* **DATA CLUMPS**

Data Clumps – Grupo de datos

Cuando existen instancias con el mismo nombre que se usan muy repetidas veces en el código, generalmente por la copia y pega de clases o métodos.

Es un problema debido a que, al momento de revisar el código, se tiende a confundir con los datos que se le pueden pasar como parámetros a algún método.

Las técnicas de refactorización que pueden aplicarse para la solución de este mal olor son:

* Extract class
* Introduce parameter object
* Preserve whole object
* **LONG CLASS**

Long class – clase extensa.

Cuando una clase contiene demasiados atributos, métodos o simplemente líneas de código. Es un problema debido a que lo vuelve poco flexible e ineficaz

Las técnicas de refactorización que pueden aplicarse para la solución de este mal olor son:

* Extract class
* Extract method
* Extract inferface
* Duplicated observed data
* **SWITCH STATEMENTS**

Switch stataments – Cambios de declaraciones

Cuando se tienen un abuso de la palabra reservada switch o un muchos if anidados.

Las tecnificas de refactorización que pueden aplicarse para la solución de este mal olor son:

* Extract method luego Move Method
* Replace type code with subclasses
* Replace conditional with polymorphism
* Introduce null Objects
* **REFUSED BEQUEST**

Refused bequest – Legado Rechazado

Cuando una clase usa métodos o atributos de otra clase sin haber herencia

Las técnicas de refactorización que pueden aplicarse para la solución de este mal olor son:

* Replace inheritance with delegation
* Extract superclass
* **SHOTGUN SURGERY**

Shotgun surgery – Cirugía de escopeta

Cuando se hace una modificación de alguna clase, método o atributo, se va a necesitar hacer un cambio a otras partes del código donde se haya usado dicha clase, método o atributo

Las técnicas de refactorización que pueden aplicarse para la solución de este mal olor son:

* Move method
* Move field
* Inline class