Design patterns

1. Factory:
   Party class implements OperationFactory interface and has method createOperation()
   package cz.cvut.fel.omo.api.impl
   EcoSystem class implements PartyFactory interface and has method createParty()
   package cz.cvut.fel.omo.api

2. Singleton:
   EcoSystem is a Singleton, has method getInstance()
   package cz.cvut.fel.omo.api

3. Composite:
   Product class implements Reportable interface and keeps List<Reportable> myComponents as a field, so user can always look at report of product(all operation provided on it) and report of any component
   package cz.cvut.fel.omo.api.product

4. Decorator:

5. Visitor:
   OperationVisitor interface and its implementation class ReportVisitor. To visit different Operation classes and provide correct reports.
   package cz.cvut.fel.omo.api
   package cz.cvut.fel.omo.api.impl

6. Observer:
   Observer and Subject interfaces which PartyImpl and ChannelImpl classes implement. It Helps Channel to notify all its participants when new Request created
   package cz.cvut.fel.omo.api
   package cz.cvut.fel.omo.api.impl

7. State:
   State abstract class and its extended classes: Preparation, Waiting, Producing and End. These states has ProductionProcess class. Convenient way to implement production process.
   package cz.cvut.fel.omo.production