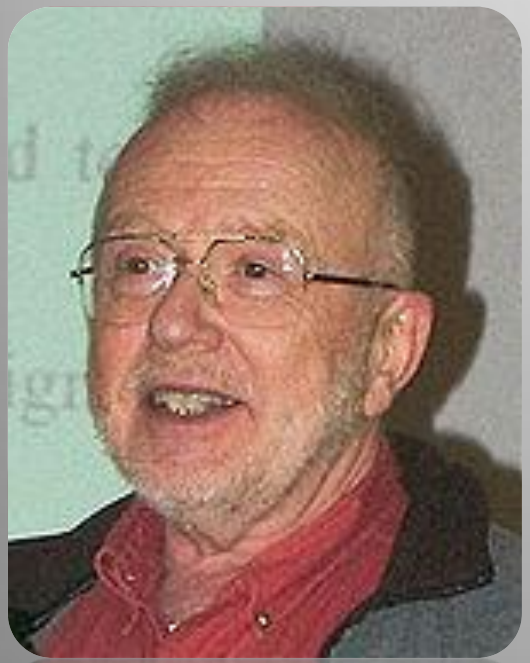
On the criteria to be used in decomposing systems into modules

David Parnas, Carnegie-Mellon University, 1972



Nacido en 1941 en Canadá.
Pionero de la ingeniería de software, que desarrolló el concepto de ocultación de información en la programación modular, elemento importante de la programación orientada a objetos en la actualidad. También es conocido por su defensa sobre la "documentación precisa".

Karina Borgna
Julián Dondero
Adrián Tamburri

¿Por qué este Paper es tan importante?

Se ataca un tema siempre interesante: la modularización de Software, en particular, cómo llevarla a cabo de manera "más eficiente o correcta" (según el autor).

Aquí Parnas provee UN criterio posible de modularización que según SU entender puede resultar favorable en la mayoría de los casos.

Se da una alternativa a los métodos convencionales de modularización.

A partir de esa alternativa, se abre un nuevo campo de estudio, nuevas maneras de pensar el Software que construimos. Es decir, contamos con una nueva herramienta que podremos utilizar o no a nuestro criterio en diversos desarrollos.

Pateó el tablero.

Comparando criterios…

La diferencia principal entre estos métodos es que en el método convencional, los módulos son considerados como subrutinas del programa. Por el contrario, en el método que propone Parnas, los módulos son asignaciones de responsabilidades.

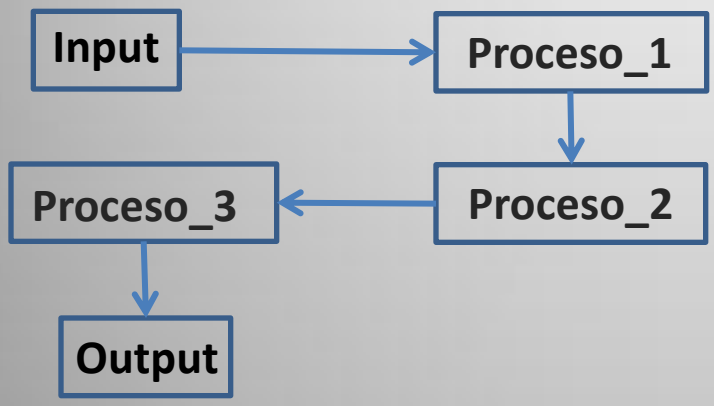
Convencionalmente, un módulo agrupa una o más subrutinas, es decir, una subrutina pertenece a un determinado módulo. Por el contrario, Parnas propone que un módulo agrupe una decisión de diseño y que las subrutinas se encuentren en n-módulos sin importar esto demasiado.

Las decisiones de diseño van más allá del tiempo de ejecución y por tanto los módulos no deben corresponderse con las etapas de procesamiento. Esto se contrapone con la idea de comenzar la modularización a partir de un diagrama de flujo que proponía el método convencional de esa época.

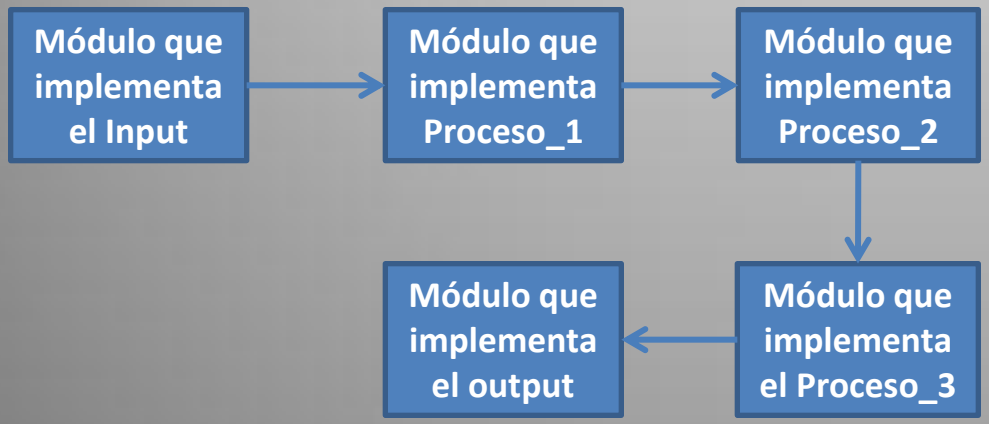
Criterios utilizados en sistemas para la descomposición en módulos – Parnas 1972

On the criteria to be used in decomposing systems into modules

Criterio convencional



Primero identifico subrutinas, subprogramas o funciones que tengo que implementar. Podría hacerse mediante un diagrama de flujo de mi aplicación.



Mi programa modularizado convencionalmente.

Criterios utilizados en sistemas para la descomposición en módulos – Parnas 1972

On the criteria to be used in decomposing systems into modules

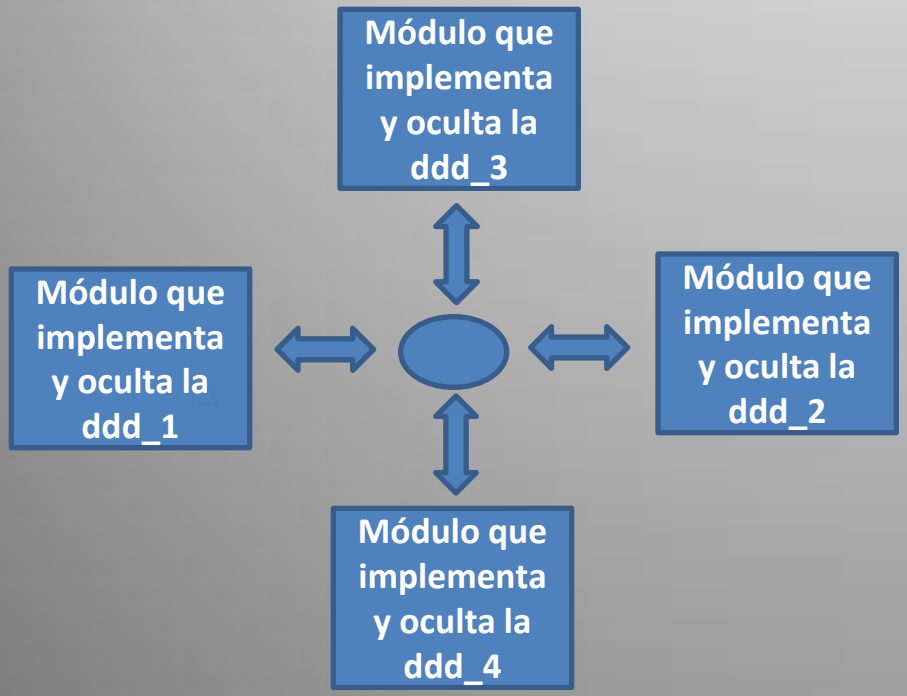
Criterio que propone Parnas basado en ocultamiento de información

Decisión de diseño_1
Decisión de diseño_2
Decisión de diseño_3
Decisión de diseño_4

Primero identifico decisiones de diseño importantes y relevantes, o que probablemente cambien.

Decisiones de diseño :

- Tipo de formato de la entrada
- Formas de almacenamiento
- Modo de encriptación de los datos
- etc.



Mi programa modularizado según criterio de Parnas.

ddd = decisión de diseño

Conclusiones y otras consideraciones importantes

Beneficios esperados de la modularización

- Menor tiempo de desarrollo: Pueden trabajar distintos grupos por separado en distintos módulos, sin necesitar demasiada comunicación.
- Flexibilidad: Debería ser posible hacer cambios drásticos a un módulo sin necesidad de cambiar el resto. (no hay dependencia)
- Comprensible: Debería ser posible conocer el sistema de a un módulo a la vez.

Como consecuencia de la modularización por decisiones de diseño, las rutinas o funciones deberían ser piezas de código susceptibles de habitar distintos módulos.

Cada módulo implementa una decisión, luego cuando el diseño sea modificado, ampliado, revisado, etc. sólo el módulo que lo contenga será el modificado.

Según Parnas, es más sencillo el entendimiento de la funcionalidad del Software que estamos construyendo si modularizamos por decisiones de diseño. (Subjetivo y dependiente)

Cada módulo ocultará a los demás módulos la decisión de diseño que implementa, proveyendo sólo lo mínimo indispensable y necesario.