```
package com.state.money;

//各种具体状态类
class FeedBackState extends AbstractState {

  @Override
  public String getCurrentState() {
    return StateEnum.FEED_BACKED.getValue();
  }
}

class GenerateState extends AbstractState {

  @Override
  public void checkEvent(Context context) {
    context.setState(new ReviewState());
  }

  @Override
  public void checkFailEvent(Context context) {
    context.setState(new FeedBackState());
  }

  @Override
  public String getCurrentState() {
    return StateEnum.GENERATE.getValue();
  }
}

class NotPayState extends AbstractState {

  @Override
  public void payOrderEvent(Context context) {
    context.setState(new PaidState());
  }

  @Override
  public void feedBackEvent(Context context) {
    context.setState(new FeedBackState());
  }

  @Override
  public String getCurrentState() {
    return StateEnum.NOT_PAY.getValue();
  }
}
```

```
class PaidState extends AbstractState {

  @Override
  public void feedBackEvent(Context context) {
    context.setState(new FeedBackState());
  }

  @Override
  public String getCurrentState() {
    return StateEnum.PAID.getValue();
  }
}

class PublishState extends AbstractState {

  @Override
  public void acceptOrderEvent(Context context) {
    //把当前状态设置为  NotPayState。。。
    //至于应该变成哪个状态，有流程图来决定
    context.setState(new NotPayState());
  }

  @Override
  public void notPeopleAcceptEvent(Context context) {
    context.setState(new FeedBackState());
  }

  @Override
  public String getCurrentState() {
    return StateEnum.PUBLISHED.getValue();
  }
}

class ReviewState extends AbstractState {

  @Override
  public void makePriceEvent(Context context) {
    context.setState(new PublishState());
  }

  @Override
  public String getCurrentState() {
    return StateEnum.REVIEWED.getValue();
  }

}
```