Chain of Responsibility Pattern [Gamma et al]

Informally it wouldn't be wrong to define CoR as a usage of strategy classes to handle variant type of requests when complexity gears up and single property (handler) isn't enough for every different type of request. Chain of Responsibility pattern is a way of interposing a chain of objects between caller and target object. This arrangement is set of classes that lets each object pass through the chain all the way long from caller to target until it gets entertained. CoR offers flexibility in complex event processing scenarios, divide the responsibility among number of elements that shape up a circular chain [54]. Each handler is a successor to another handler in the chain until the current one finds its appropriate request by terminating the triggered event. Most common example to demonstrate CoR is ***Coin Sorter of Bank ATM machine*** which puts coin to correct location marked by value (cents, dollars etc.). Following are the variants of Chain of Responsibility Pattern:

Handling strategy [54]

This strategy deals with exact logic of handler behavior. They include Default Handler (A case handler that acts as a default in the chain which is useful when there is no explicit forwarding class exists), Handle and Extend (This variant involves adding to base behavior as the event is propagated along the chain) and Dynamic Handler (By changing the message passing behavior at runtime using setter method for each element of the chain basically allows to define and modify the chain).

Forwarding Strategy [54]

This strategy deals with handling and forwarding request types. This includes Handle by Default (handle any message that is not forwarded before), Propagate by Default (forward any message that is not handled explicitly), Forward by Default Handler (any message not handled at component and custom handler level can be forwarded to default handler) and Ignore by Default (messages that are not used in the application are discarded before being processed to reduce chatter).

Bureaucracy pattern [55]

Bureaucracy pattern is a composite pattern which can be used to create self-contained hierarchical structure capable enough to maintain its inner consistency by themselves and need no external control. The bureaucracy pattern share its role with three other behavioral patterns i.e. Mediator and Observer along with Chain of Responsibility pattern therefore participate in different overlapping pattern instantiations. Following is the role Diagram of the Bureaucracy pattern:

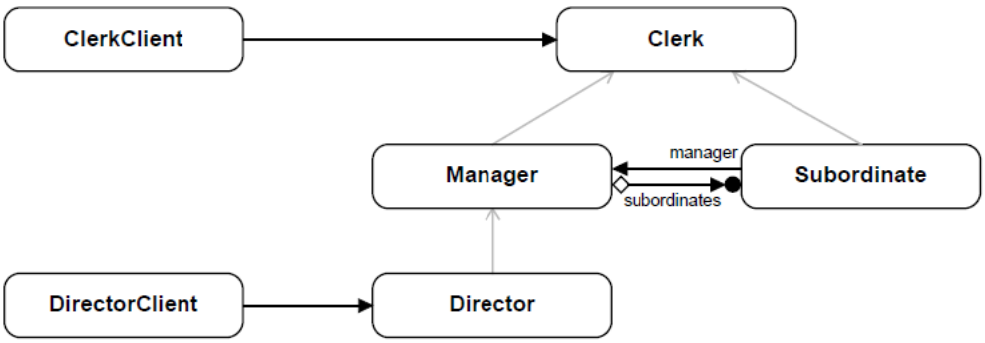

Figure 1.1 Role Diagram of Bureaucracy Pattern