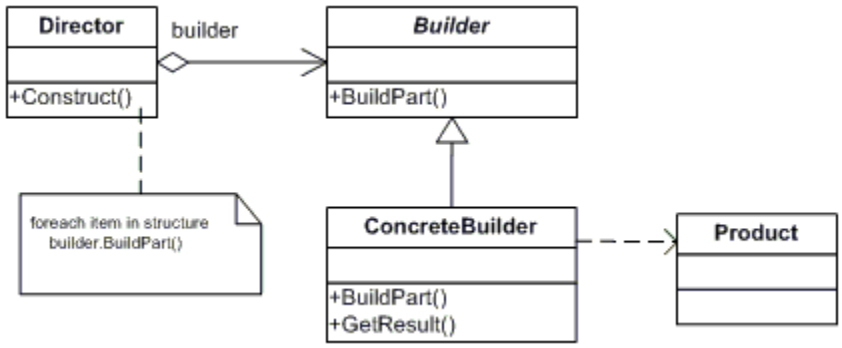
Instrument Builder - A .NET *Builder* Design Pattern Example

Pattern Summary: Separate the construction of a complex object from its representation so that the same construction process can create different representations. My example creates a representation that builds string instruments

UML



Components

Builder (*Instrument Builder)*: I would name this class 'abstract builder' because it only creates *methods* to produce parts of the Product object. These methods build 4 parts of the instrument; the neck, body, pickups, and 'hardware' (including tuners, bridge, control dials, etc). This class exists to be inherited.

Concrete Builder (GuitarBuilder, BassBuilder, ViolinBuilder): implements Builder interface to create an instrument object instance. Builds the instrument, or Product, object to its own specification.

Director (Manufacturer): creates a construct method to build an instrument object, using the builder interface. Calls each individual abstract build method from the Builder

Product (Instrument): Defines the assembly for an instrument object. Defines a Dictionary for the Instrument object's parts