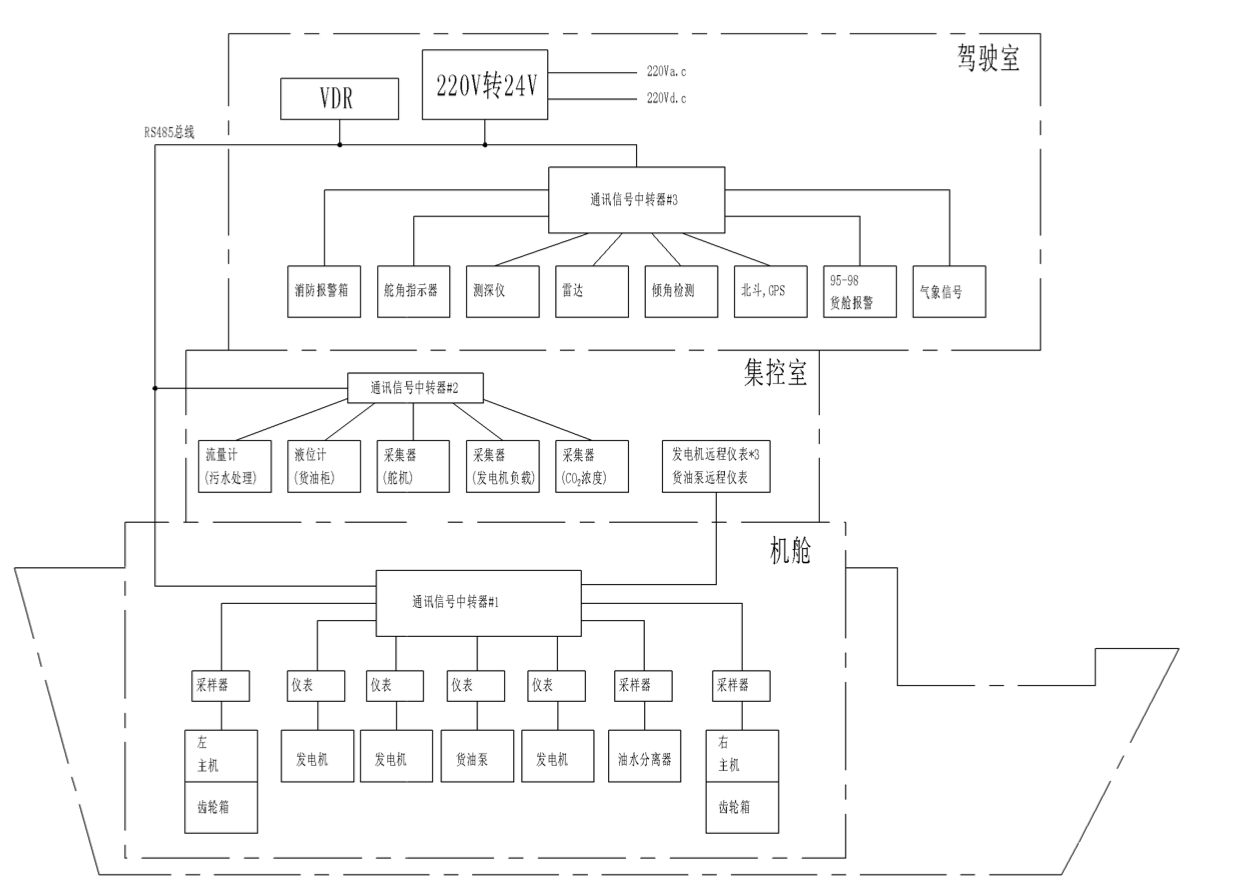
这里举一个事件驱动的例子，是我本科阶段做过的一个项目。

文档见<https://shimo.im/docs/rL8dM3Oy174Nr1bt/read>

1. **项目说明**

“智慧长江”项目主要负责通过各类传感器收集长江船只的信息，并上传到服务器，通过人工智能算法对信息进行分析，指导船员进行安全的操作。我们主要负责的是信息收集和上传。

拓扑结构如下：



如同所示，系统主要由以下构件组成：

1. 各类传感器，对应事件驱动风格系统中的事件接收器；
2. 通讯信号中转器，对应系统中的管理类的子系统，负责收集传感器信息与接受最上级指令。
3. 树莓派（图中没画出），是系统中的最高级管理子系统，负责发送指令与处理异常（异常全部抛到这一系统中处理），上传信息给服务器。

工作流程大致为：

1. 树莓派发送查询指令->2
2. 中转器接收指令->3
3. 中转器处理指令，将请求发送到指令对应传感器->4
4. 传感器接收指令->5
5. 传感器上传信息（包括异常）->6
6. 中转器上传信息->7
7. 树莓派接收并处理信息，上传至服务器->1
8. 主要解决的问题：

**2.1.解决地址冲突**

智慧船舶是基于长江现有船舶进行改造，主体思路是采集船舶上各传感器的数据，这些传感器基本与船舶的数字仪表相连接，为了维持原有的状态，我们只能依照原传感器的设置进行采集信号，由于采用的是485通信协议，那么则会出现地址冲突问题，比如流量计和风向测量仪的从机地址都是0x01，若这两个传感器挂在同一总线上，则会发生地址冲突。

项目中给每个传感器分配了一个独立的板子收集信息，给板子重新编号，将每个传感器看成一个独立的子系统，解决了地址冲突问题。

**2.2.降低系统耦合**

因为传感器的通信协议各有不同，先将各传感器接转换板进行数据采集，转换为统一格式后利用CAN总线上传到通信信号中转器，即该系统一旦设计好，对于不同传感器只需要更改CAN转换板的程序即可。