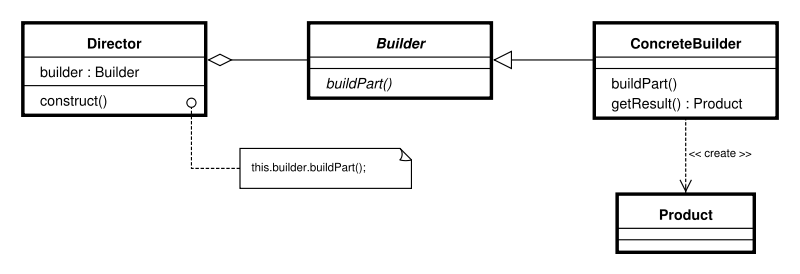
**Constructor (Builder)**

*Nombre:* Builder  
  
*Problema y Contexto:*   
  
Un único proceso de construcción debe ser capaz de construir distintos objetos complejos, abstrayéndonos de los detalles particulares de cada uno de los tipos.  
  
Se aplica cuando:

* Nuestro sistema trata con objetos complejos (compuestos por muchos atributos) pero el número de configuraciones es limitada.
* El algoritmo de creación del objeto complejo puede independizarse de las partes que lo componen y del ensamblado de las mismas.

*Solución y Estructura:*  
  
La solución será crear un constructor que permita construir todos los tipos de objetos, ayudándose de constructores concretos encargados de la creación de cada tipo en particular. Un objeto director será el encargado de coordinar y ofrecer los resultados.  
  
La estructura es la siguiente:  
  
  
  
Donde:  
  
Director: Se encarga de construir un objeto utilizando el Constructor (Builder).  
  
Builder: Interfaz abstracta que permite la creación de objetos.  
  
Concrete Builder: Implementación concreta del Builder definida para cada uno de los tipos. Permite crear el objeto concreto recopilando y creando cada una de las partes que lo compone.  
  
Product: Objeto que se ha construido tras el proceso definido por el patrón.

Consecuencias:

* POSITIVAS:
  + Reduce el acoplamiento.
  + Permite variar la representación interna del objeto, respetando la clase builder. Es decir, conseguimos independizar la construcción de la representación.
* NEGATIVAS:
  + Introduce complejidad en los programas.

*Patrones Relacionados:* Método Factoría (Factory Method)