

2.5 Cohesión

2.5.1 ¿Qué es la cohesión?

La cohesión se puede definir como la medida de la fuerza o relación funcional de los elementos de un módulo, entendiendo por elementos a la sentencia o grupo de sentencias que lo componen, a las definiciones de datos o a las llamadas a otros módulos.

Un módulo coherente ejecuta una tarea sencilla en un programa o procedimiento y requiere poca interacción con otros procedimientos que se ejecuten en otras partes del programa.

Un módulo coherente sólo debe hacer (idealmente) una cosa. El objetivo que se intenta conseguir es obtener módulos con una alta cohesión.

La cohesión se estudia a nivel de módulo, el sistema total no tiene cohesión.

Asegurar que los módulos tienen una buena cohesión es la mejor manera de minimizar el acoplamiento.

No es un criterio matemático, es decir, no se puede saber exactamente que cohesión tiene un módulo sino más o menos.

2.5.2 Relación entre acoplamiento-cohesión

Matemáticamente no se puede establecer una correlación entre el acoplamiento y la cohesión. Para ver que hay cierta relación entre ambos vamos a ver un ejemplo:

Supongamos dos ciudades A y B donde existen factorías de determinada actividad, de la factoría A nos interesa información referente a las viviendas de sus empleados mientras de la factoría B nos interesa información referente a su planta de embalaje. Tenemos dos posibles soluciones a la hora de agrupar estas actividades en módulos, la primera de ellas consistiría en realizar esta agrupación en función de las fábricas por un lado y de la información específica que se requiere de cada una de éstas por otro (figura 1), la segunda de ellas consistiría en realizar esta agrupación en función de las actividades de cada una de las ciudades (figura 2). Veamos gráficamente como serían cada una de estas soluciones.

Obsérvese, en la figura 1, que realmente no existe ninguna relación entre la factoría A y la factoría B, lo mismo ocurre con las actividades del módulo de la derecha. Nos

encontramos ante una solución que tiene poca cohesión y gran acoplamiento, lo cual se traduce en un tráfico intenso entre ambos módulos.

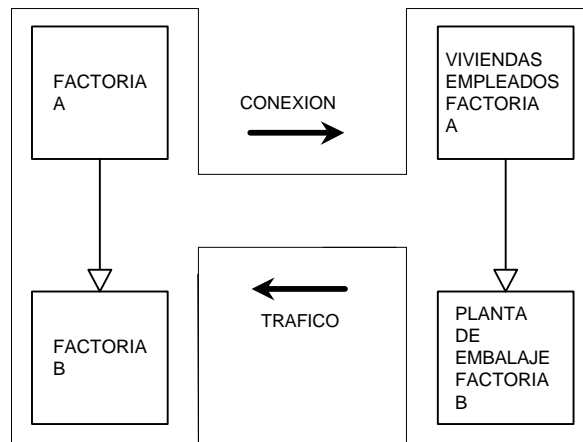


figura 1: Relación entre acoplamiento y cohesión. Solución A

Otra solución:

Obsérvese, en la figura 2, que las actividades de ambos módulos ahora si que están relacionadas, los módulos tienen mejor cohesión y mas bajo acoplamiento, es decir, existe menos tráfico entre ellos.

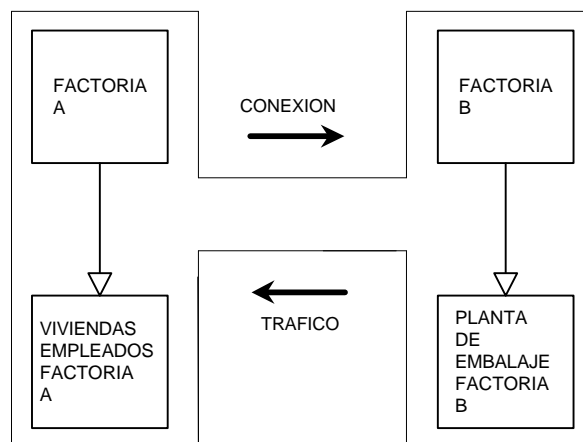


figura 2: Relación entre acoplamiento y cohesión. Solución B

Sirva el ejemplo para concienciar de la importancia que tiene la agrupación correcta de las actividades de un sistema en módulos pues de ello va a depender en gran medida el buen funcionamiento y/o rendimiento de éste.

Acoplamiento y Cohesión están íntimamente relacionados en este sentido.

2.5.3 Escala de cohesión

Fue introducida en los años 70 por L. Constantine, Yourdon, Stevens y Myers.

Mejor mantenimiento (Cohesión más fuerte)

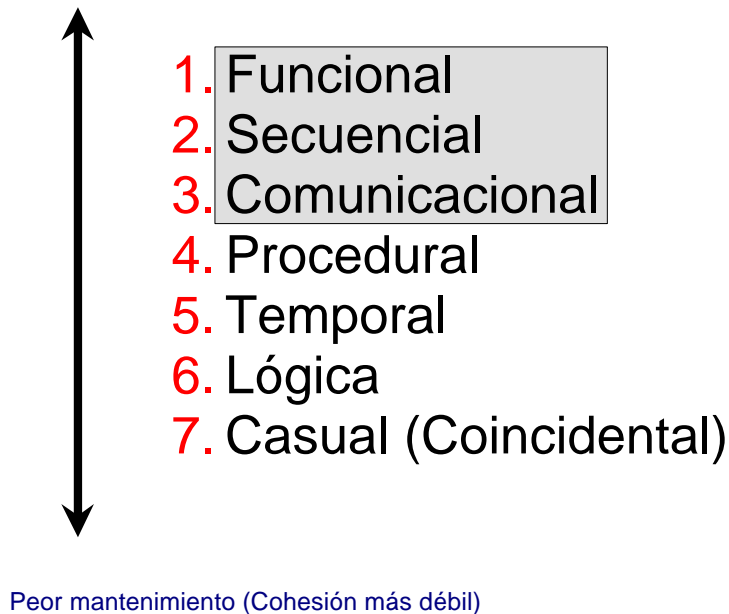


figura 3: Escala de cohesión

La escala de cohesión no es lineal. Esto significa que una cohesión baja es mucho *peor* que la de rango medio, la cual es casi tan *buen*a como una gran cohesión.

Esta tabla sirve para determinar un buen mantenimiento y una buena modificabilidad. No se puede determinar exactamente que cohesión tiene un módulo.

Lo conveniente es que la mayor parte de los módulos tengan un nivel aceptable de cohesión (es lo que se considera como sistema bueno).

2.5.4 Tipos de Cohesión

2.5.4.1 COHESIÓN FUNCIONAL

Un módulo tiene cohesión funcional si contiene elementos que contribuyen todos a la implementación de una sola función relacionada con el entorno del problema.

COHESION FUNCIONAL

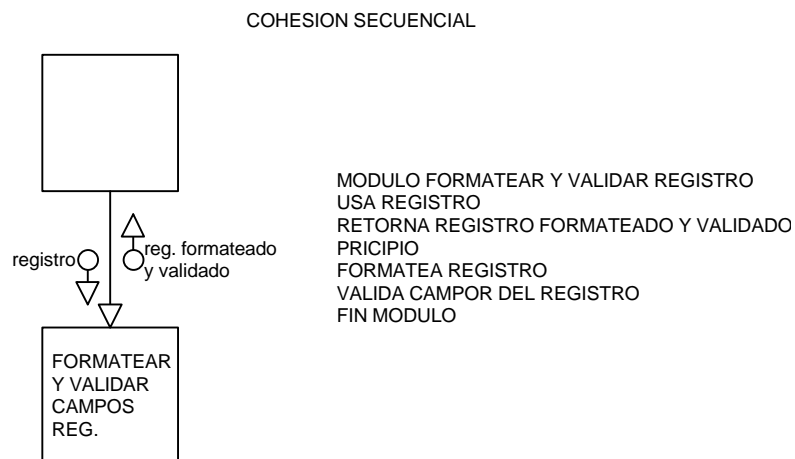
CALCULAR RAZ CUADRADA	1. LAVAR CARROCERIA
CALCULAR COSENO ANGULO	2. RELLENAR LOS HUECOS
CALCULAR PAGO NETO ASALARIADOS	3. LIJAR LA CARROCERIA
VERIFICAR CAMPO ALFABETICO	4. APLICAR UNA CAPA DE PINTURA
CALCULAR PUNTO DE IMPACTO DE MISIL	5. APLICAR CAPA FINAL DE PINTURA

figura 4: Cohesión funcional

Los sistemas construidos principalmente con acoplamiento normal y con módulos coherentes funcionalmente son los más fáciles de mantener.

2.5.4.2 COHESIÓN SECUENCIAL

Un módulo tiene cohesión secuencial si realiza distintas tareas dentro de él en secuencia, de forma que las entradas de cada tarea son las salidas de la anterior.

*figura 5: Cohesión secuencial***2.5.4.3 COHESIÓN COMUNICACIONAL**

Se dice que un módulo tiene cohesión comunicacional si realiza actividades paralelas que usan los mismos datos de entrada y/o los mismos datos de salida.

COHESION COMUNICACIONAL

1. ENCONTRAR TITULO DEL LIBRO
2. ENCONTRAR PRECIO DEL LIBRO
3. ENCONTRAR EDITORIAL DEL LIBRO
4. ENCONTRAR AUTOR DEL LIBRO

figura 6: Cohesión comunicacional

La diferencia fundamental entre la cohesión secuencial y comunicacional es que en la primera es importante el orden de las actividades.

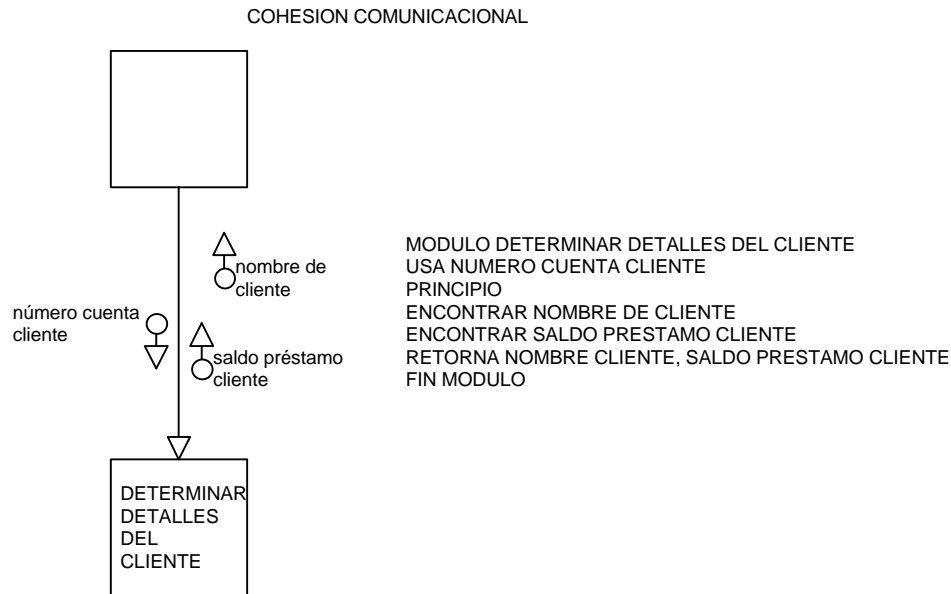


figura 7: Ejemplo de cohesión comunicacional

2.5.4.4 COHESIÓN PROCEDURAL

Un módulo tiene cohesión procedural si contiene elementos que están envueltos en actividades diferentes y posiblemente sin relacionar donde el flujo de control pasa de una actividad a la siguiente. Es importante señalar que suele haber poca relación entre los datos de entrada y salida de los módulos (sin embargo no quiere decir que siempre sea así).

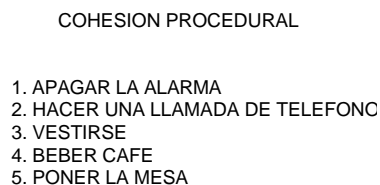


figura 8: Cohesión procedural

Veamos otro ejemplo:

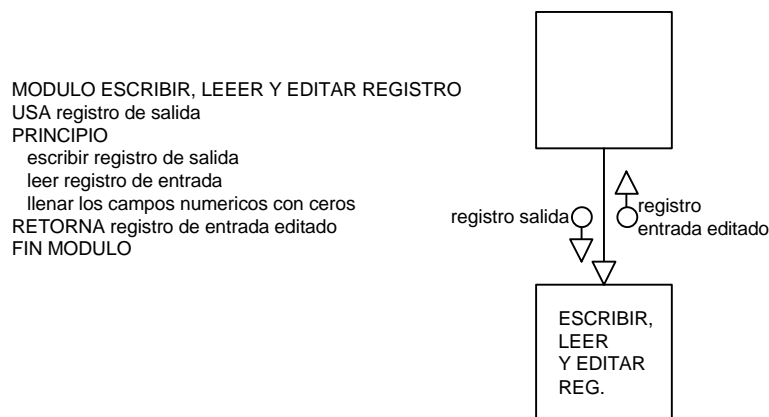


figura 9: Otro ejemplo de cohesión procedural

2.5.4.5 COHESIÓN TEMPORAL

Un módulo tiene cohesión temporal si contiene elementos que están involucrados en actividades que están únicamente relacionadas con el tiempo (decidimos que se ejecuten esas actividades en el mismo tiempo secuencialmente).

En la cohesión procedural importa el orden de las actividades sin embargo en la cohesión temporal no importa el orden, esta es la única diferencia.

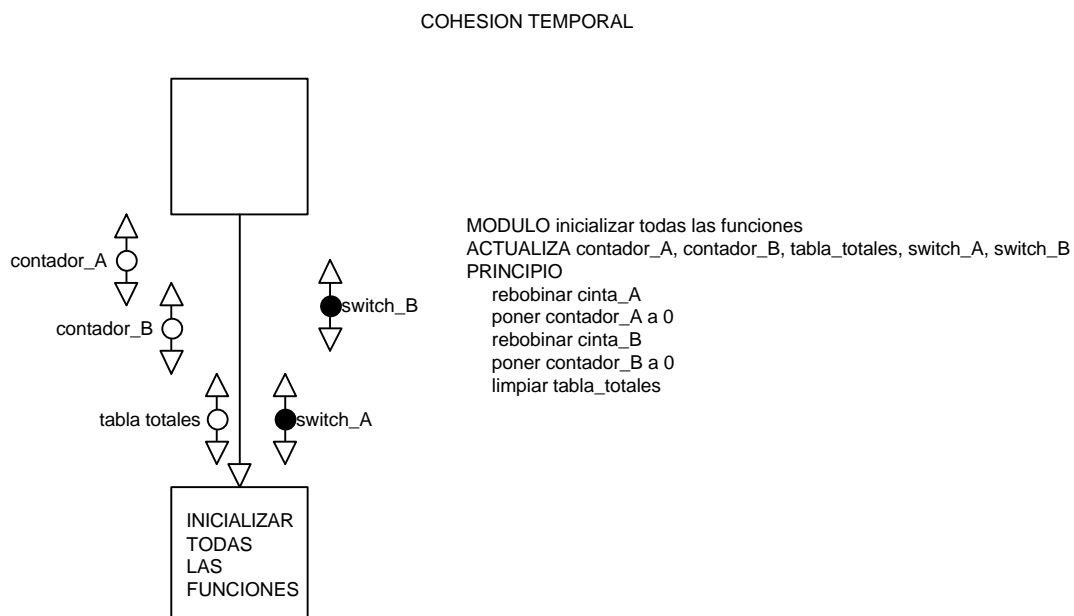


figura 10: Cohesión temporal

2.5.4.6 COHESIÓN LÓGICA

Un módulo con cohesión lógica contiene actividades de la misma categoría general donde la actividad o actividades a usar en cada momento se seleccionan desde fuera del módulo.

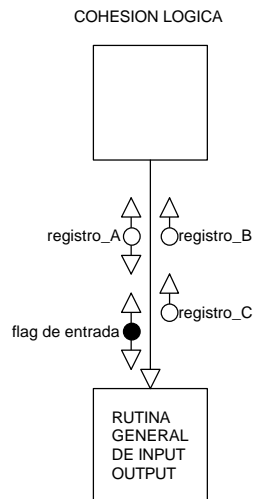


figura 11: Cohesión lógica

```

MODULO Rutina general de input/output
usa flag_de_entrada /*Elige la función a usar*/
actualiza registro_A
define registro_B de 80 caracteres
define registro_C de 120 caracteres
principio
  if flag_de_entrada = 1
  then escribir registro_A en
        nuevo_fichero_maestro
        leer fichero_1 de transacciones
        en registro_B
  elseif flag_de_entrada = 2
  then if registro_A = todo blancos
        then leer fichero_1 de transacciones
              en registro_B
        endif
        leer fichero_2 de transacciones en
        registro_C
  elseif flag_de_entrada = 3
  then leer fichero_1 de transacciones en
        registro_B
        leer fichero_1 maestro en registro_C
        leer fichero_2 maestro en registro_A
  endif
return registro_B, registro_C
fin modulo

```

2.5.4.7 COHESIÓN CASUAL

Un módulo tiene cohesión casual si contiene un grupo de actividades sin relación significativa entre ellas. Las características más frecuentes de los módulos que presentan cohesión lógica o casual son:

- ♦ difíciles de entender y difíciles de mantener.
- ♦ sus módulos jefes tienen que utilizar flags totalmente artificiales para seleccionar la parte del módulo que desean usar.

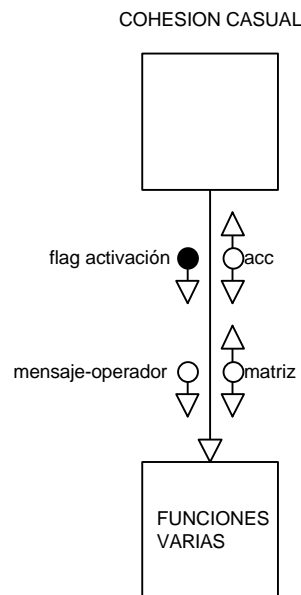


figura 12: Cohesión casual

```

MODULO Funciones Varias
usa flag_activacion, mensaje_operador
actualiza acc, matriz_numerica
principio
  if flag_activacion = 1
  then inicializar matriz_numerica a 0
        inicializar acumulador acc a 1
  else if flag_activacion = 2 or
        flag_activacion = 3
        then rebobinar cinta_A
  if flag_activacion = 3 then imprimir
        cabeceras
  else display mensaje operador
fin modulo

```

El origen de los módulos con cohesión casual o lógica ocurre cuando alguien se empeña en englobar un conjunto de instrucciones parecidas dentro de un mismo módulo pero no se van a ejecutar a la vez.

2.5.5 Determinación de la cohesión de un módulo

Podemos utilizar para determinar la cohesión:

1. Una sentencia descriptiva

Consiste en inspeccionar el módulo y tratar de escribir una sentencia que describe que hace el módulo (hay que mirar dentro de un módulo) de esta forma se puede deducir que cohesión tiene el módulo.

Los módulos con cohesión funcional están descritos por una sentencia formada por un verbo imperativo y un nombre.

Los módulos con cohesión secuencial quedan descritos mediante sentencias que contienen nombres de varias funciones.

Los módulos con cohesión comunicacional quedan descritos mediante sentencias que contienen varios nombres de funciones estando relacionadas estas funciones por el hecho de que trabajan con los mismos datos de entrada y salida.

Los módulos con cohesión procedural quedan descritos por una sentencia que contiene nombres que hacen referencia a partes de organigramas y de flujos de control.

Los módulos con cohesión temporal suelen contener referencias al tiempo.

Los módulos con cohesión lógica suelen contener nombres de propósito general.

Los módulos con cohesión casual suelen contener nombres poco significativos.

2. Un árbol de decisión

El módulo realiza una función particular del problema?

SI funcional

NO ¿Cómo están relacionadas las actividades del módulo?

DATOS ¿Es importante la secuencia?

SI secuencial

NO comunicacional

FLUJO DE CONTROL ¿Es importante la secuencia?

SI procedural

NO temporal

NADA ¿Están las actividades en la misma categoría general?

SI lógica

NO coincidental

3. Reglas de cadenas en serie

Para establecer una partición de un módulo en actividades es bastante subjetivo. Las actividades de un módulo muestran distintos niveles de cohesión entonces se aplica el nivel de cohesión mas bajo.

El módulo tendrá la cohesión de nivel más bajo entre el nivel 1 y el nivel 2.

4. Reglas de cadenas en paralelo

Si todas las actividades de un módulo están relacionadas por más de un nivel de cohesión, entonces el módulo tiene la cohesión del nivel más fuerte.

2.5.6 Factores a los que afecta la Cohesión

Nivel de Cohesión	Acoplamiento	Facilidad Implantación	Comprensib.	Modificab.	Mantenimiento Sistema
FUNCIONAL	BUENO	BUENA	BUENA	BUENA	BUENO
SECUENCIAL	BUENO	BUENA	BUENA	BUENA	BAST. BUENO
COMUNICAC.	MEDIO	MEDIA	MEDIA	MEDIA	MEDIO
PROCEDURAL	VARIABLE	POBRE	VARIABLE	VARIABLE	MALO
TEMPORAL	POBRE	MEDIA	MEDIA	MEDIA	MALO
LOGICA	MALO	MALA	MALA	POBRE	MALO
COINCIDEN.	MALO	POBRE	MALA	MALA	MALO

A continuación se ofrecen una serie de ideas prácticas que pueden ayudar a mejorar la calidad del diseño:

- ♦ hay que reducir el número de parámetros que intercambian los módulos tanto como sea posible, es mejor que los datos de comunicación que se pasen sean elementos de registros.
- ♦ hay que evitar pasar parámetros de un módulo a otro, a menos que tengan una utilidad práctica.
- ♦ no se debe compartir código entre las actividades que están incluidas en un módulo. Esto hace difícil modificar una actividad sin cambiar la otra.
- ♦ se debe parar la descomposición cuando no se encuentren funciones bien definidas, pues no se debe crear un módulo con una serie de líneas de código agrupadas aleatoriamente.
- ♦ se puede parar la descomposición cuando la interface con un módulo sea tan complicada como el módulo en sí mismo.
- ♦ se deberán conseguir diseños lo más equilibrados que sea posible. Puede decirse que un diseño está equilibrado si y sólo si, trata con datos lógicos en su parte alta y con datos físicos en su parte baja.