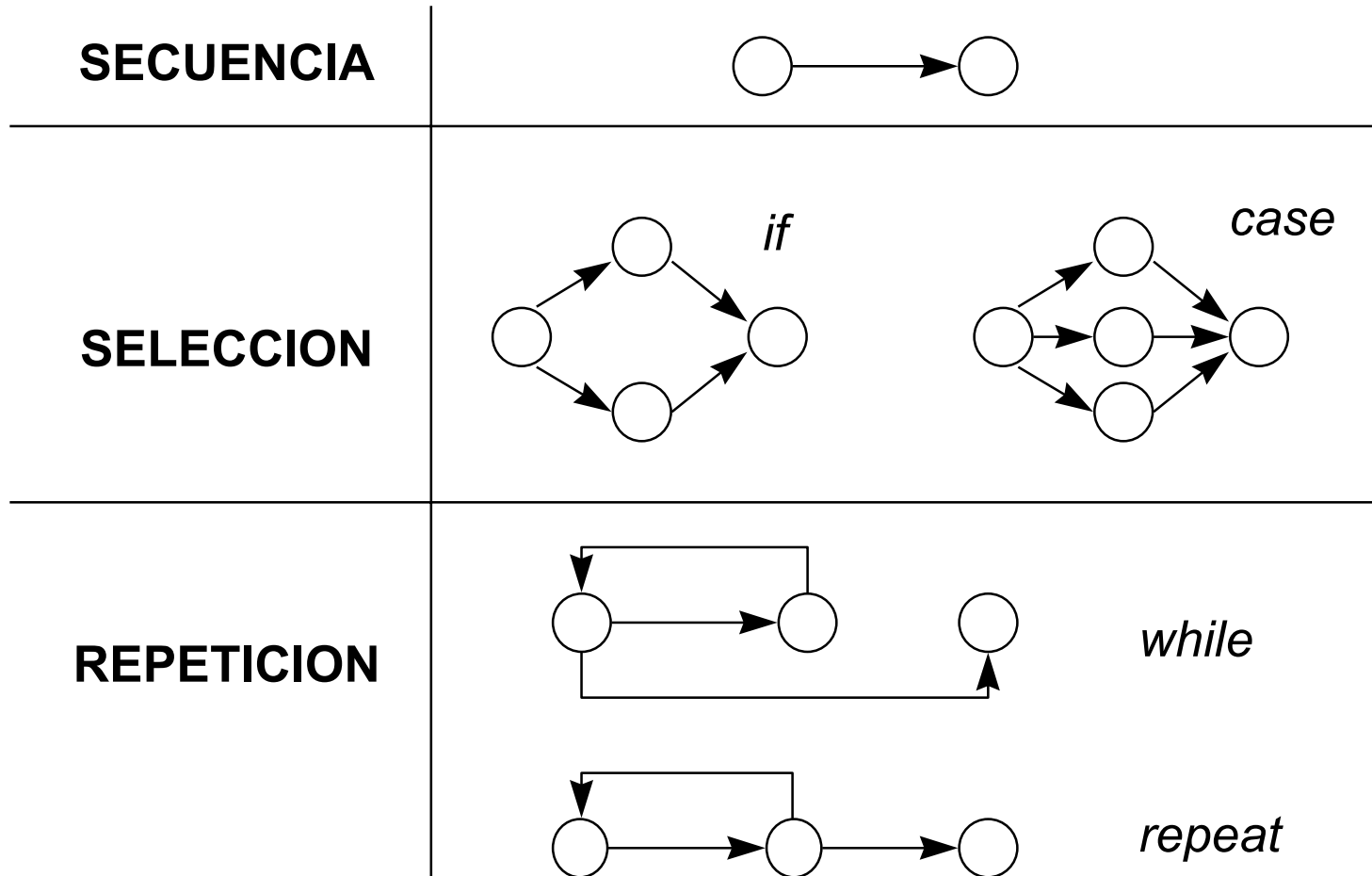


SALIDA

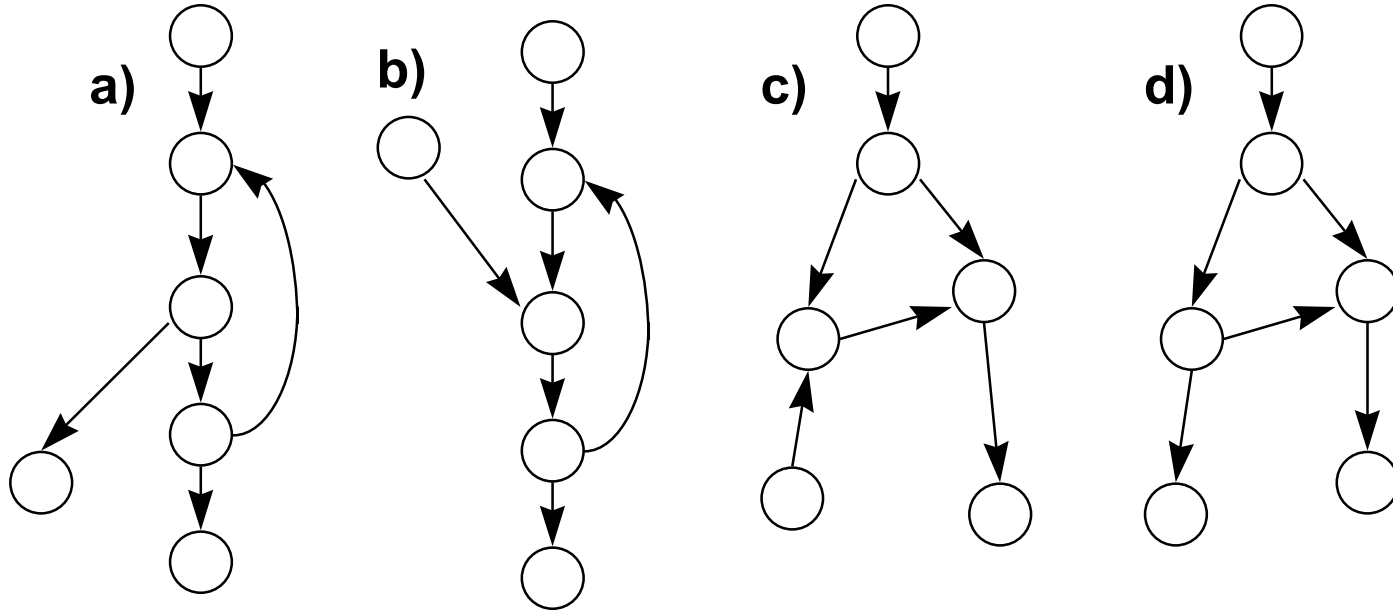
- Nombres estándar
- Código revisado
- Nuevas copias
- Referencias cruzadas

REPOSITORIO CASE

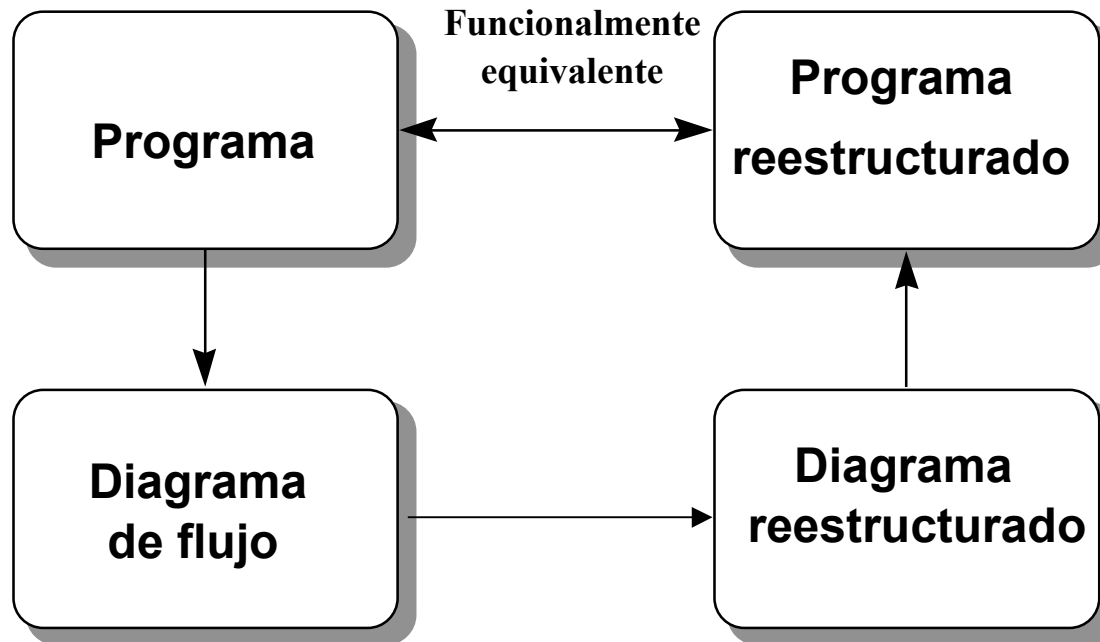
REESTRUCTURACION DE PROCESOS



CONSTRUCCIONES NO PERMITIDAS EN UN PROGRAMA ESTRUCTURADO

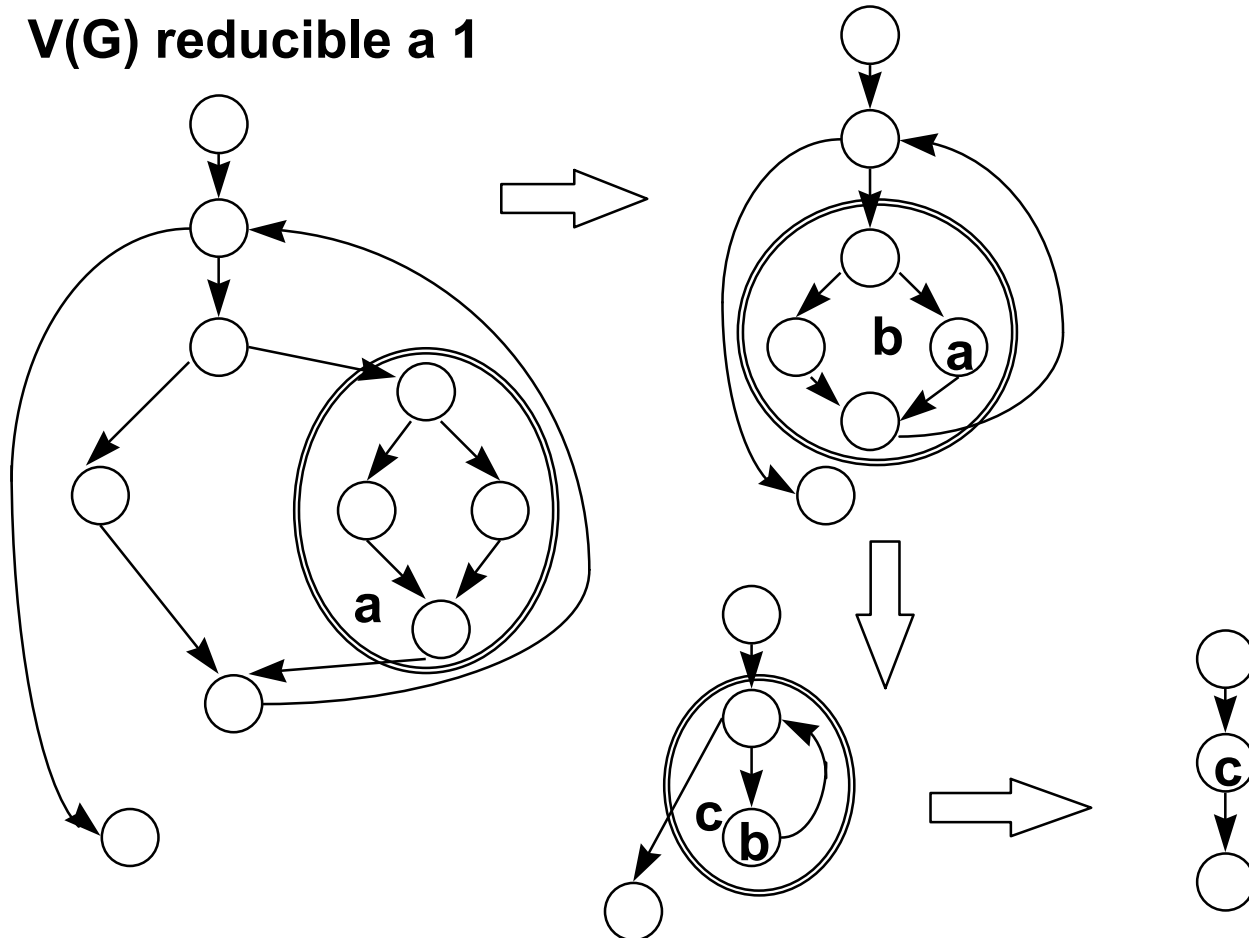


PROCESO NORMALMENTE EFECTUADO POR LAS HERRAMIENTAS DE REESTRUCTURACION DE LA LOGICA DEL PROGRAMA



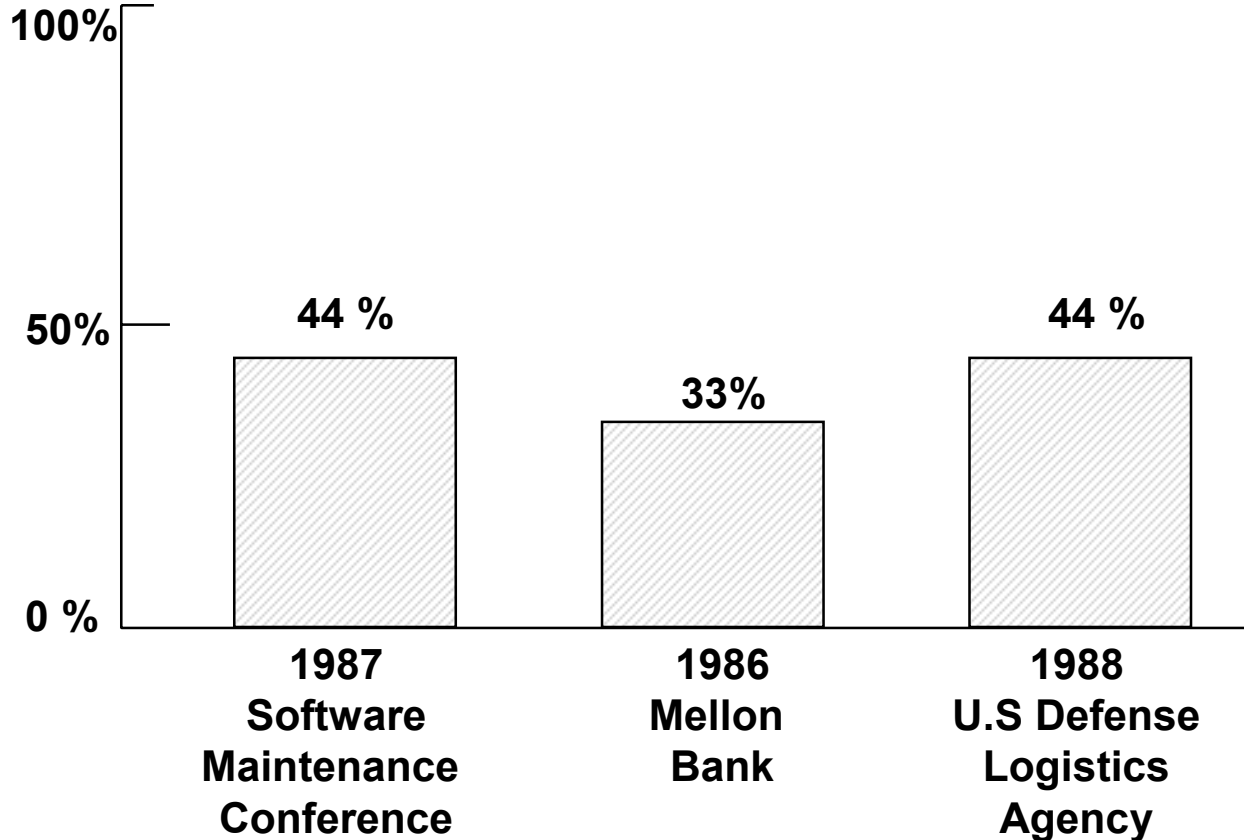
UN PROGRAMA ES ESTRUCTURADO SI
LA COMPLEJIDAD CICLOMATICA
ES REDUCIBLE A UNO

V(G) reducible a 1



BENEFICIOS DE LA REESTRUCTURACION DE CODIGO

Reducción
esfuerzo de mantenimiento

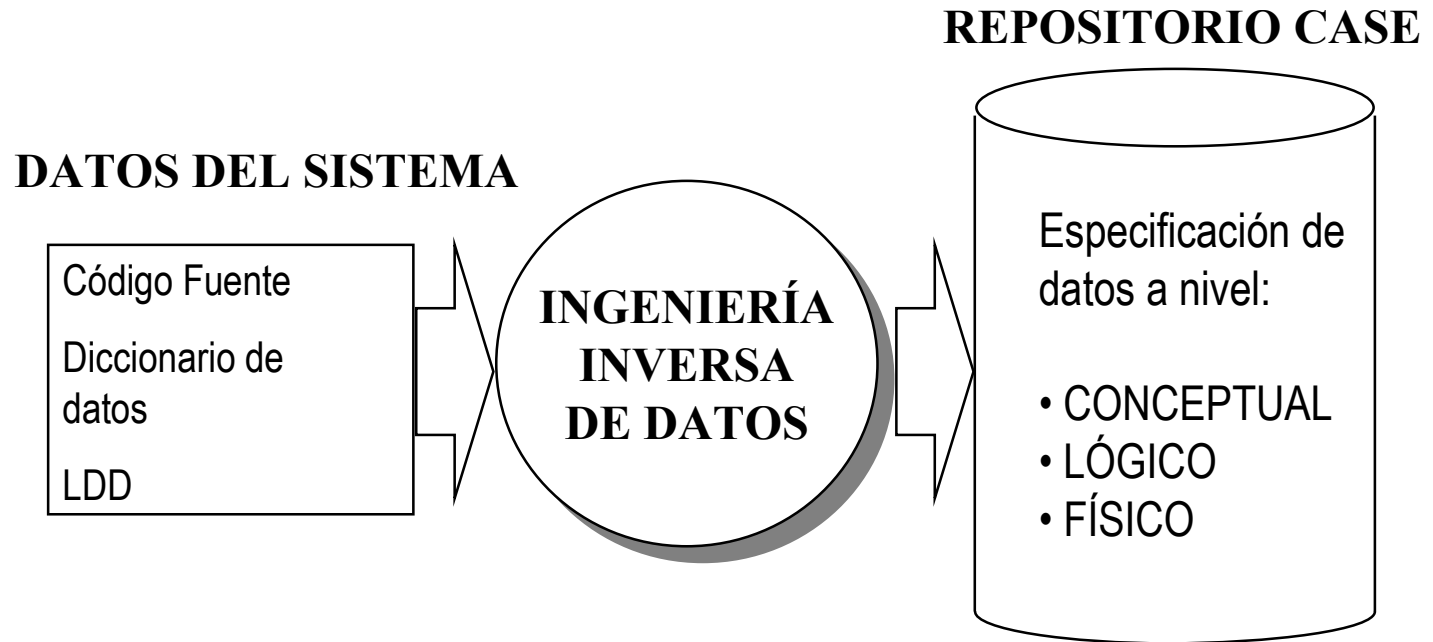


INGENIERÍA INVERSA

OBJETIVOS

- ☐ ***Detectar efectos laterales:*** *los cambios sobre un sistema pueden generar efectos laterales no deseados*
- ☐ ***Facilitar la reutilización:*** *mediante las técnicas de ingeniería inversa podemos detectar los componentes candidatos a reutilizar de sistemas existentes*

INGENIERIA INVERSA DE DATOS



INGENIERIA INVERSA DE PROCESOS

- ☑ *Modelos a nivel de análisis: como diagramas de flujo de datos (DFD) y sus correspondientes descripciones de proceso y el diccionario de datos*
- ☑ *Modelos a nivel de diseño: como los diagramas de estructuras, que representa la jerarquía de llamadas a los módulos*
- ☑ *Modelos de interfaz de usuario: jerarquías de menús y pantallas*