Chain of Responsability

Mario Vila Durán

Enginyeria del software

Definició

- El patró s'utilitza principalment quan hi ha més d'un objecte que pot realitzar una petició d'un client

Funcionament

- Cadena d'objectes.
- La petició recorre la cadena.
- Si l'objecte pot, realitza la petició.
- En cas contrari passa petició al successor.

Classes principals

- Handler:
  - Interfície per manipular peticions.
  - Pot implementar l'enllaç successor.
- Concret Handler:
  - Manipulen les peticions.
  - Tenen accés al seu successor.
- Client:
  - Crear petició
  - Pot defineix cadena d'objectes

Beneficis

- Proporciona menor acoblament entre remitent i receptor d'una petició.
- Conjunt de peticions manipulades i l'ordre de la cadena pot ser decidit dinàmicament per el client en temps d'execució.
- Simplifica objectes, no han de conèixer l'estructura de la cadena, només una referència al seu successor.

Utilització

- Principalment s'utilitza en sistemes de finestres.
  - Clicks del ratolí
  - Events del teclat.

Punt fosc

- No es garanteix l'execució de la petició. Pot fallar al final de la cadena si cap objecte ha pogut realitzar la petició.

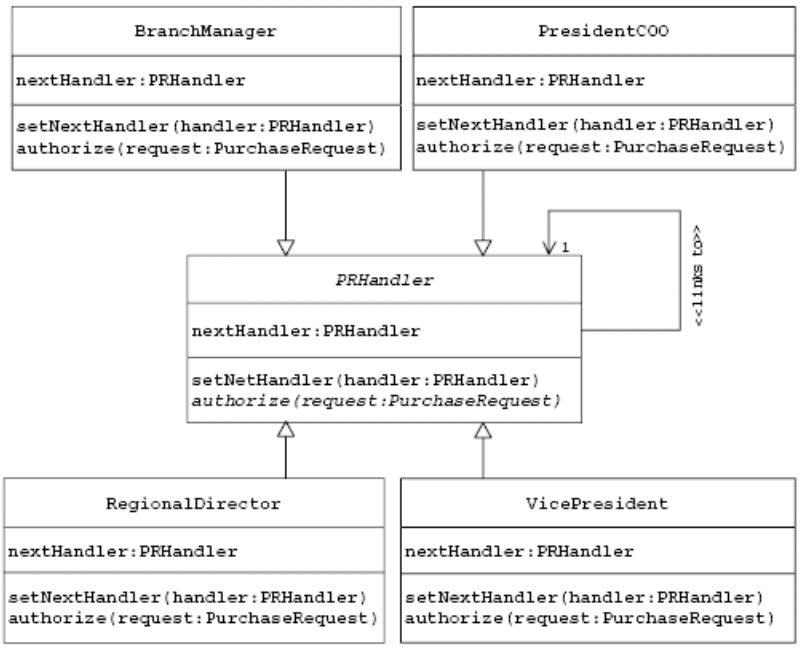
Exemple I

- Aplicació que simula el procés d'autoritzacions de compra.
- Petició ha de ser autoritzada per el responsable correcte en cada cas.
- 4 nivells de responsabilitat

Management Level	*Authorization Limit*
Branch Manager	$25,000
Regional Director	$100,000
Vice President	$200,000
President and COO	$400,000

Exemple II

- Diagrama de classes



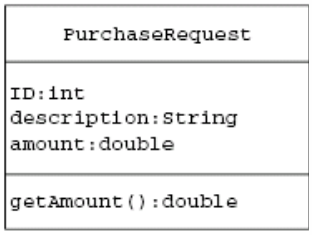
Exemple III

- Classes dels responsables

```
class BranchManager {
  static double LIMIT = 25000;
          …
          …
}//End of class
class RegionalDirector {
  static double LIMIT = 100000;
          …
          …
}//End of class
class VicePresident {
  static double LIMIT = 200000;
          …
          …
}//End of class
class PresidentCOO {
  static double LIMIT = 400000;
          …
          …
}//End of class
```

Exemple IV

- Classe PurChaseRequest (Petició de compra)




```
class PurchaseRequest {
  private int ID;
  private String description;
  private double amount;
  public PurchaseRequest(int id, String desc, double amt) {
    ID = id;
    description = desc;
    amount = amt;
  }
```

Exemple V

- Interfície PRHandler

```
public abstract class PRHandler {
  private PRHandler nextHandler;
  private String handlerName;
  public PRHandler(String name) {
    handlerName = name;
  }
  public String getName() {
    return handlerName;
  }
  public abstract boolean authorize(PurchaseRequest request);
  public PRHandler getNextHandler() {
    return nextHandler;
  }
  public void setNextHandler(PRHandler handler) {
    nextHandler = handler;
  };
}
```

Exemple VI

- Codi dels responsables

```
class BranchManager extends PRHandler {
  static double LIMIT = 25000;
  public BranchManager(String name) {
    super(name);
  }
  public boolean authorize(PurchaseRequest request) {
    double amount = request.getAmount();
    if (amount <= LIMIT) {
      System.out.println(" Branch Manager " + getName() +
                         " has authorized the PR - " + request);
      return true;
    } else {
      //forward the request to the next handler
      return getNextHandler().authorize(request);
    }
  }
}//End of class
```

```
class PresidentCOO extends PRHandler {
  static double LIMIT = 400000;
  public PresidentCOO(String name) {
    super(name);
  }
  public boolean authorize(PurchaseRequest request) {
    double amount = request.getAmount();
    if (amount <= LIMIT) {
      System.out.println(" President & COO " + getName() +
                         " has authorized the PR - " + request);
      return true;
    } else {
      System.out.println("PR - " + request +
                         " couldn't be authorized.\n " +
                         "Executive Board needs to be " +
                         "consulted for approval \n" +
                         "reason: Amount too large");
      return false;
    }
  }
}//End of class
```

Exemple VII

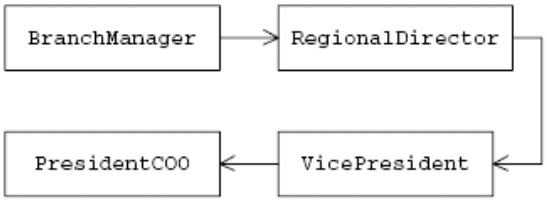
- El client:

```
public class PRManager {
  private BranchManager branchManager;
  private RegionalDirector regionalDirector;
  private VicePresident vicePresident;
  private PresidentCOO coo;
  public static void main(String[] args) {
    PRManager manager = new PRManager();
    manager.createAuthorizationFlow();
    PurchaseRequest request =
      new PurchaseRequest(1, "Office Supplies",10000);
    manager.branchManager.authorize(request);
    request = new PurchaseRequest(2, "HardWare Procurement",
              175000);
    manager.branchManager.authorize(request);
    request = new PurchaseRequest(3, "AD Campaign",800000);
    manager.branchManager.authorize(request);
  }
```

Exemple VIII

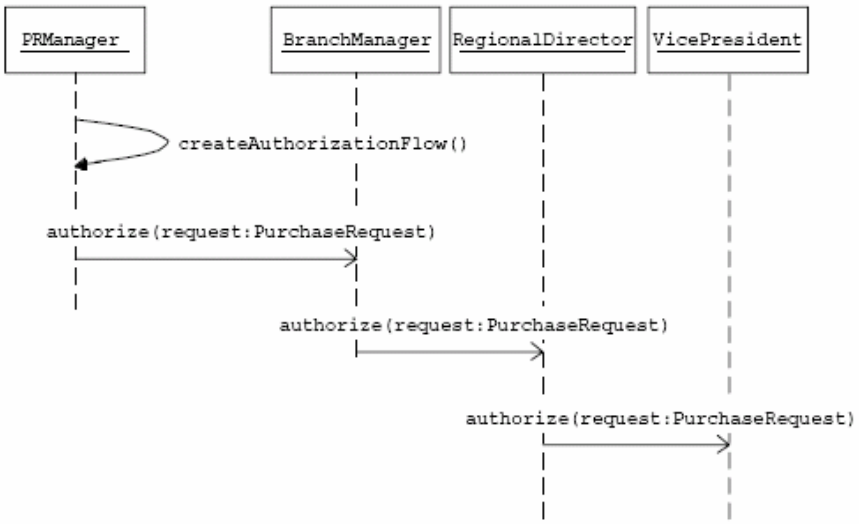
- Creació de la cadena

```
public void createAuthorizationFlow() {
  branchManager = new BranchManager("Robin");
  regionalDirector = new RegionalDirector("Oscar");
  vicePresident = new VicePresident("Kate");
  coo = new PresidentCOO("Drew");
  branchManager.setNextHandler(regionalDirector);
  regionalDirector.setNextHandler(vicePresident);
  vicePresident.setNextHandler(coo);
}
```



Exemple IX

- Diagrama d'activitat d'una petició de $150.000



Chain of Responsability

Mario Vila Durán

Enginyeria del software