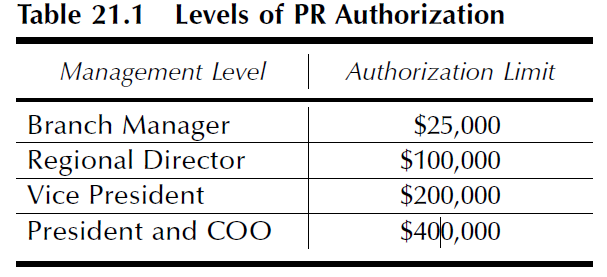
El patrón de cadena de responsabilidad es un patrón donde se propone un bajo acoplamiento entre objetos que reciben y manejan una petición. El objeto que recibe inicialmente la petición decide si tiene la capacidad de tratar la petición o pasarla al siguiente objeto en la cadena.

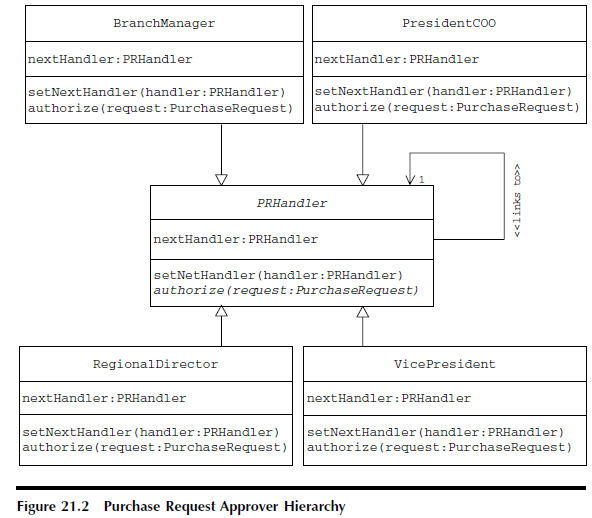
* El conjunto de objetos manejadores y el orden en la cadena de responsabilidad puede establecerse por el cliente en tiempo de ejecución dependiendo del estado de la aplicación.
* El cliente puede tener varios conjuntos de objetos manejadores para diferentes tipos de solicitud dependiendo de su estado actual.
* Cada objeto manejador puede pasar la solicitud a otro dependiendo del tipo de solicitud por lo cual debe proveer una interfaz consistente. (Puede que el objeto manejador sea una realización de una interfaz o una especialización de una clase abstracta).
* El objeto cliente que pasara la solicitud al objeto manejador no requiere conocer las capacidades del objeto que recibe la solicitud y por consiguiente tampoco los objetos en la cadena de responsabilidad requieren conocer cual objeto puede manejar la petición.
* Puede que la petición no sea manejada, es decir, la solicitud puede pasar por toda la cadena de responsabilidad sin ser satisfecha.

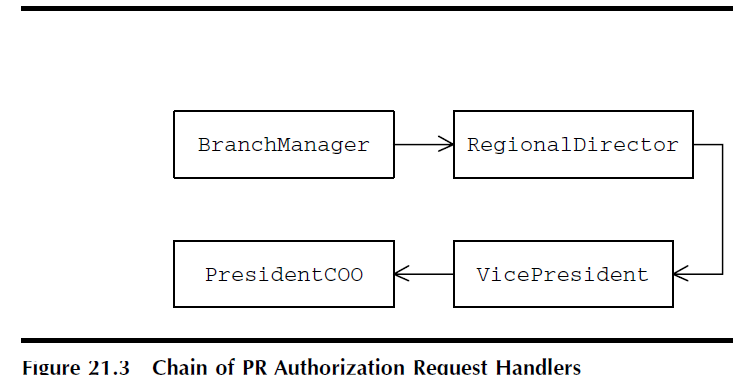
Ejemplo

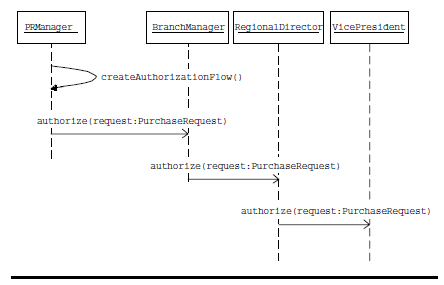


Para que la autorización de la compra se de el cliente debe realizar los siguientes pasos:

1. Crear los potenciales objetos responsables de la petición y los ordena por límite de autorización. El cliente debe conectar cada objeto responsable con el siguiente a través del método setNextHandler. Esto resulta en la cadena de responsabilidad de la petición de compra.
2. Enviar la petición al primer PRHandler en la cadena al invocar el método authorize de ese objeto pasando la orden de compra como argumento. Si monto es inferior al límite autorizado para ese responsable, la petición se aprueba de lo contrario se pasa al siguiente responsable. Si la petición llega hasta el final de la cadena sin ser autorizada se despliega un mensaje.







Se trata de dar tratamiento a una petición del cliente donde cada objeto de la jerarquía PRHandler podría darle tratamiento a esta petición dependiendo de las características de la misma. Para esto se define flujo de autorización para que el primer objeto valide si tiene la capacidad de tratarla, de lo contrario le envia la petición al siguiente objeto del flujo de autorización.

De esta manera:

1. Cliente crea a los objetos de tratamiento de la petición y los ordena en un el flujo de autorización “createAuthorizationFlow()” definido con “setNextHandler(PRHandler)”.
2. La cliente crea la petición de orden de compra y se la envía al primer objeto de la cadena para que sea tratada (authorize(PurchaseRequest)).
3. Este objeto valida si tiene la capacidad de dar tratamiento y si no pasa la petición al siguiente objeto en la cadena (getNextHanlder()) y envia el mensaje authorize. Asi hasta que se consiga dar tratamiento a la petición o hasta que se llegue al ultimo objeto de la cadena.
4. Ni el cliente ni los objetos de la cadena saben si el siguiente objeto puede dar tratamiento cada objeto debe validar esto una vez recibe la petición.
5. Puede que ningún objeto pueda tratar la petición.