Patrón de diseño: Builder

José Jahir Pérez Sánchez

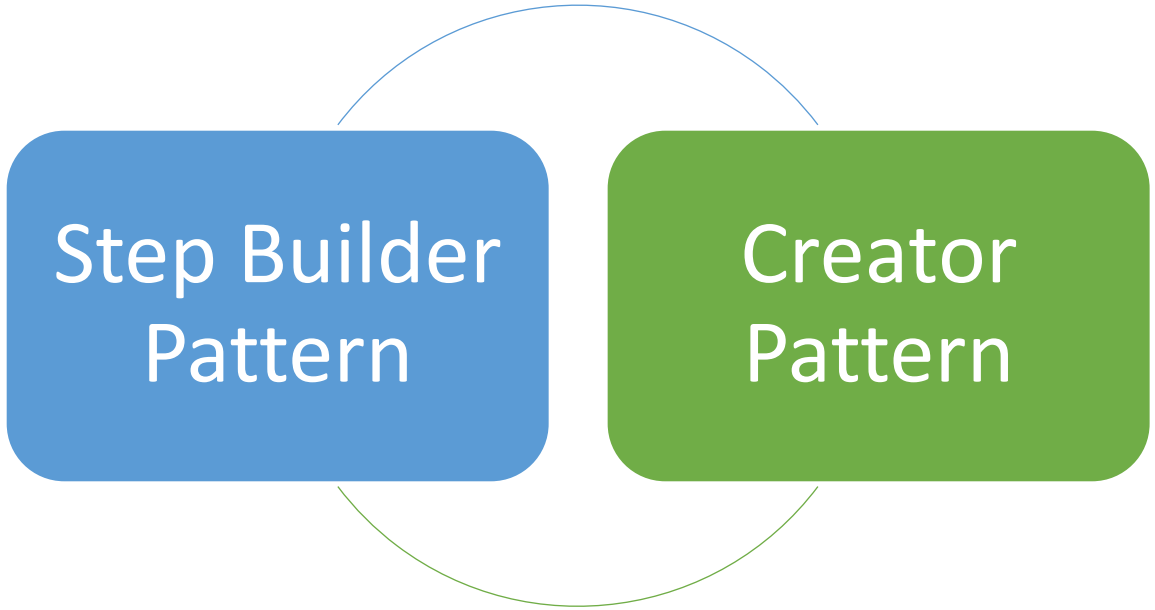
Builder

- El patrón de diseño Builder pertenece al tipo de patrón de creación, estos patrones se utilizan cuando debemos crear objetos pero debemos tomar decisiones dinámicamente en el proceso de creación.

Intención

- Tiene la intención de que un único proceso de construcción debe ser capaz de construir distintos objetos complejos, abstrayéndonos de los detalles particulares de cada uno de los tipos.

Otros nombres



Estructura y Participantes

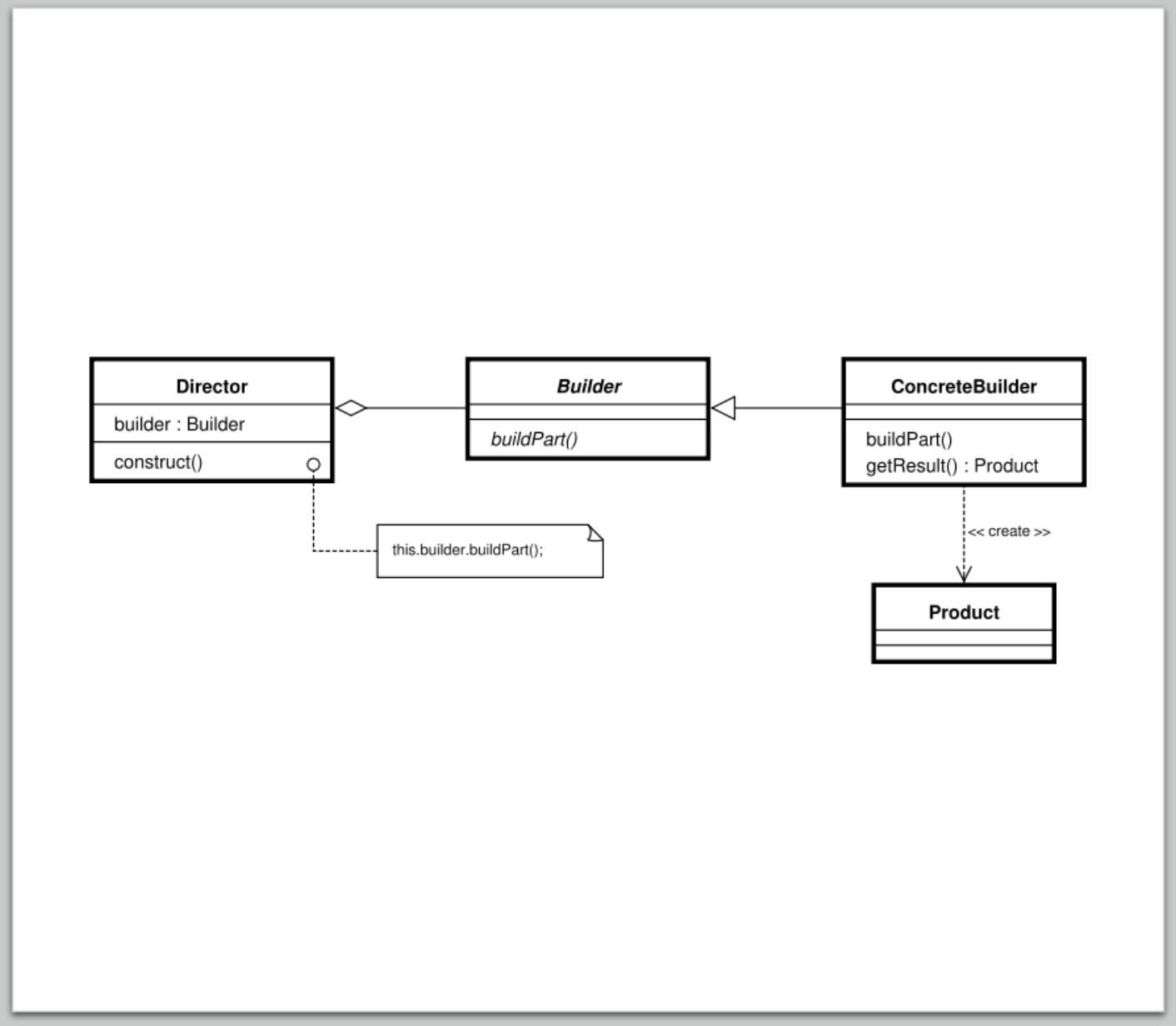
- Donde:

- **Director**: Se encarga de construir un objeto utilizando el Constructor.

- **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.



Aplicación y uso

- 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.

Ventajas y desventajas

- El código resulta más fácil de mantener cuando los objetos tienen mucha cantidad de atributos.
- Disminuye los errores al crear el objeto porque el builder especifica paso a paso cómo crearlos y qué atributos necesita.
- La mayor desventaja es la necesidad de mantener la duplicidad de de atributos que deben estar en la clase destino y en el builder.
- Más líneas de código