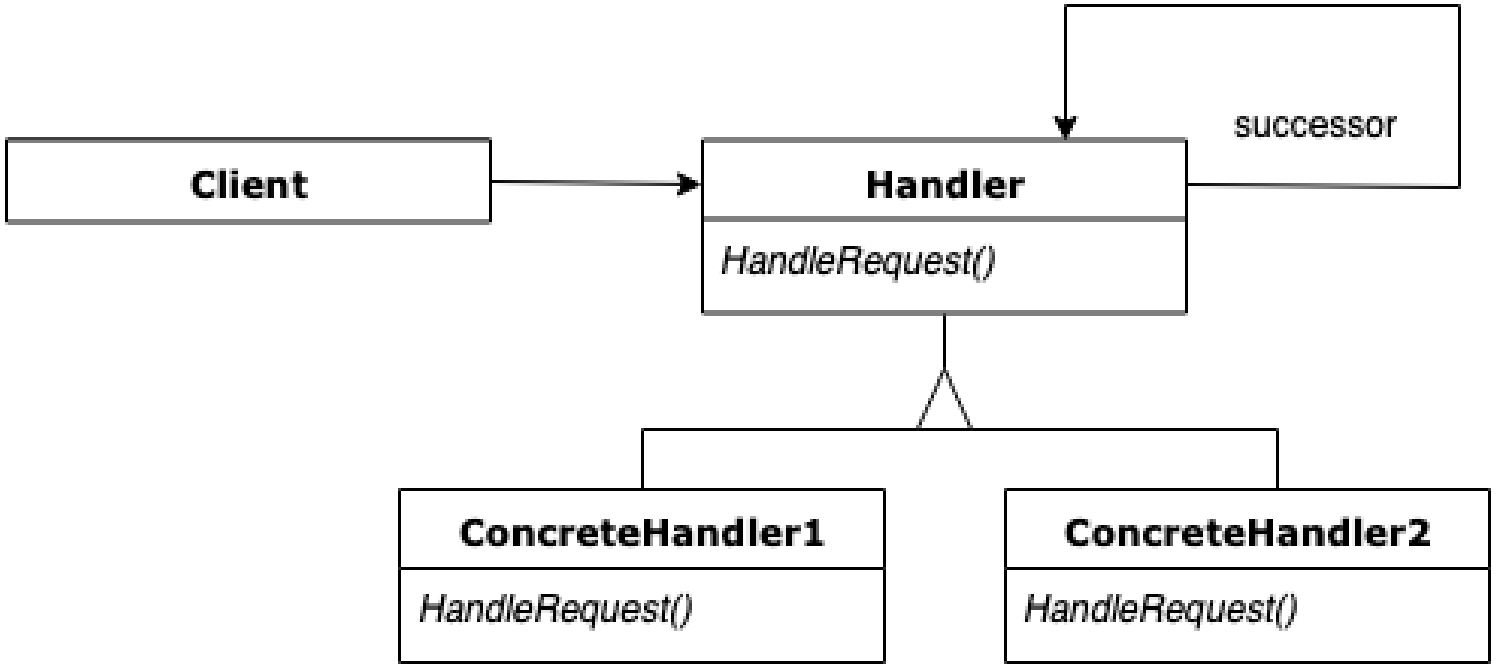
Chain of responsibility

Christophe Damas

José Vander Meulen

Objectif

- Chaînage d'un ensemble d'objets de traitement.
  - On soumet un objet de commande à la chaîne.
  - Chaque objet traitement soit traite la commande, soit passe la commande à l'objet traitement suivant.
- Exemple: Inscription IPL

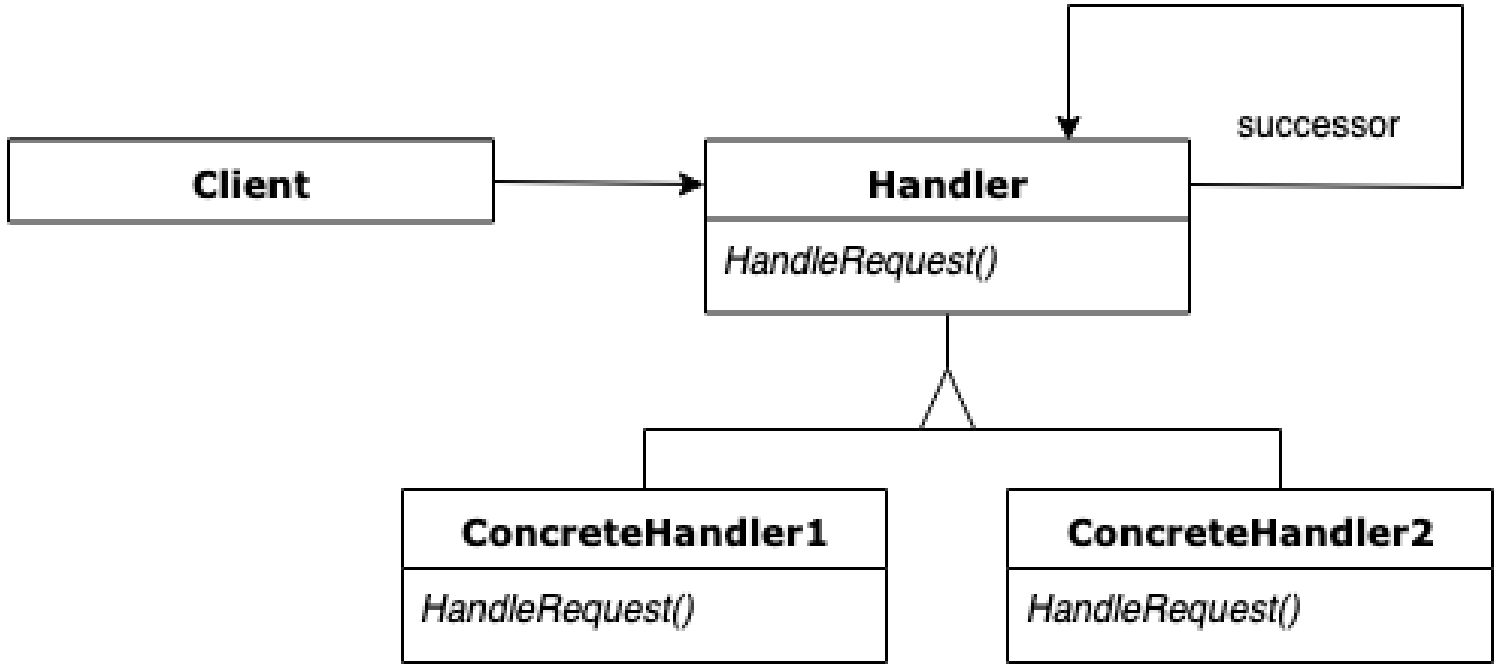


Exemple très très simple

- Programme qui prend un entier en paramètre et affiche
  - « positif » si l'entier > 0
  - « négatif » si l'entier < 0
  - « nul » si l'entier = 0

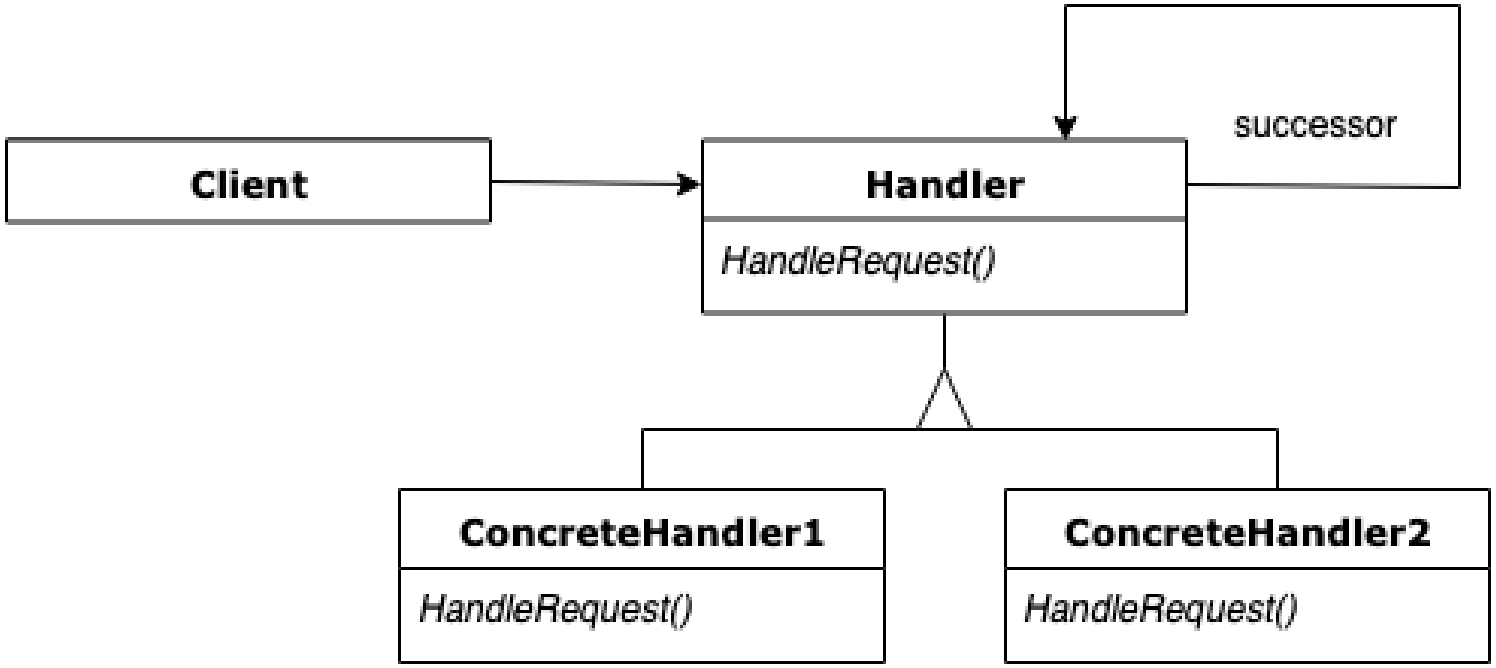
```
public static void main(String[] args) {
      Scanner s = new Scanner(System.in);
      int i = Integer.parseInt(s.nextLine());
      if (i > 0) System.out.println("positif");
      else if (i == 0) System.out.println("nul");
      else if (i < 0) System.out.println("négatif");
      s.close();}
```

En utilisant COR



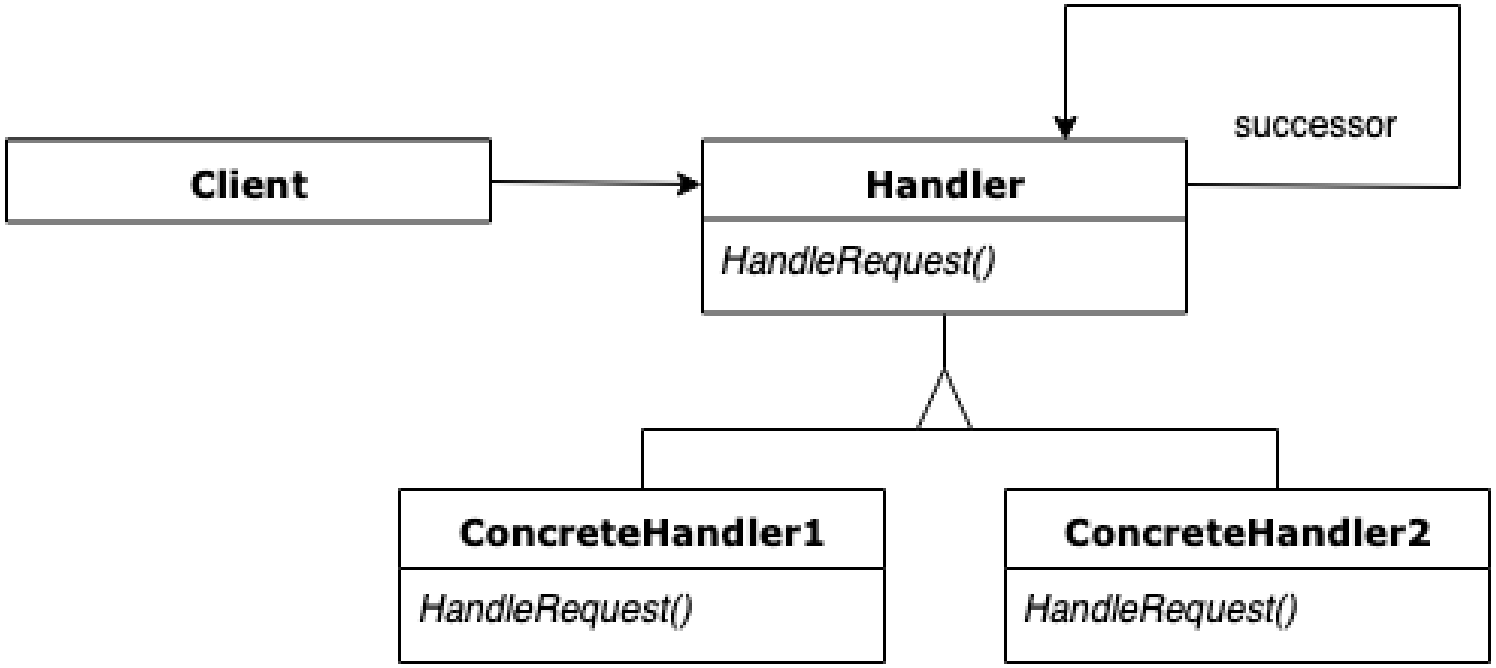

```
public abstract class Handler {
	private Handler successeur;
	public Handler(Handler successeur) {this.successeur=successeur;}
	public void traiterNombre(int x) {
		if (successeur == null) throw new RuntimeException();
		successeur.traiterNombre(x);}
}
```

En utilisant COR




```
public class PositiveHandler extends Handler {
	public PositiveHandler(Handler successeur) {super(successeur);}
	@Override
	public void traiterNombre(int x) {
		if (x>0) System.out.println("positive");
		else super.traiterNombre(x);}
}
```

En utilisant COR




```
public class Client {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        int i = Integer.parseInt(s.nextLine());
        Handler h = new PositiveHandler(new NegativeHandler(new ZeroHandler(null)));
        h.traiterNombre(i);
        s.close();}
}
```

Avantages

- Les différentes conditions ne sont plus statiques et peuvent être modifiées à l'exécution
- Pattern très utile quand plusieurs objets peuvent gérer une requête
  - Permet de découpler la classe qui envoie la requête et celle qui la gère