

Ingeniería de Software

Retroingeniería - Reingeniería

Introducción

¿Porqué hace falta cambiar el software ?



Aparecen nuevos requerimientos (Negocio)

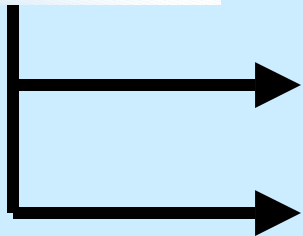


Cambia el software para corregir errores.



Mejorar el desempeño

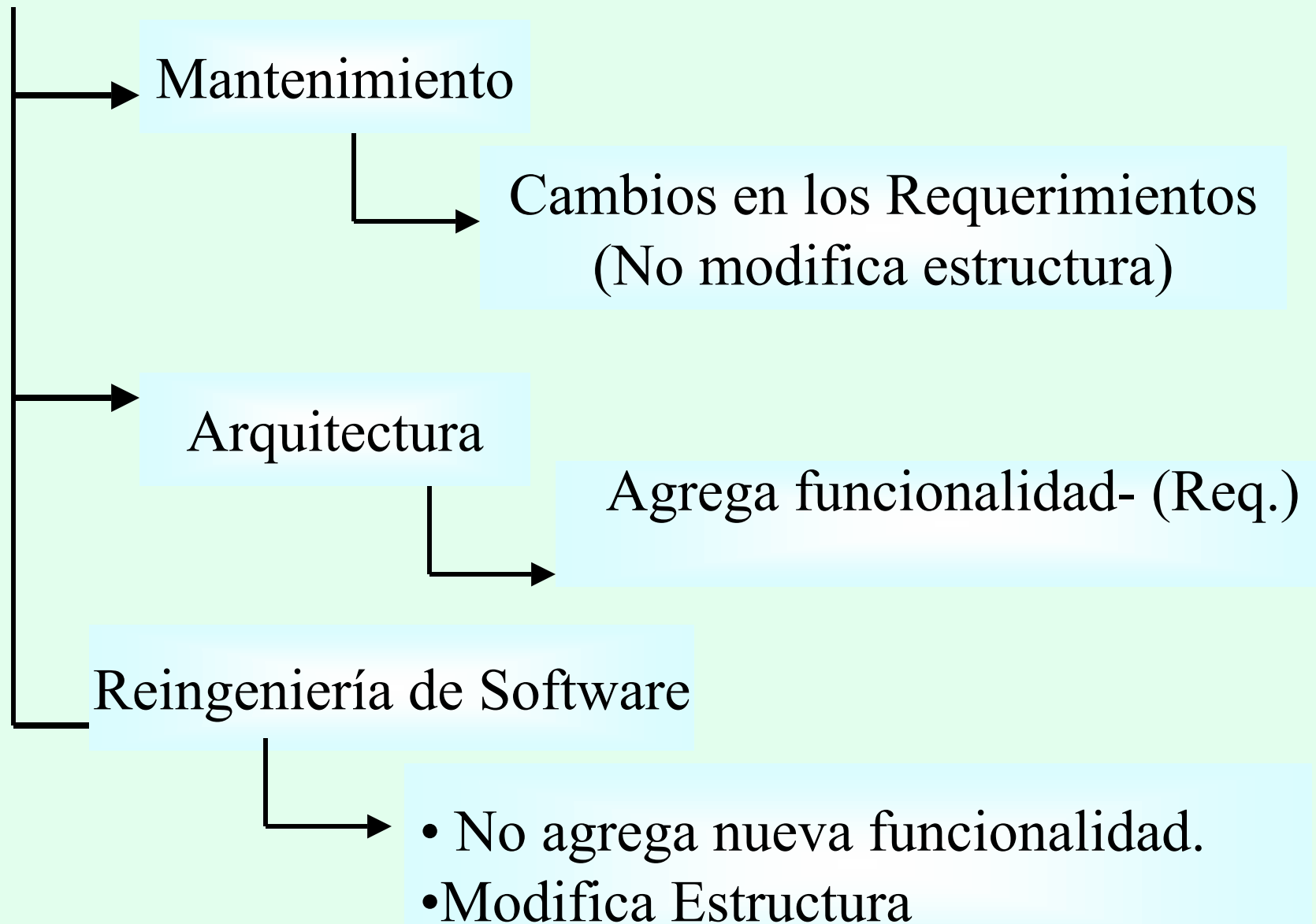
Importancia



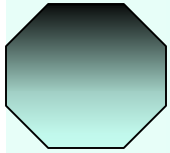
Negocio

Problema con los cambios

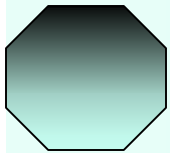
Estrategias para cambiar el software



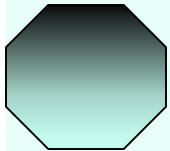
Qué es la Reingeniería?



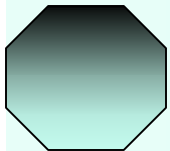
Es el proceso de reorganizar y modificar sistemas software existentes para hacerlos más mantenibles



Se re-estructuran o re-escriben partes o la totalidad de un sistema "heredado" (legacy system) sin cambiar su funcionalidad.




Es aplicable cuando alguna de las partes de un gran sistema requiere un mantenimiento frecuente




Comprende:


- Redocumentación del Sistema.
- Organización y Reestructuración del sistema.
- Traducción a otro lenguaje
- Modificación y actualización de la Estructura/Datos

¿**Cuándo** hacer Reingeniería?

 Cuando los cambios en el sistema se hacen mayormente sobre una parte de un sistema, entonces dicha parte puede ser objeto de reingeniería.

 Cuando el soporte hardware o software queda obsoleto.

 Cuando se dispone de herramientas automáticas para reestructurar el sistema.

 Tiene sentido económico cuando tiene un alto valor en el negocio, pero el mantenimiento es costoso.

 **Perspectiva**

 **Término Reingeniería**

 **Ventajas**

REINGENIERÍA- Definición

Reingeniería es:

la transformación sistemática de un Sistema existente a una nueva forma para realizar mejoras de la calidad en operación, capacidad del sistema, funcionalidad, rendimiento o capacidad de evolución a bajo coste, con un plan de desarrollo corto y con bajo riesgo para el cliente.

Ingeniería y Reingeniería

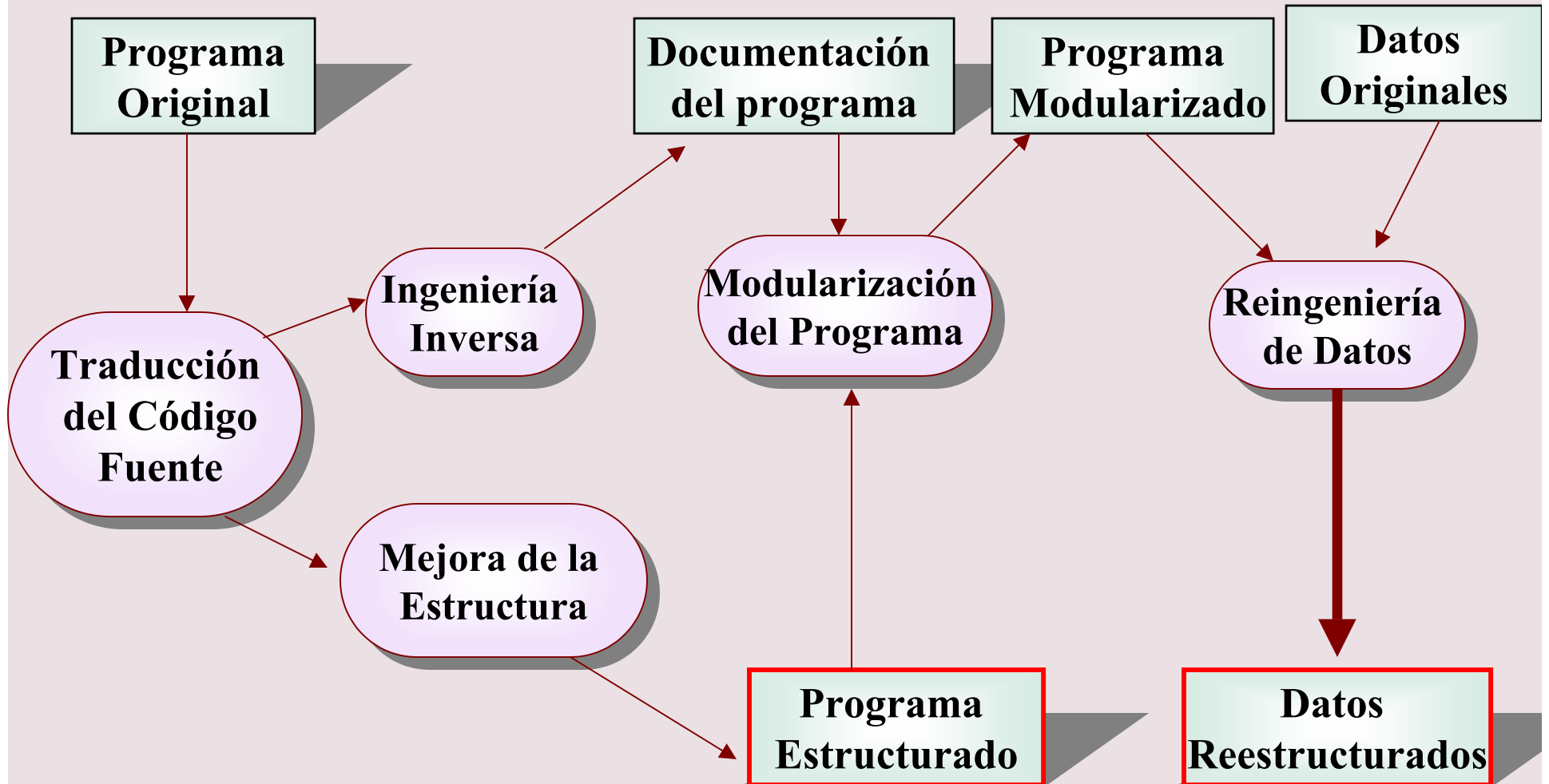
Aplicación de la Ingeniería



Aplicación de Reingeniería



El Proceso de Reingeniería



Factores de Costo de Reingeniería



Calidad del software que necesita reingeniería.



Herramientas disponibles.



Cantidad de datos que requieren una conversión.

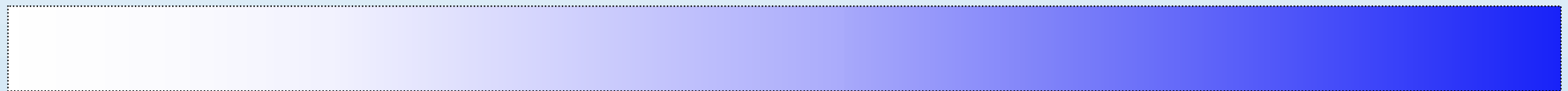


Disponibilidad de personal con experiencia.

APROXIMACIONES

**Reestructuración
Automatizada del programa**

**Reestructuración del
programa y de los datos**



**Conversión
Automatizada del
Código Fuente**

**Reestructuración
Automatizada con cambios en
la documentación**

**Reestructuración mas
los cambios
arquitectónicos**

Incremento del costo

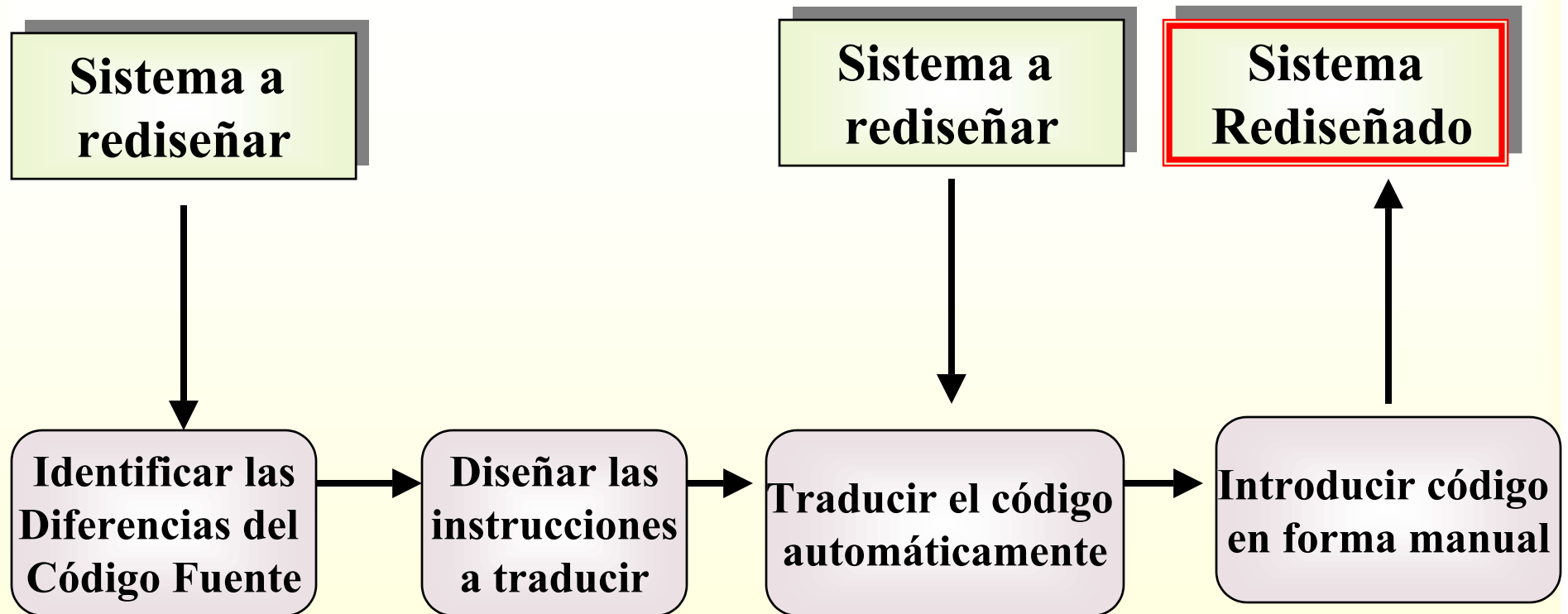
Traducción de Código

Se trata de convertir el código desde un lenguaje (o una determinada versión de un lenguaje) a otro (por ejemplo desde Fortran a C).

Puede ser necesario debido a:

- Actualizaciones en la plataforma hardware
- Escasez de personal idóneo.
- Cambios en la política de la organización
- Ausencia de software de soporte
- Solamente es “realista” si se dispone de un traductor

Proceso de Traducción de Código



RETROINGENIERÍA-Ingeniería Inversa

Orígenes –Concepto → Hardware

Definición 1: Proceso de recuperación de Diseño

Definición 2: Proceso que analiza un programa en un
esfuerzo de crear una representación
del programa a mayor nivel de abstracción
que el código fuente

Las herramientas
de Ing. Inversa
extraen información

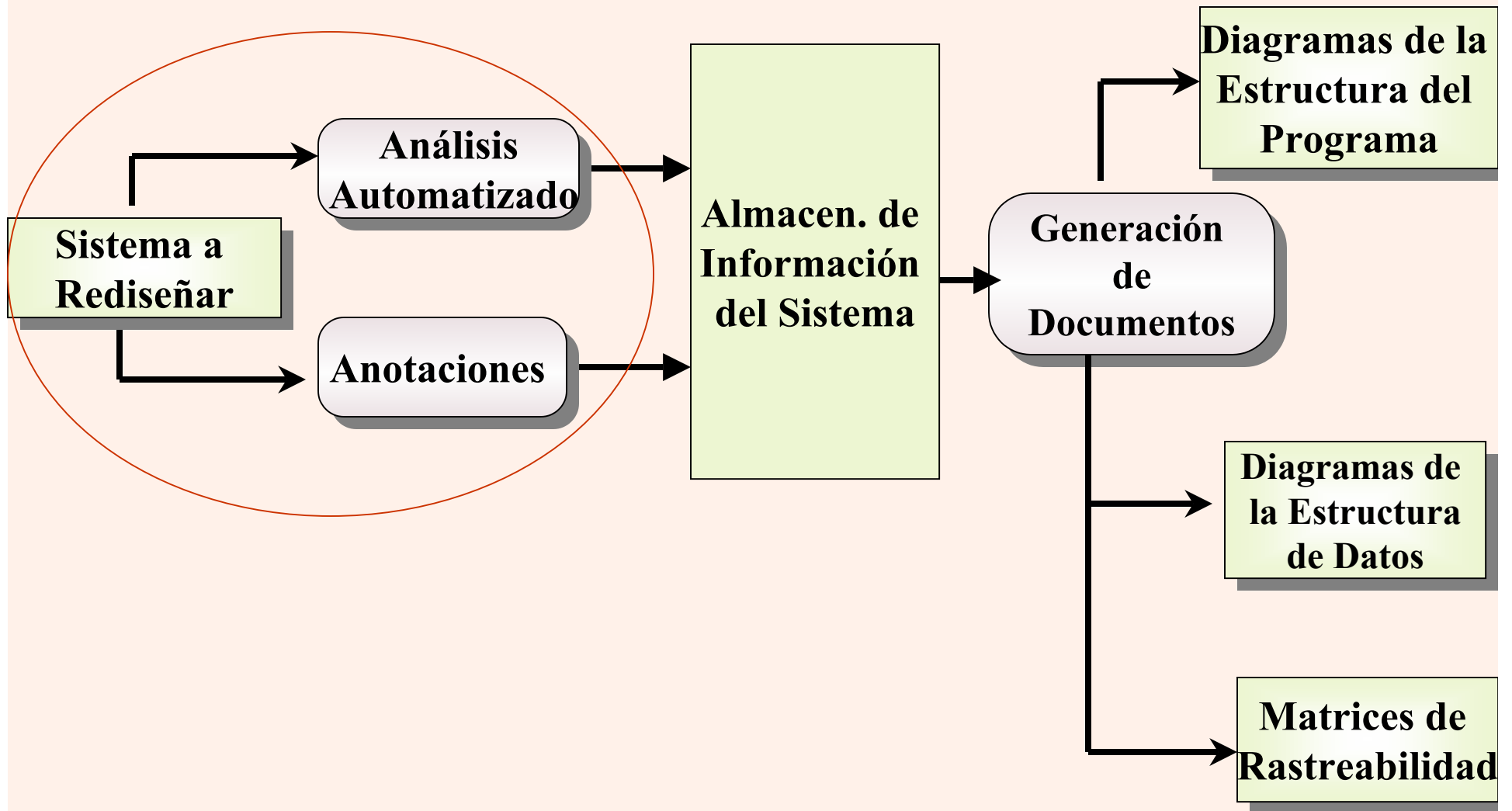


- Diseño de Datos
- Diseño Arquitectónico
- Diseño Procedimental

Ingeniería Inversa- Observaciones

- Analiza el software con el objetivo de recuperar su diseño y especificación. **La funcionalidad del software no cambia. (El programa no cambia).**
- Puede usarse como parte del proceso de reingeniería o también puede utilizarse para reespecificar el sistema para su re-implementación
- Se pueden utilizar herramientas de apoyo tales como *browsers*, generadores de referencias cruzadas, etc.
- No siempre participa de la reingeniería.

Proceso de la Retroingeniería



Mejora de la estructura del programa

- ❑ Sucesivos mantenimientos tienden a corromper la estructura del programa. Éste cada vez se vuelve más difícil de comprender.(Cond- Acciones).
- ❑ El programa puede reestructurarse automáticamente para eliminar los saltos incondicionales.
- ❑ Las condiciones pueden simplificarse para hacerlas más legibles.

Lógica Spaghetti

```
Start: Get (Time-on, Time-off, Time, Setting, Temp, Switch)
if Switch = off goto off
if Switch = on goto on
goto CntrlId
off: if Heating-status = on goto Sw-off
goto loop
on: if Heating-status = off goto Sw-on
goto loop
CntrlId: if Time = Time-on goto on
if Time = Time-off goto off
if Time < Time-on goto Start
if Time > Time-off goto Start
if Temp > Setting then goto off
if Temp < Setting then goto on
Sw-off: Heating-status := off
goto Switch
Sw-on: Heating-status := on
Switch: Switch-heating
loop: goto Start
```


Lógica de control estructurada

loop

-- The Get statement finds values for the given variables from the system's - environment.

Get (Time-on, Time-off, Time, Setting, Temp, Switch) ;

case Switch of

when On => **if** Heating-status = off **then**

Switch-heating ; Heating-status := on ;

end if ;

when Off => **if** Heating-status = on **then**

Switch-heating ; Heating-status := off ;

end if;

when Controlled =>

if Time >= Time-on **and** Time <= Time-off **then**

if Temp > Setting **and** Heating-status = on **then**

Switch-heating; Heating-status = off;

elsif Temp < Setting **and** Heating-status = off **then**

Switch-heating; Heating-status := on ;

end if;

end if ;

end case ;

end loop ;

Simplificación de Condiciones

Las condiciones complejas se pueden simplificar como parte del Proceso de Reestructuración del Programa.

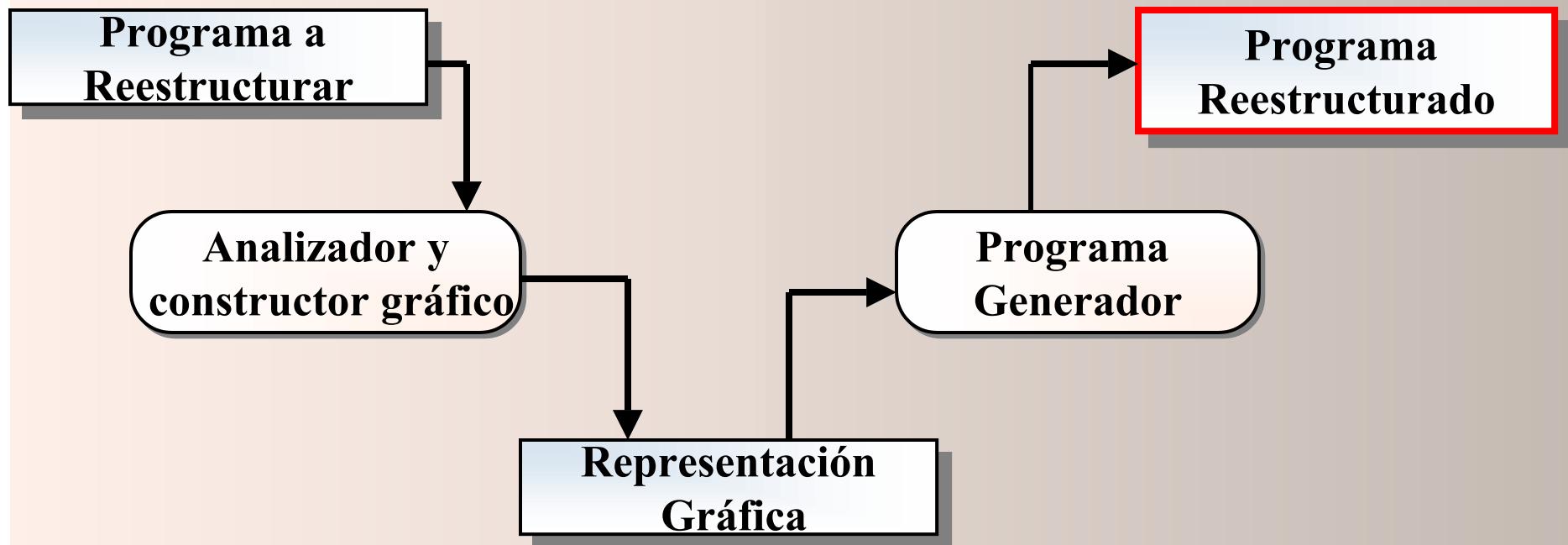
Condición Compleja

if not (A > B and (C < D or not (E > F)))...

Condición Simplificada

if (A <= B and (C>= D or E > F)...

Reestructuración automática de programas



Problemas con la reestructuración

Algunos de los mas significativos son:

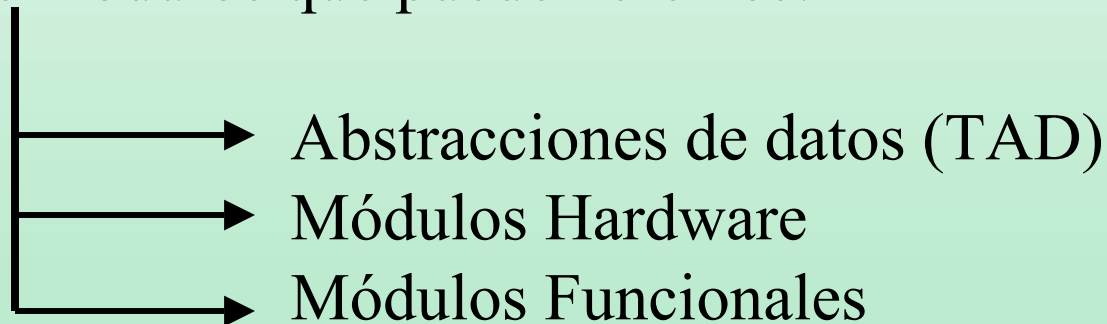
- Pérdidas de comentarios
- Pérdidas de Documentación
- Requerimientos de computación muy elevados
- **La reestructuración no ayuda si la modularización es pobre y los componentes relacionados están disperso por todo el código.**
- **La comprensibilidad de los programas conducidos por los datos puede no mejorarse con la reestructuración.**

Para reestructurar tener en cuenta.....

Modularización de Programas

- Es el proceso de reorganizar el programa para que las partes del mismo que estén relacionadas se agrupen en un mismo módulo.
- El objetivo es la optimización de sus interacciones y la simplificación de su interfaz con el resto del programa.
- Normalmente se lleva a cabo de forma manual mediante inspecciones de programas seguidos de una reorganización del código.

- Tipos de módulos que pueden crearse:



Recuperación de Abstracciones de Datos

- Muchos sistemas heredados utilizan tablas compartidas y datos globales para ahorrar espacio de memoria.
- Cambiar estas áreas es muy costoso --> Modularización se basa en identificar abstracciones de datos.

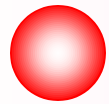
Los datos globales compartidos pueden convertirse a objetos o TAD's:

Analizar áreas de datos comunes para identificar abstracciones lógicas.

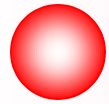
Creación del TAD u objeto para dichas abstracciones.

Utilizar una herramienta automática para generar todas las referencias a los datos y sustituirlas por las funciones adecuadas.

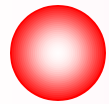
Reingeniería de Datos



Es el proceso de analizar y reorganizar las estructuras de datos (y a veces los valores de los datos) en un sistema para hacerlo mas comprensible.



El objetivo es crear un entorno gestionado de datos .



Puede formar parte del proceso de migración de datos de un sistema basado en ficheros a un sistema basado en un gestor de base de datos o de un sistema de bases de datos a otros.

Algunas razones para realizar reingeniería de datos son :



Degradación de los datos

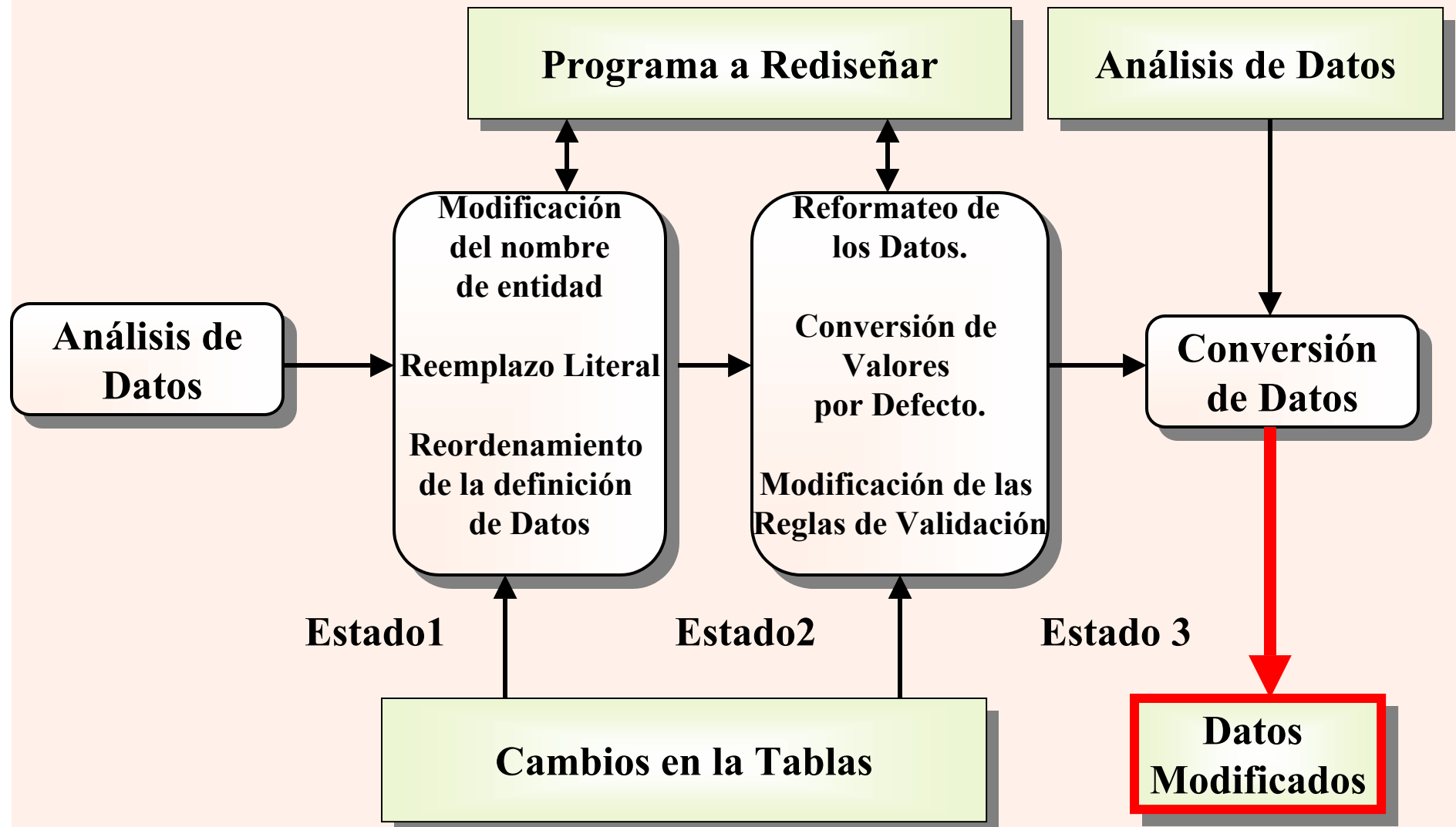


Variación de los límites del volumen de datos a manejar



Evolución de la arquitectura

Proceso de Reingeniería de Datos



Puntos Claves

El objetivo de la reingeniería es mejorar la estructura del programa para hacerlo mas comprensible y mantenible.

El proceso de reingeniería incluye: traducción de código, ingeniería inversa , mejora de la estructura del programa, modularización del programa y reingeniería de datos.

El objetivo de la ingeniería inversa es derivar el diseño o especificación de un sistema a partir del código fuente (o a veces, a partir del ejecutable)

La mejora de la estructura del programa incluye la supresión de saltos incondicionales y mejora de condiciones.

Los costos de la reingeniería de datos crecen significativamente si los datos existentes tienen que convertirse a un nuevo formato