- You want to provide a class library of products, and reveal only their interfaces, not their implementations.

Builder Pattern

The Builder pattern separates the construction of a complex object from its representation so the same construction process can create different objects. The Builder pattern allows a client object to construct a complex object by specifying only its type and content. The client is shielded from the details of the object's construction. This simplifies the creation of complex objects by defining a class that builds instances of another class. The Builder pattern produces one main product and there might be more than one class in the product, but there is always one main class. Figure 4–2 illustrates the Builder pattern. When you use the Builder pattern, you create the complex objects one step at a time. Other patterns build the object in a single step.

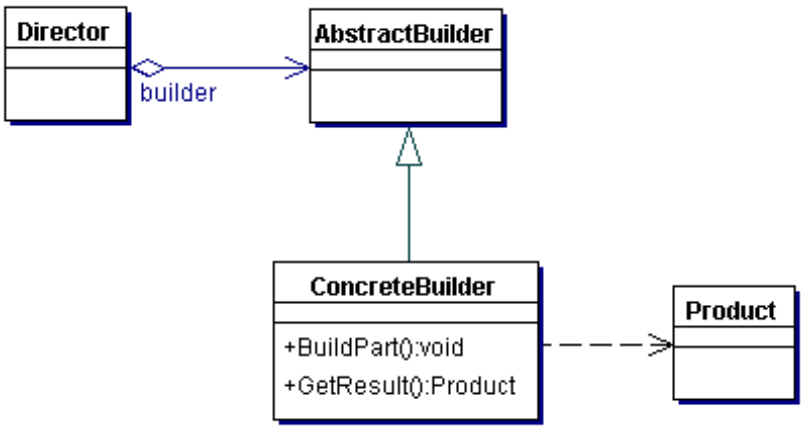

Figure 4–2 *The Builder Pattern*

BENEFITS

The following lists the benefits of using the Builder pattern:

- Lets you vary a product's internal representation.
- Isolates code for construction and representation.
- Gives you greater control over the construction process.