FCS Design:  
Messages:  
addKitsToQueue(ArrayList<PartType> parts,int numOfKits){  
 orders.add(new Order(parts,numOfKits));  
}  
stopMakingKit(ArrayList<PartType> parts){  
 for(Order o: orders){  
 if(o.parts.equals(parts)) o.state=cancel;  
 }  
}  
startProduction(){  
 state=OrderState.started;  
}  
  
OrderFinished(){  
 state=myState.started;  
}  
  
Scheduler:  
if(state==MyState.started){  
 if(!orders.isEmpty()){  
 for(Order o:Orders){  
 if(o.state==OrderState.pending){  
 placeOrder(o);  
 }  
 }  
}  
  
Actions:  
placeOrder(Order o){  
 o.state=OrderState.ordered;  
 state=myState.loaded;  
 stand.MakeThisManyKits(o.numberOfKits, o.parts);  
 partsRobot.HereIsKitConfiguration(o.parts);  
 for(int i=0;i<nests.size();i++){  
 nests[i].HereIsPartType(o.parts[i]);  
 }  
 List<Bin> bins = new List<Bin>();  
 for(int i=0;i<o.parts.size();i++)  
 {  
 bins.add(new Bin(o.parts[i]));  
 }  
 gantry.HereIsBinConfig(bins);  
}  
  
  
Data:  
private StandAgent stand;  
private PartsRobot partsRobot;  
private GantryAgent gantry;  
private List<NestAgent> nests;  
private myState state;  
private List<Order> orders;  
  
public enum myState {pending, started, loaded};  
public enum OrderState {pending, ordered }  
  
private class Order{  
public OrderState state;  
public ArrayList<PartType> parts;  
public int numberOfKits;  
public Order(ArrayLIst<PartType> parts, int numKits){  
this.state=OrderState.pending;  
this.parts=parts;  
numberOfKits=numKits;}  
}