

Acoplamiento

Problema 1

Análisis estructural

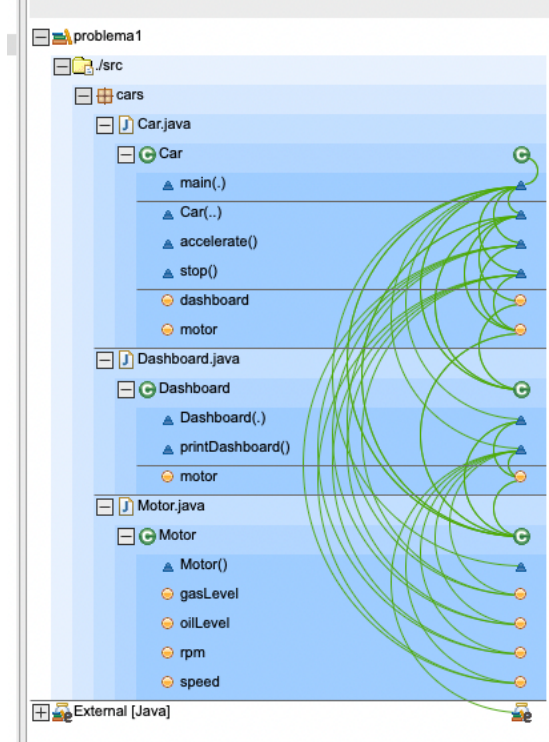
Structure	
Entangled code (%):	0.00
Critically entangled code:	0
Entangled code:	0
Relative entanglement (%):	0.00
Cycle groups:	0
Java Packages	
Unresolved cycle groups:	0
Biggest cycle group (lines of code):	0
Maintainability level:	98.00
Propagation cost:	66.67
System ACD:	2.00
Highest module ACD:	2.00
Ignored entangled code (%):	0.00
Critically ignored entangled code:	0
Entangled ignored code:	0
Ignored cycle groups:	0
To be fixed entangled code (%):	0.00
Critically to be fixed entangled code:	0
Entangled to be fixed code:	0
To be fixed cycle groups:	0
Components	
Cyclic Java packages:	0
Structural debt index:	0
Component dependencies to remove:	0
Parser dependencies to remove:	0
Ignored cyclic Java packages:	0
Cyclic components:	0
Structural debt index:	0
Component dependencies to remove:	0
Parser dependencies to remove:	0
Ignored cyclic components:	0

Según el análisis estructural de este problema podemos decir que el Average component dependency (ACD) tiene un valor de 2, lo cual nos indica que en promedio las modificaciones a un archivo del programa implican realizar modificaciones en otros 2 archivos. Teniendo en cuenta que el programa está compuesto por un paquete de 3 archivos, este número indicaría un ACD un poco alto, frente a la cantidad de archivos, más no en extensión del archivo como tal.

Al realizar el análisis de la cantidad de dependencias que presenta cada archivo en el proyecto, podemos ver que cada uno está enlazado por dos dependencias de otros archivos.

Level: Logical Programming Element				Scope: problema1 (Module)	
Metric [4]	Categories	Provider	Min	Max	Values Histogram Pie Chart
Logical Cohesion (Module)	Cohesion/Coupling	Core			Logical Cohesion (Moc
Logical Cohesion (System)	Cohesion/Coupling	Core			
Logical Coupling (Module)	Cohesion/Coupling	Core			
Logical Coupling (System)	Cohesion/Coupling	Core			
					Element [3]
					cars.Motor 2
					cars.Dashboard 2
					cars.Car 2
					Values: 3 Arithmetic average: 2.00 Lower threshold: n/a
					Minimum: 2 Standard deviation: 0.00 Upper threshold: n/a
					Maximum: 2 Median: 2.00

Diagrama de dependencias



Refactor de la clase motor

Structure		
Entangled code (%):	<div></div>	0.00
Critically entangled code:		0
Entangled code:		0
Relative entanglement (%):	<div></div>	0.00
Cycle groups:	<div></div>	0
Unresolved cycle groups:		
Unresolved cycle groups:		0
Biggest cycle group (lines of code):		0
Maintainability level:		97.00
Propagation cost:		56.25
System ACD:		2.25
Highest module ACD:		2.25
Ignored entangled code (%):	<div></div>	0.00
Critically ignored entangled code:		0
Entangled ignored code:		0
Ignored cycle groups:		0
To be fixed entangled code (%):	<div></div>	0.00
Critically to be fixed entangled code:		0
Entangled to be fixed code:		0
To be fixed cycle groups:		0
Java Packages		
Cyclic Java packages:		0
Structural debt index:		0
Component dependencies to remove:		0
Parser dependencies to remove:		0
Ignored cyclic Java packages:		0
Components		
Cyclic components:		0
Structural debt index:		0
Component dependencies to remove:		0
Parser dependencies to remove:		0
Ignored cyclic components:		0

Se puede observar que se incrementó la cohesión con respecto a la clase cars

Logical Cohesion (Module)	
Element [4]	
cars.Car	3
cars.Motor	3
cars.Dashboard	2
cars.StandardMotor	2

Al modificar la dependencia de una clase a una de una interface, se puede apreciar la disminución de las dependencias entre clases.

