西安电子科技大学
XIDIAN UNIVERSITY

计算机科学与技术学院
SCHOOL OF COMPUTER SCIENCE AND TECHNOLOGY
国家示范性软件学院
NATIONAL PILOT SCHOOL OF SOFTWARE ENGINEERING

软件体系结构

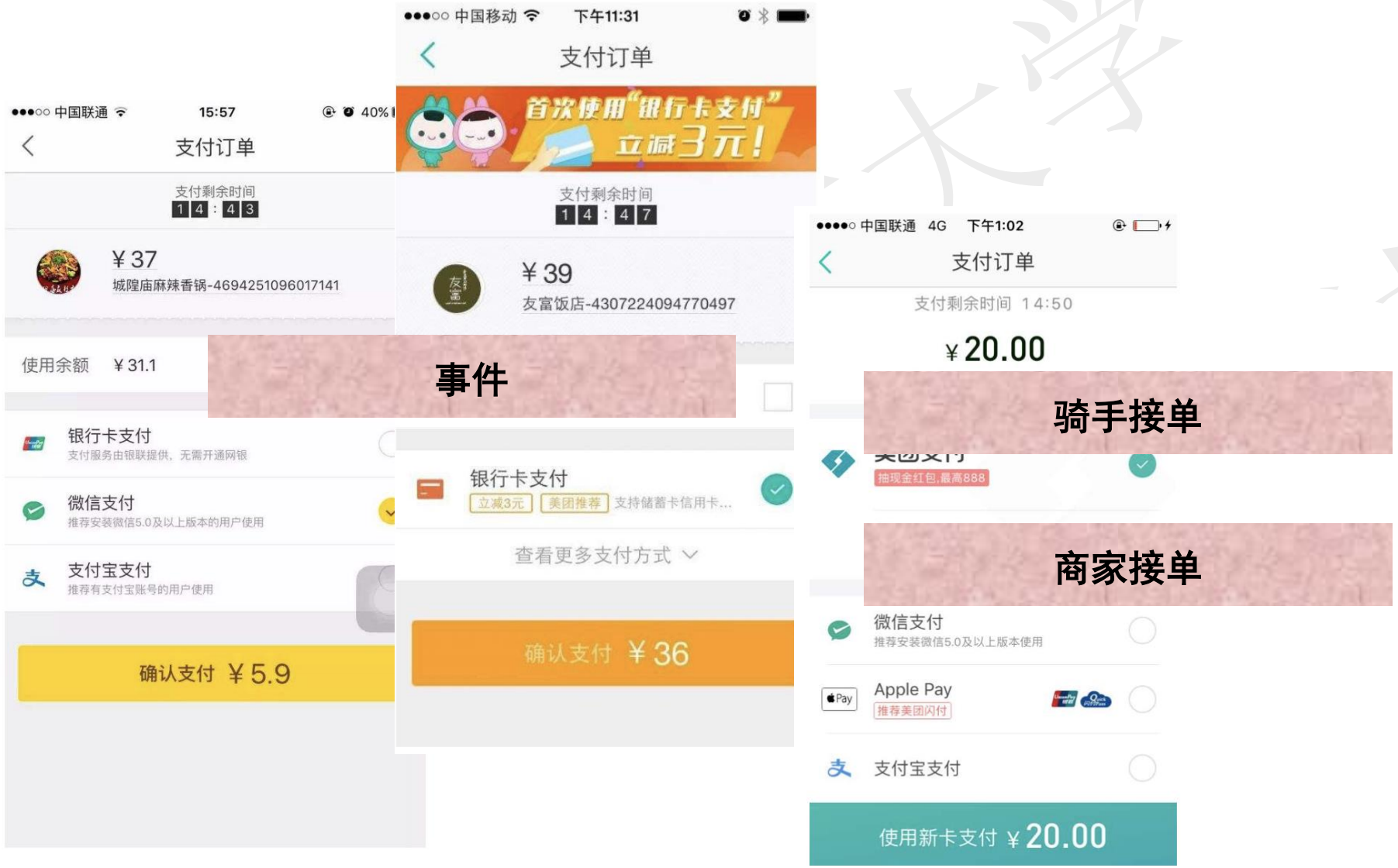
—事件系统

主讲人：蔺一帅 讲师

本节内容

1. 事件
2. 事件系统
3. 事件系统调度机制

事件



事件

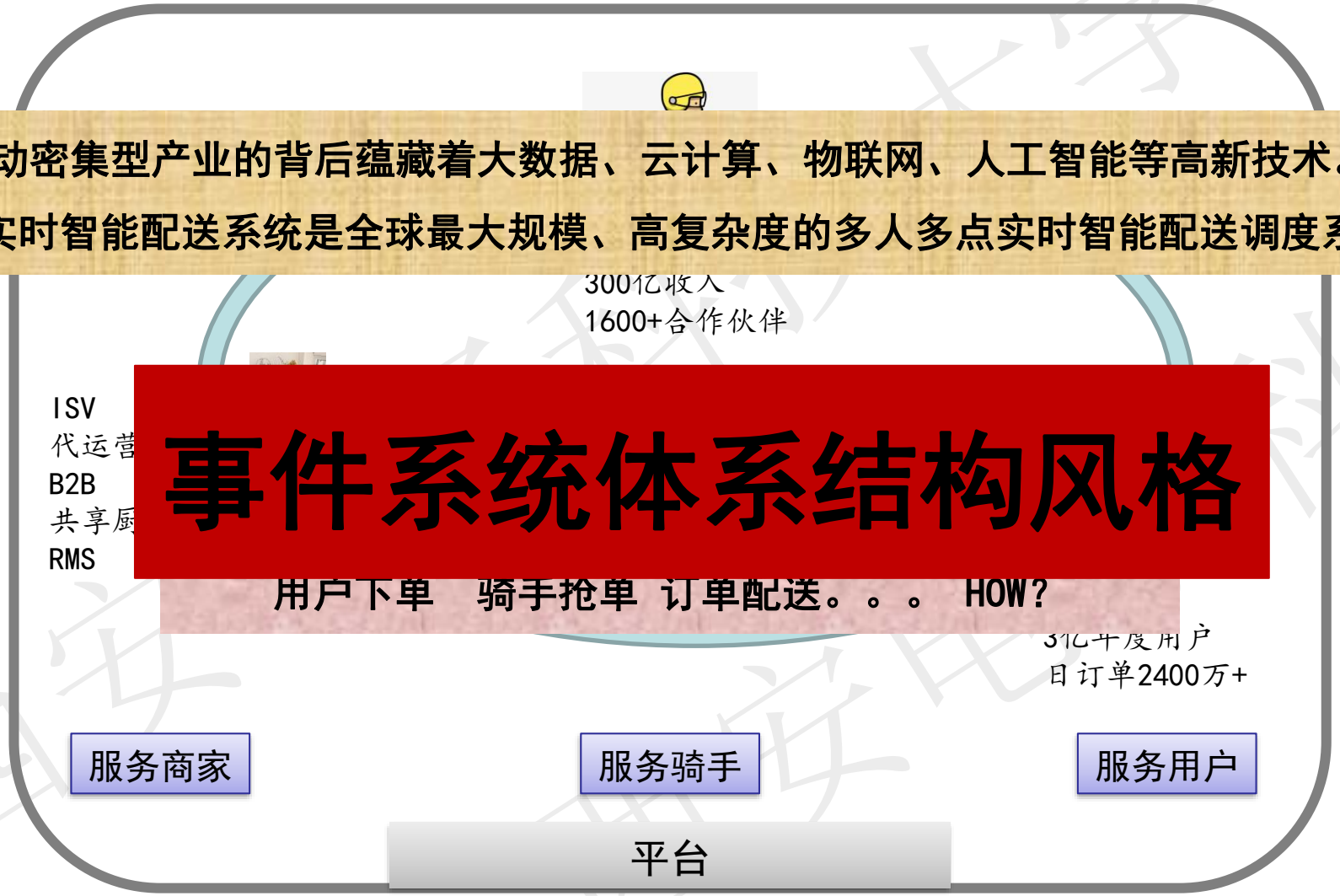
骑手接单

商家接单

事件

劳动密集型产业的背后蕴藏着大数据、云计算、物联网、人工智能等高新技术。

美团实时智能配送系统是全球最大规模、高复杂度的多人多点实时智能配送调度系统。



事件系统体系结构风格

事件

“外卖经济体”：通过移动互联网的结合与加持，外卖这一业态已经形成了包括商家、用户、骑手、生态伙伴在内的完整经济体，在推动行业变革以及促进就业等社会效益上发挥着越来越大的作用。

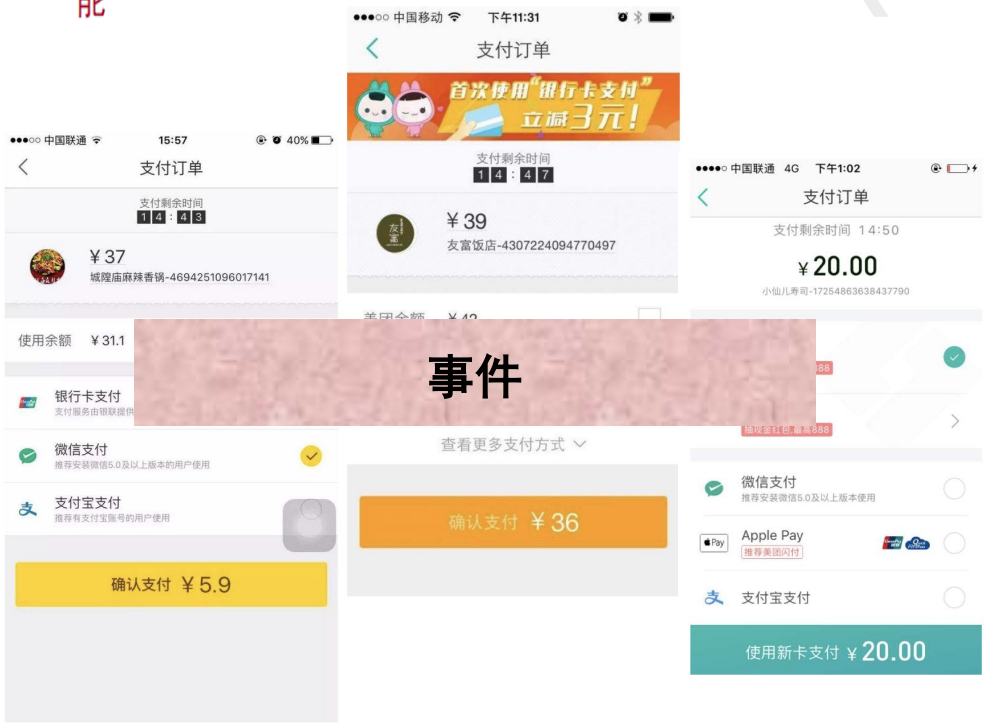
- **骑手：**在美团外卖带动就业的270万骑手中，77%来自农村。有67万骑手来自全国94%的贫困县。有超4成的骑手在工作之余，通过看书、报名培训等进行学习提升。2018年，骑手社交网站的三个关键词为“坚持”、“奋斗”、“成长”。
- **用户：**我国已有45.4%的网民使用过网络外卖服务，网络外卖用户规模高达3.6亿。国家信息中心报告显示，2018年，我国在线外卖收入约 4712 亿元。
- **平台：**商户、用户和骑手通过外卖平台获得了价值，推动平台快速发展。外卖平台作为连接用户、商户、骑手的枢纽，有责任去助力这个经济体中各个群体的发展。

事件

Event（事件）

能够激活对象功能的动作。当发生这种动作后将给所涉及对象发送一个消息，对象便可执行相应的功能

对某一对象发生什么事件，该对象便找到相应的处理该事件的程序去处理这一事件

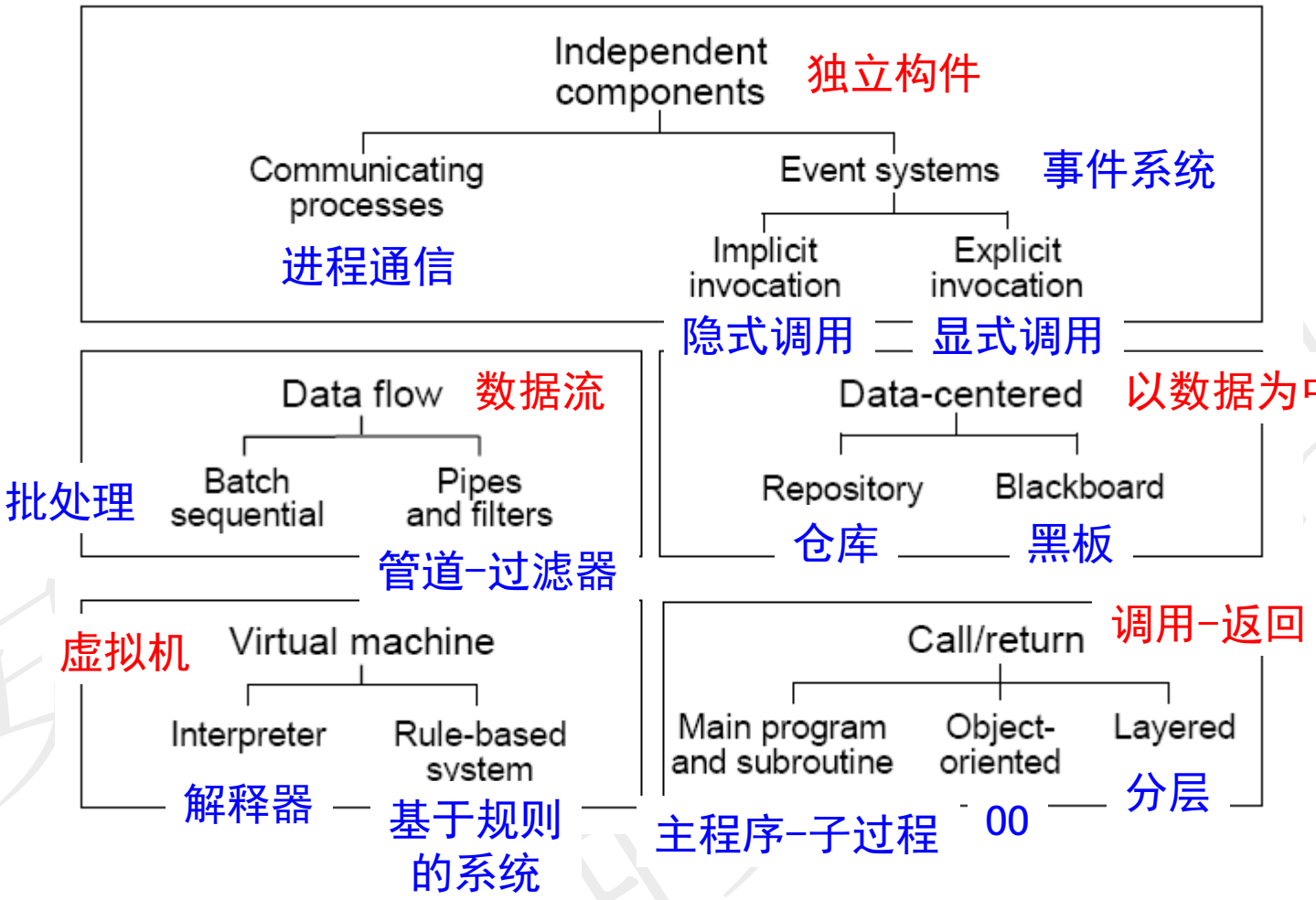


事件

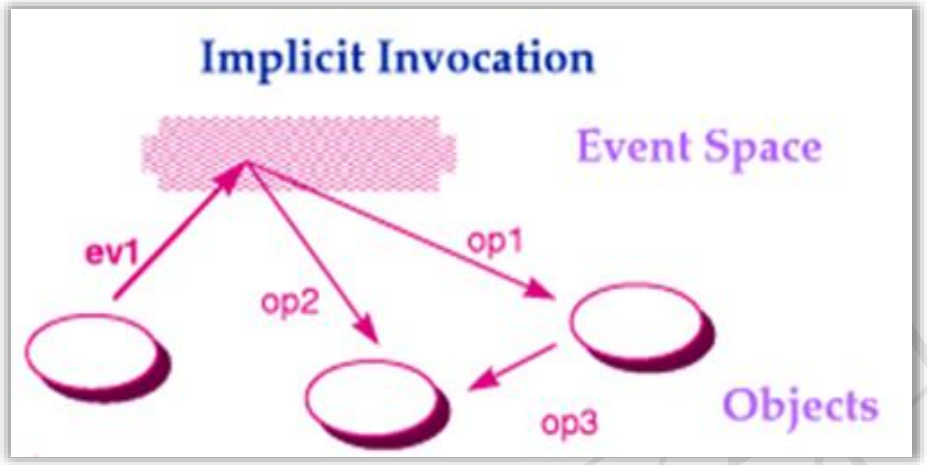
骑手接单

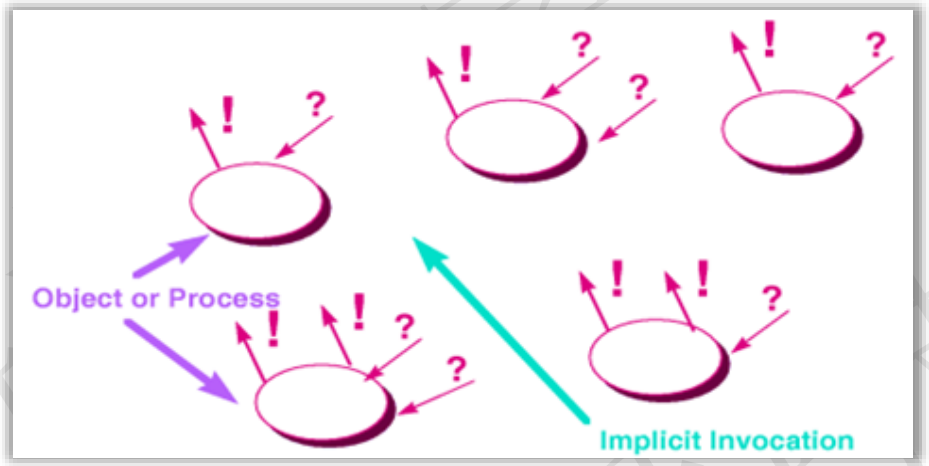
商家接单

事件系统-Taxonomy of styles



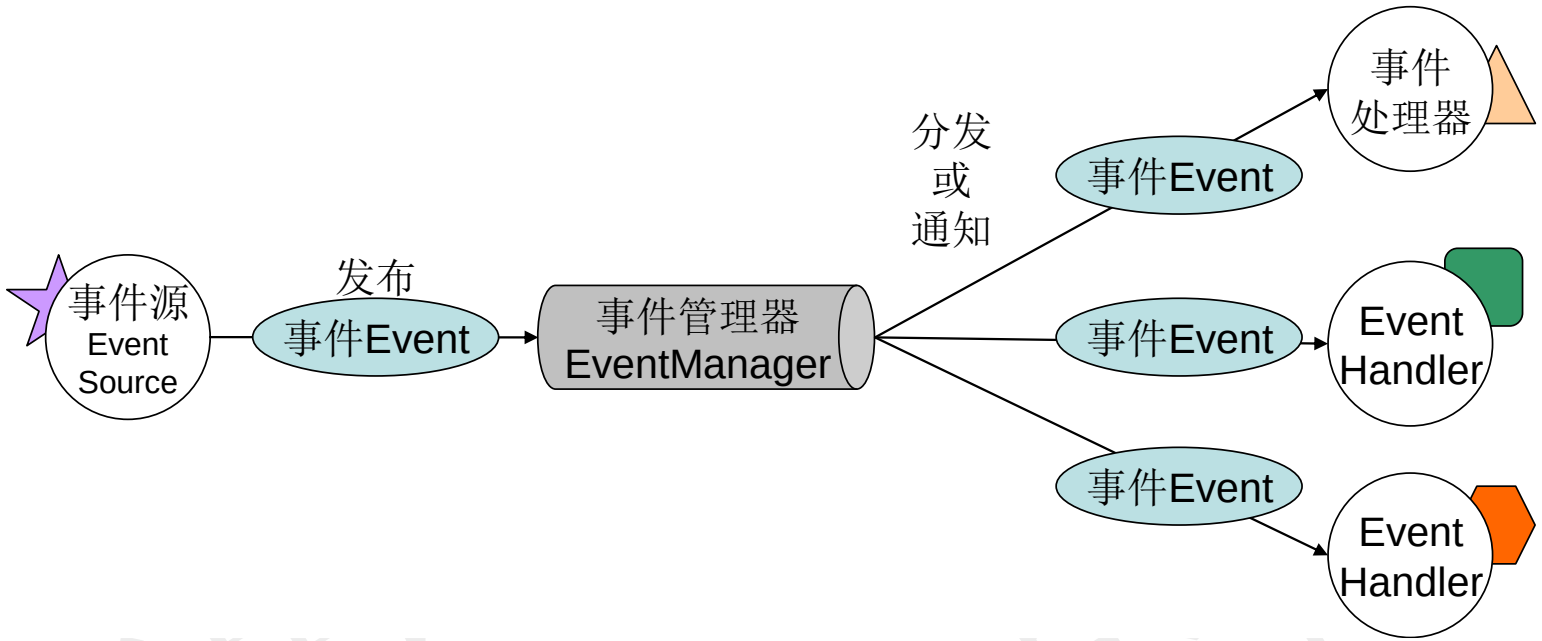
事件系统-隐式调用





- **主要特点**
  - ✓ 事件的触发者并不知道哪些构件会被这些事件影响，相互保持独立。
  - ✓ 不能假定构件的处理顺序，甚至不知道哪些过程会被调用。
  - ✓ 各个构件之间彼此之间无连接关系，各自独立存在，通过对事件的发布和注册实现关联。

事件系统-基本结构与工作原理(1/2)



特点	描述
分离的交互	事件发布者并不会意识到事件订阅者的存在。
一对多通信	采用发布/订阅消息传递，一个特定事件可以影响多个订阅者。
基于事件的触发器	由事件触发过程调用。
异步	支持异步操作。

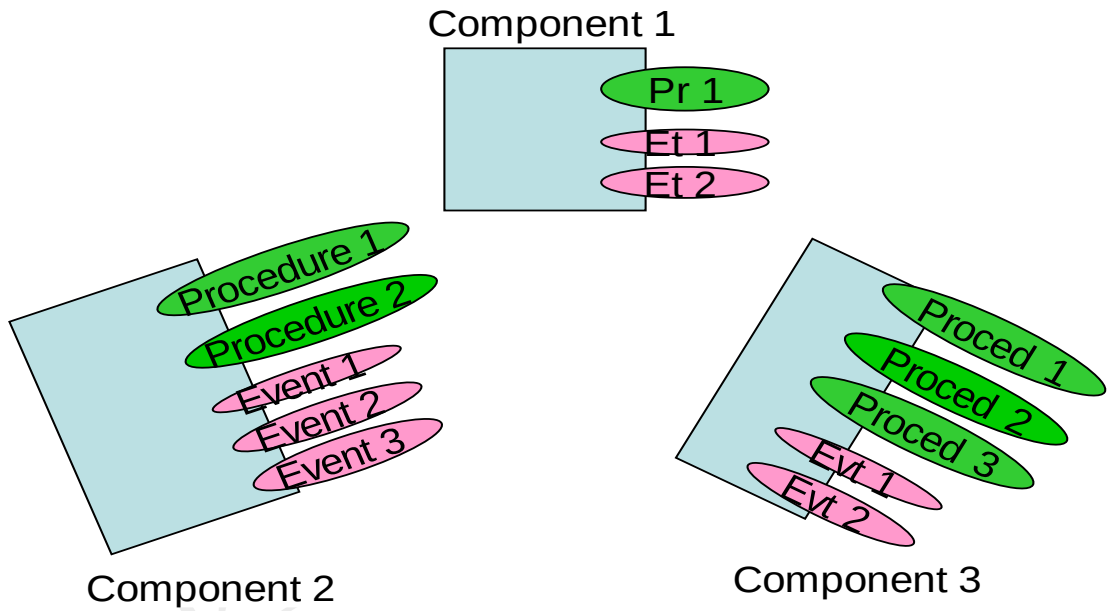
事件系统-基本结构与工作原理(2/2)

In an implicit-invocation-style systems, an event implicitly causes the invocation of procedures in other modules.

- A component(Event Source) can annonce (or broadcast) one or more events. 一个组件可以广播一些事件。事件源
- Other components(Event Handlers) in the system can register an interest in an event by associating a procedure with it. 系统中的其它构件可以注册自己感兴趣的事件，并将自己的某个过程与相应的事件进行关联。事件处理器
- When the event is announced the system (Event Manager) itself invokes all of the procedures that have been registered for the event. 当一个事件被发布，系统自动调用在该事件中注册的所有过程。事件管理器

事件系统-基本构件

- Components of event systems: objects or processes whose interfaces provide both:
  - ✓ a collection of procedures (过程或函数，充当事件源或事件处理器的角色)
  - ✓ a set of events (事件)

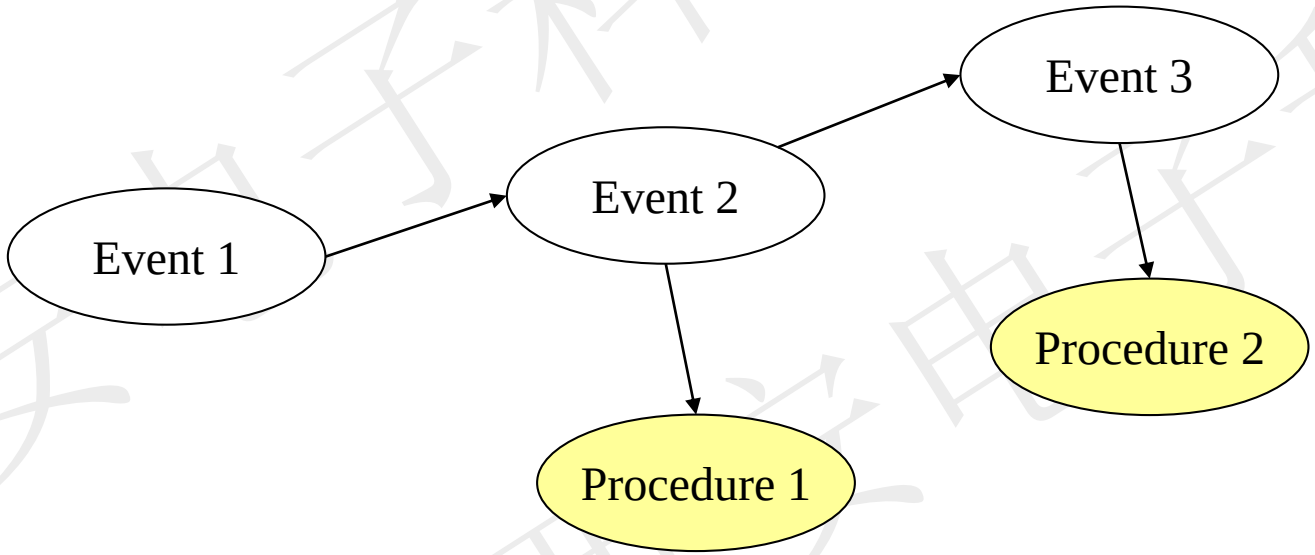


事件系统-连接机制(1/2)

- Connectors of event systems : event-procedure bindings
  - ✓ Procedures are registered with events (过程<事件处理器>向特定的事件进行注册)
  - ✓ Components communicate by announcing events at "appropriate" times (构件<事件源>发布事件)
  - ✓ when an event is announced the associated procedures are (implicitly) invoked (当某些事件被发布时，向其注册的过程被隐式调用)
  - ✓ Order of invocation is non-deterministic (调用的次序是不确定的)

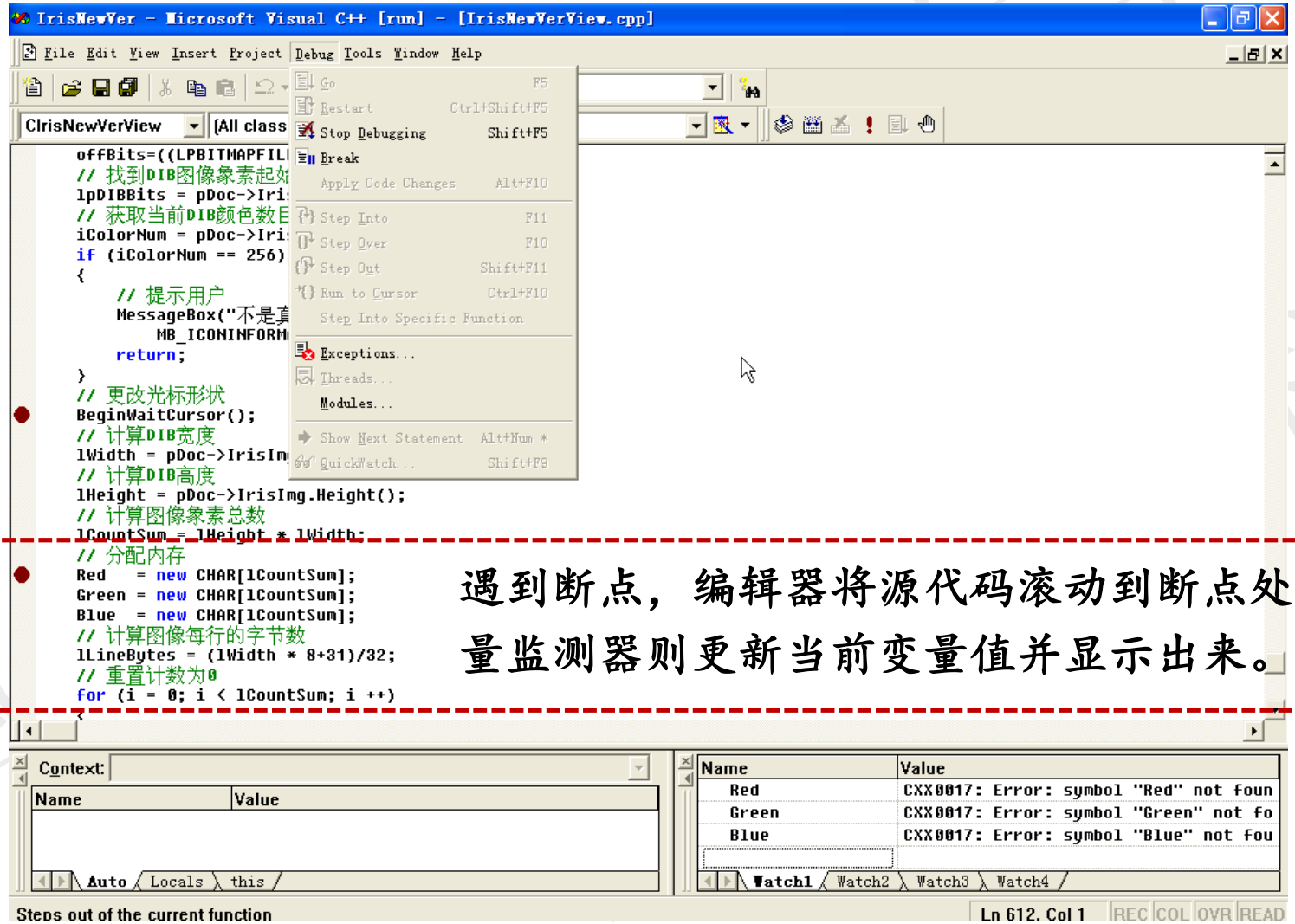
事件系统-连接机制(2/2)

- In some treatments, connectors are event-event bindings (在某些情况下，一个事件也可能触发其他事件，形成事件链).



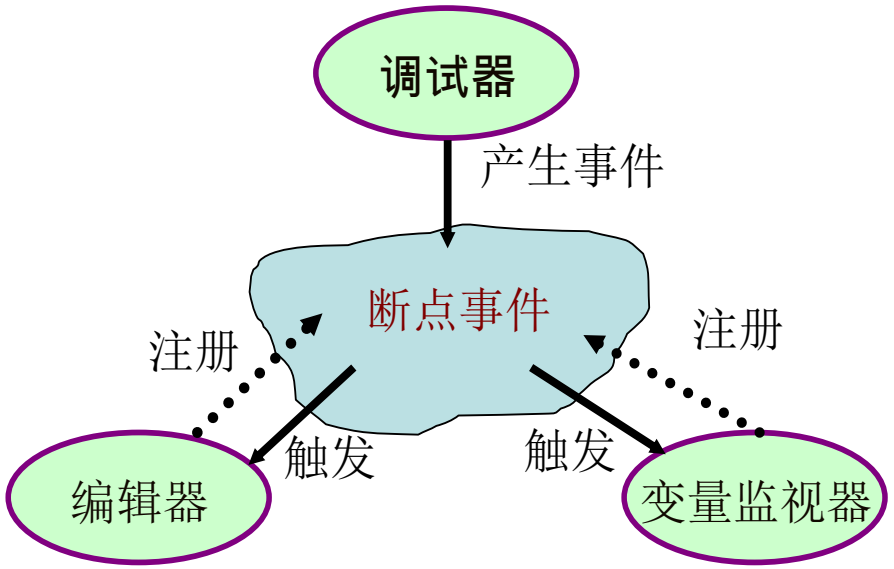
事件系统-应用实例1(1/2)

Example：调试器中的断点处理?



遇到断点，编辑器将源代码滚动到断点处，变量监测器则更新当前变量值并显示出来。How?

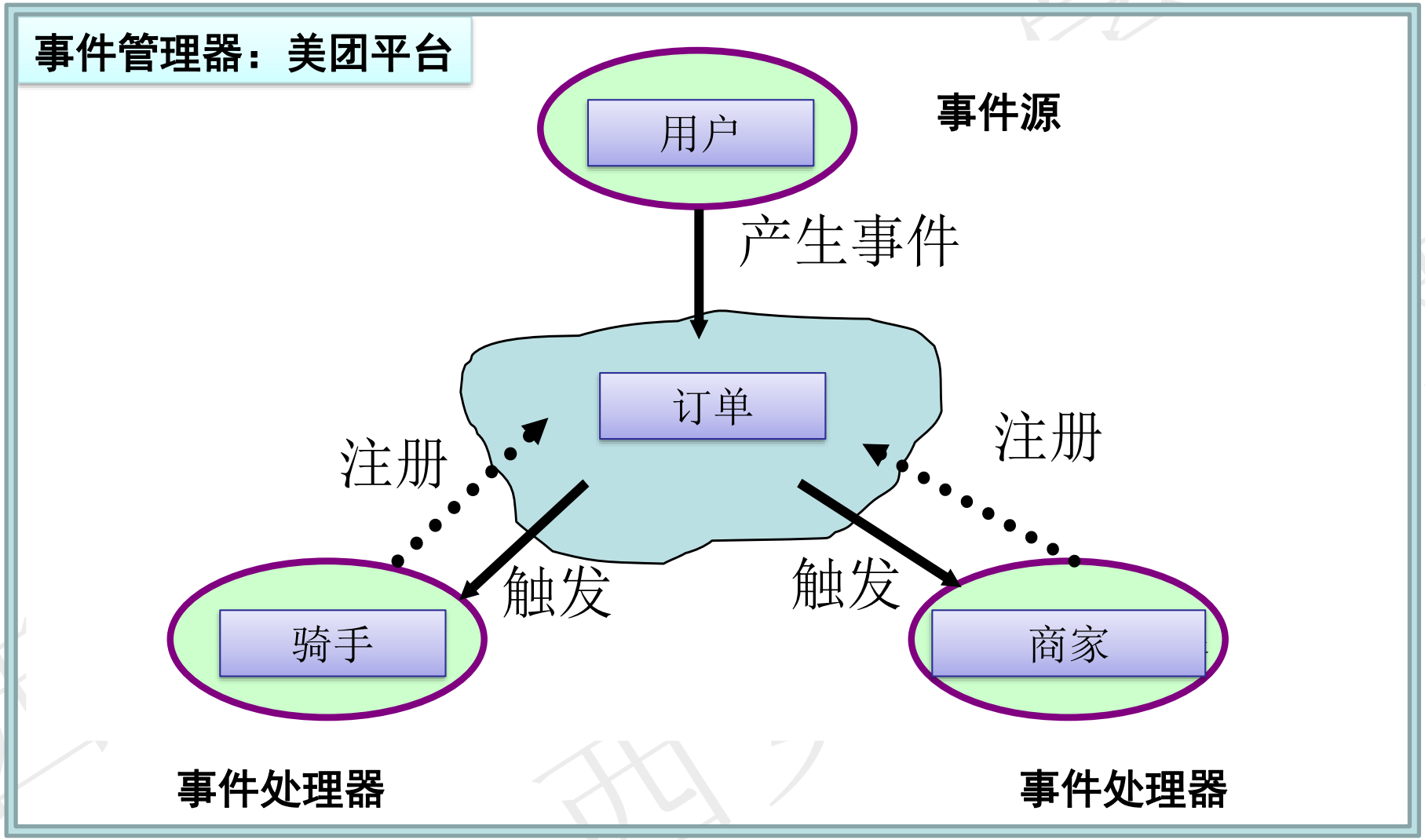
事件系统-应用实例1(2/2)



- **Event Source：debugger**（*调试器*）
- **Event Handler：editor and variable monitor**（*编辑器与变量监视器*）
- **Event Manager：IDE**（*集成开发环境*）

- **How?**
  - ✓ **编辑器与变量监视器向调试器注册，接收“断点事件”；**
  - ✓ **遇到断点，调试器发布事件，从而触发“编辑器”与“变量监测器”**
  - ✓ **编辑器将源代码滚动到断点处；**
  - ✓ **变量监测器则更新当前变量值并显示出来。**

事件系统-应用实例2(1/1)



事件系统派遣机制-调度机制(1/2)

当事件发生时，已向此事件注册过的过程被激发并执行。事件将以何种方式派遣到已注册的过程？过程将如何执行？

- **Two strategies**
  - ✓ EventManager without a central dispatcher module (无独立调度派谴模块的事件管理器)
  - ✓ EventManager with separated dispatcher module (带有独立派谴模块的事件管理器)

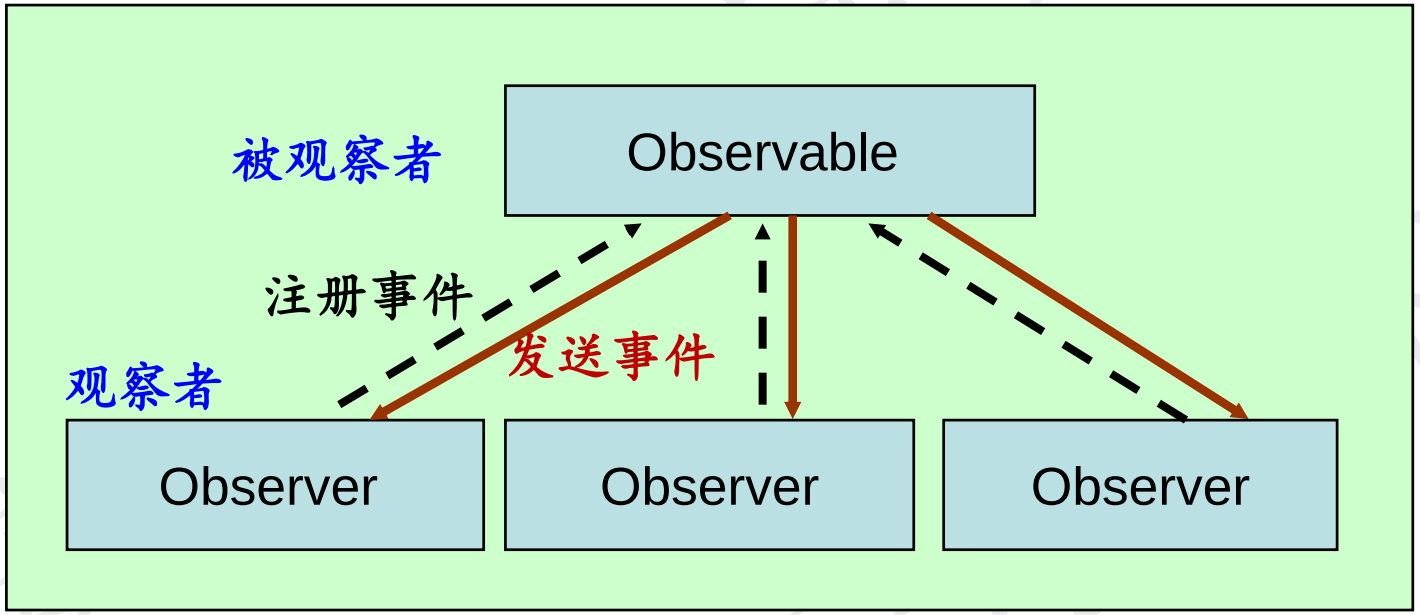
事件系统派遣机制-调度机制(1/2)

无独立调度模块的事件系统

- This module is usually called Observable/Observer (被观察者/观察者).
- Each module allows other modules to declare interest in events that they are sending. (每一个模块都允许其他模块向自己所能发送的某些消息表明兴趣)
- Whenever a module sends an event, it sends that event to exactly those modules that registered interest in that event. (当某一模块发出某一事件时，它自动将这些事件发布给那些曾经向自己注册过此事件的模块)

事件系统派遣机制-调度机制(1/2)

被观察者/观察者模式



Observable/Observer module

Register Event　　Send event

事件系统派遣机制-调度机制(1/2)

有独立事件派遣模块的事件系统

事件派遣模块?

The dispatcher module is responsible for receiving all incoming events and dispatching them to other modules in the system.

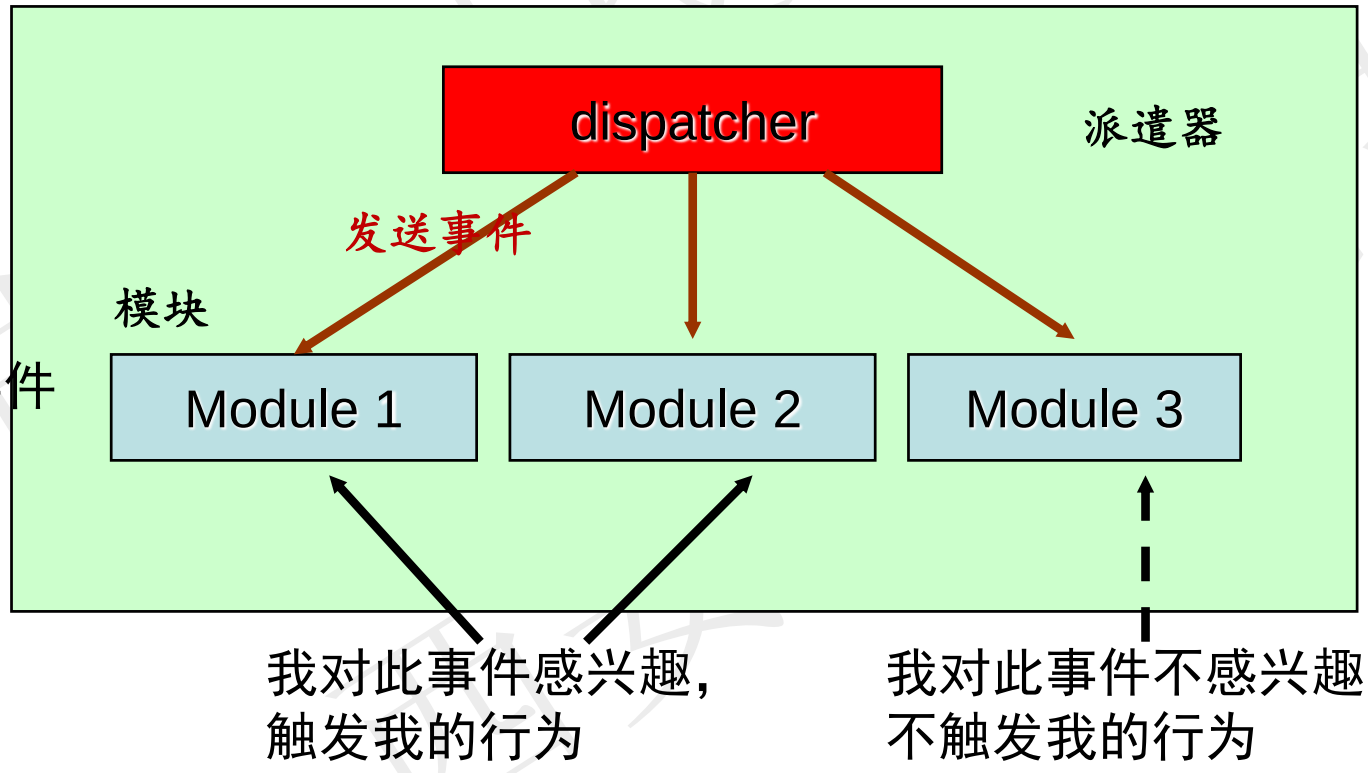
事件派遣模块是负责接收到来的事件并派遣它们到其它模块。

怎样派遣?

- 全广播式(All broadcasting)：派遣模块将事件广播到所有的模块，但只有感兴趣的模块才去取出事件并触发自身的行为；
- 选择广播式(Selected broadcasting)：派遣模块将事件送到那些已经注册的模块中。

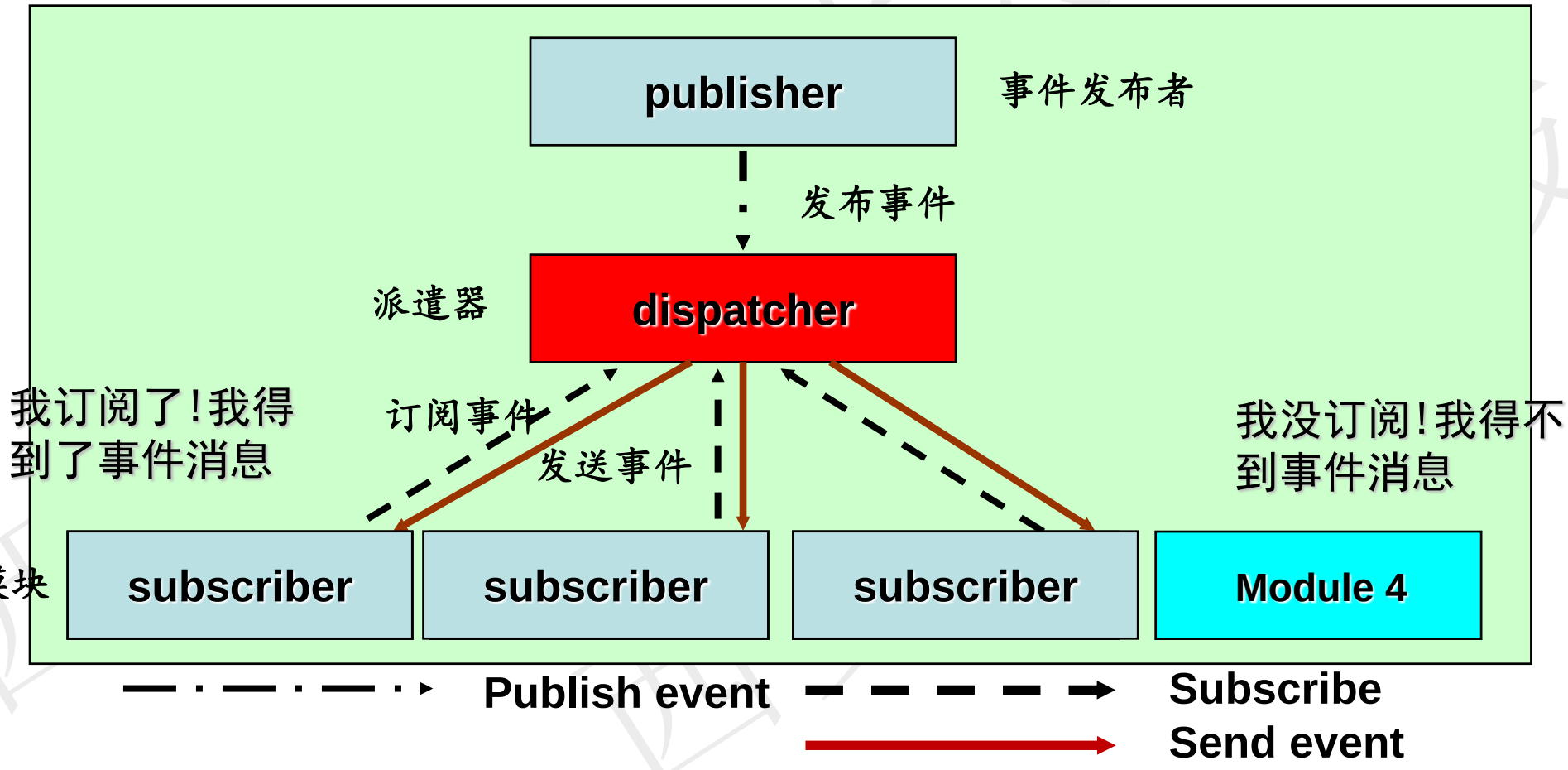
事件系统派遣机制-调度机制(1/2)

全广播式：无目的广播，接受者自行决定是否加以处理或者简单抛弃



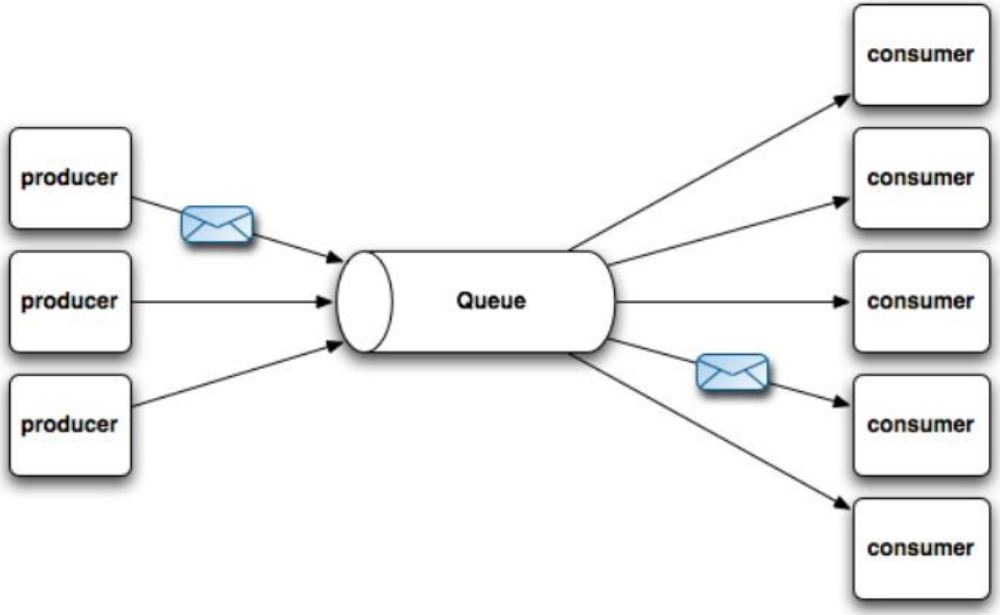
事件系统派遣机制-调度机制(1/2)

选择广播式：事件只发送给特定模块



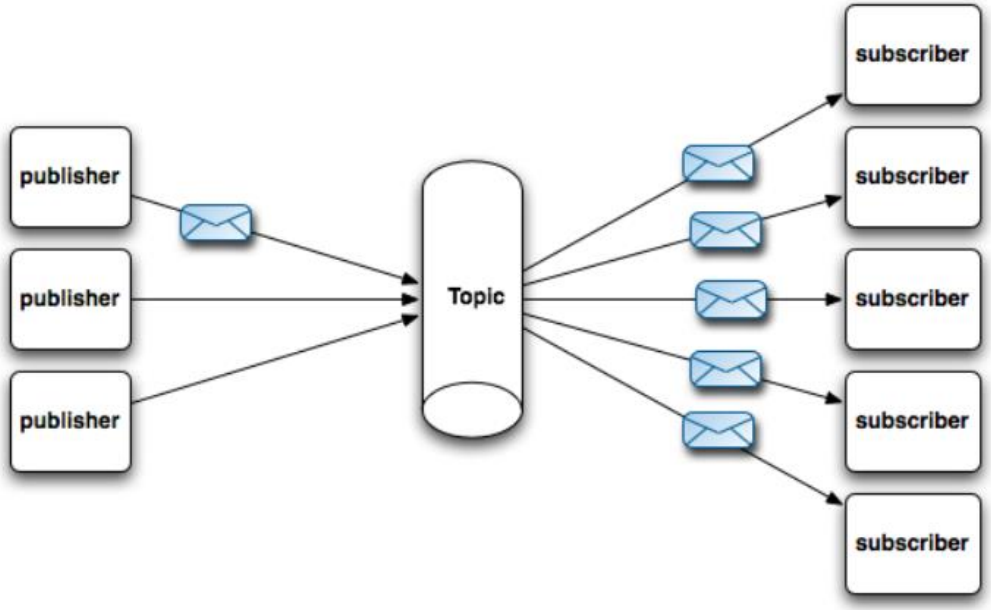
事件系统派遣机制-调度机制(1/2)

选择广播式的两种策略：基于事件被执行的方式



Point-to-Point (message queue)

(点对点模式：基于消息队列)



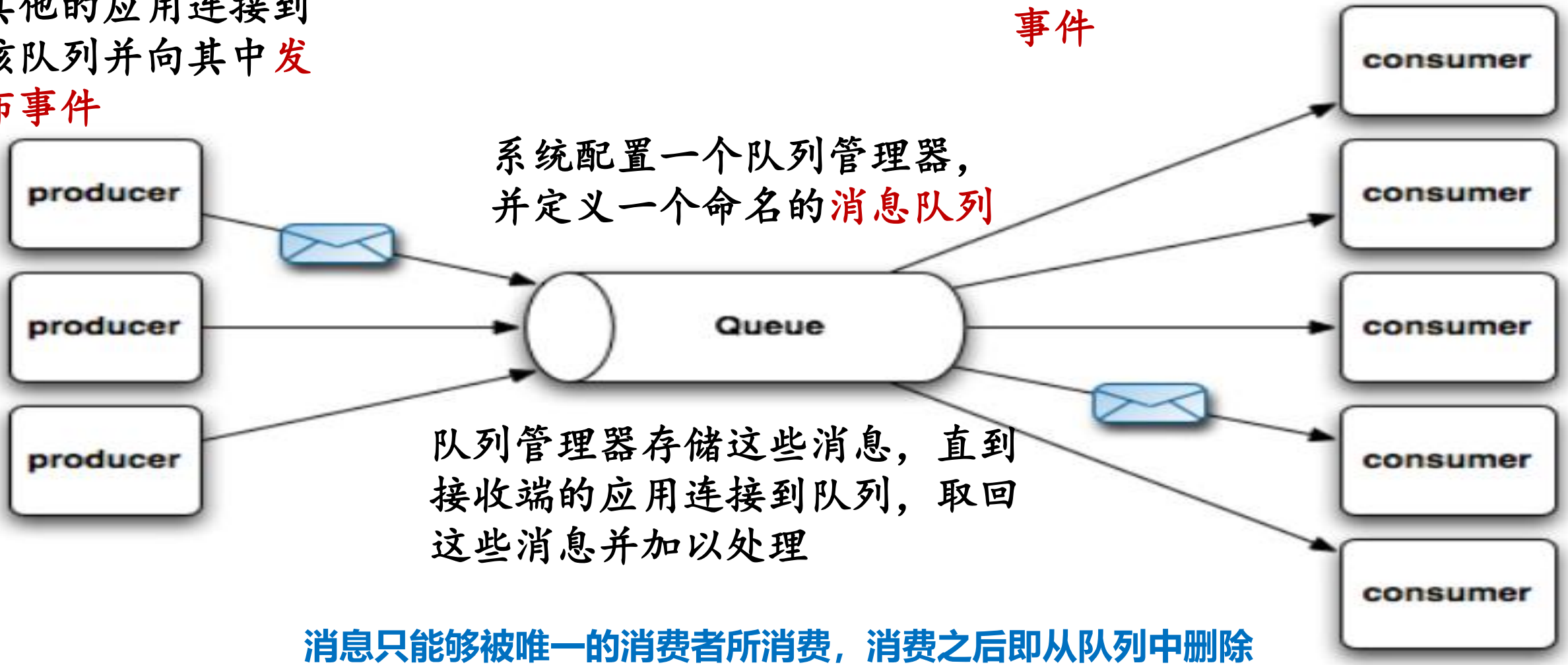
Publish-Subscribe

(发布-订阅模式)

事件系统派遣机制-调度机制(1/2)

某个应用向消息队列注册，以监听并处理被放置在队列里的事件

其他的应用连接到该队列并向其中发布事件



系统配置一个队列管理器，并定义一个命名的消息队列

队列管理器存储这些消息，直到接收端的应用连接到队列，取回这些消息并加以处理

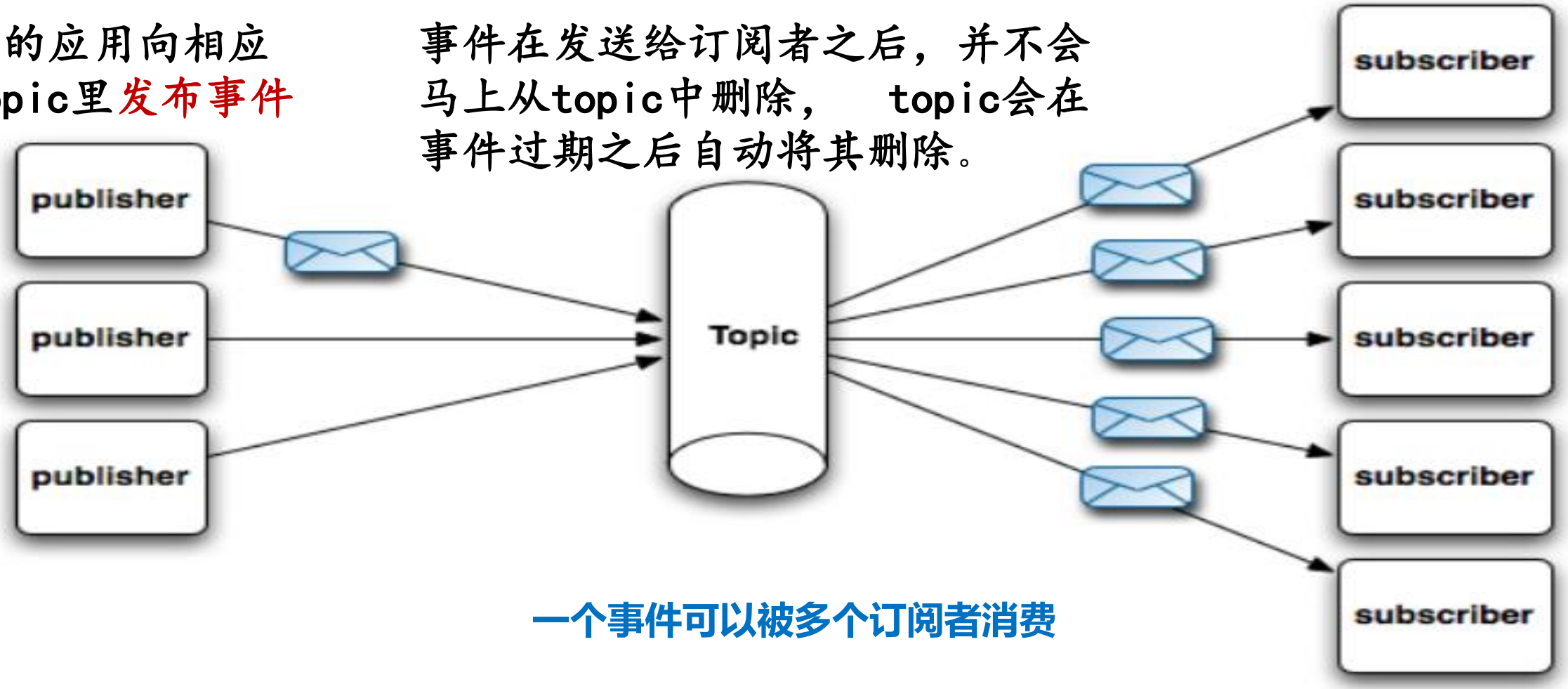
消息只能够被唯一的消费者所消费，消费之后即从队列中删除

事件系统派遣机制-调度机制(1/2)

某个应用向topic注册，以监听并处理被放置在topic里的事件

其他的应用向相应的topic里发布事件

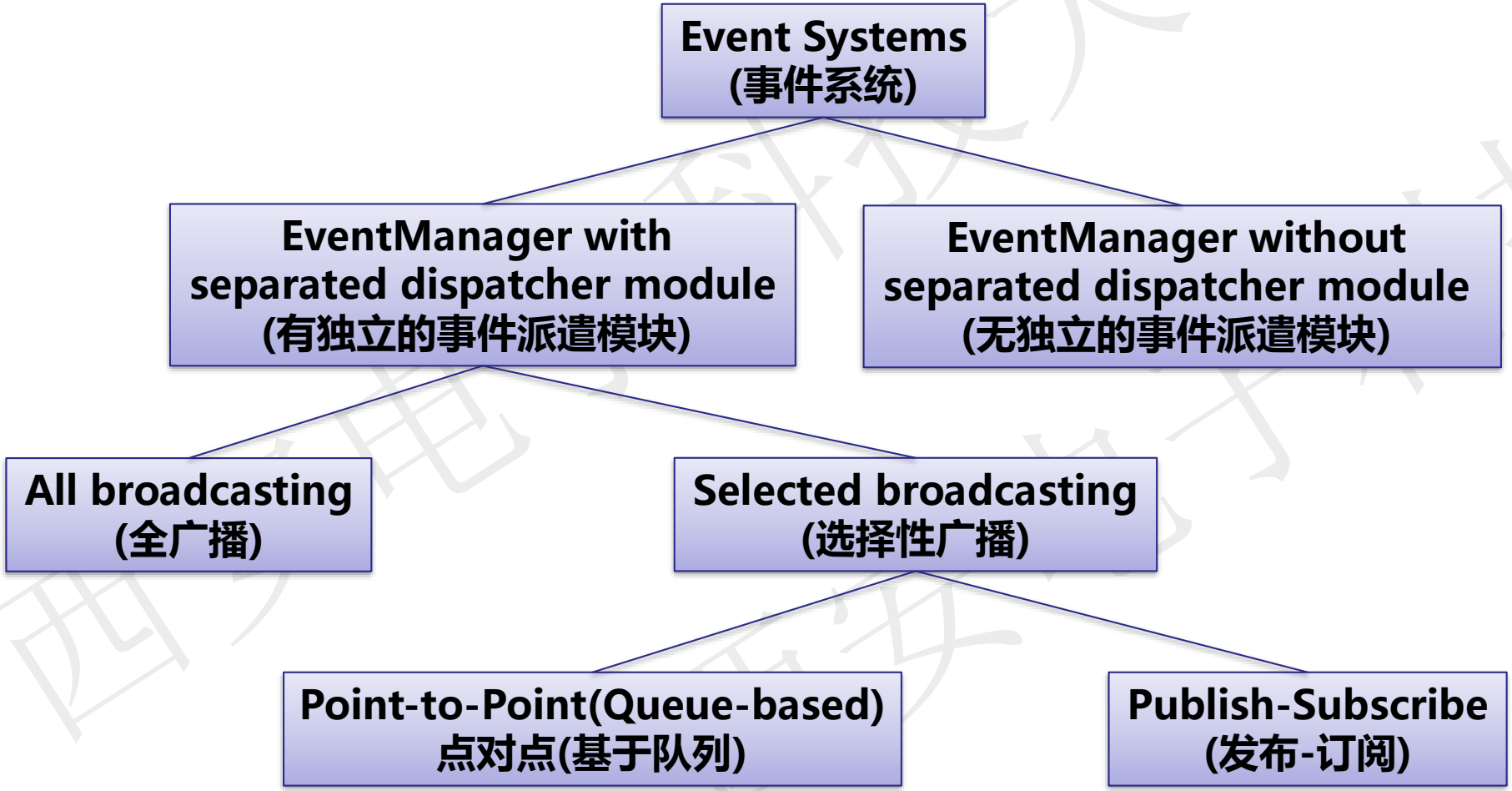
事件在发送给订阅者之后，并不会马上从topic中删除， topic会在事件过期之后自动将其删除。



一个事件可以被多个订阅者消费

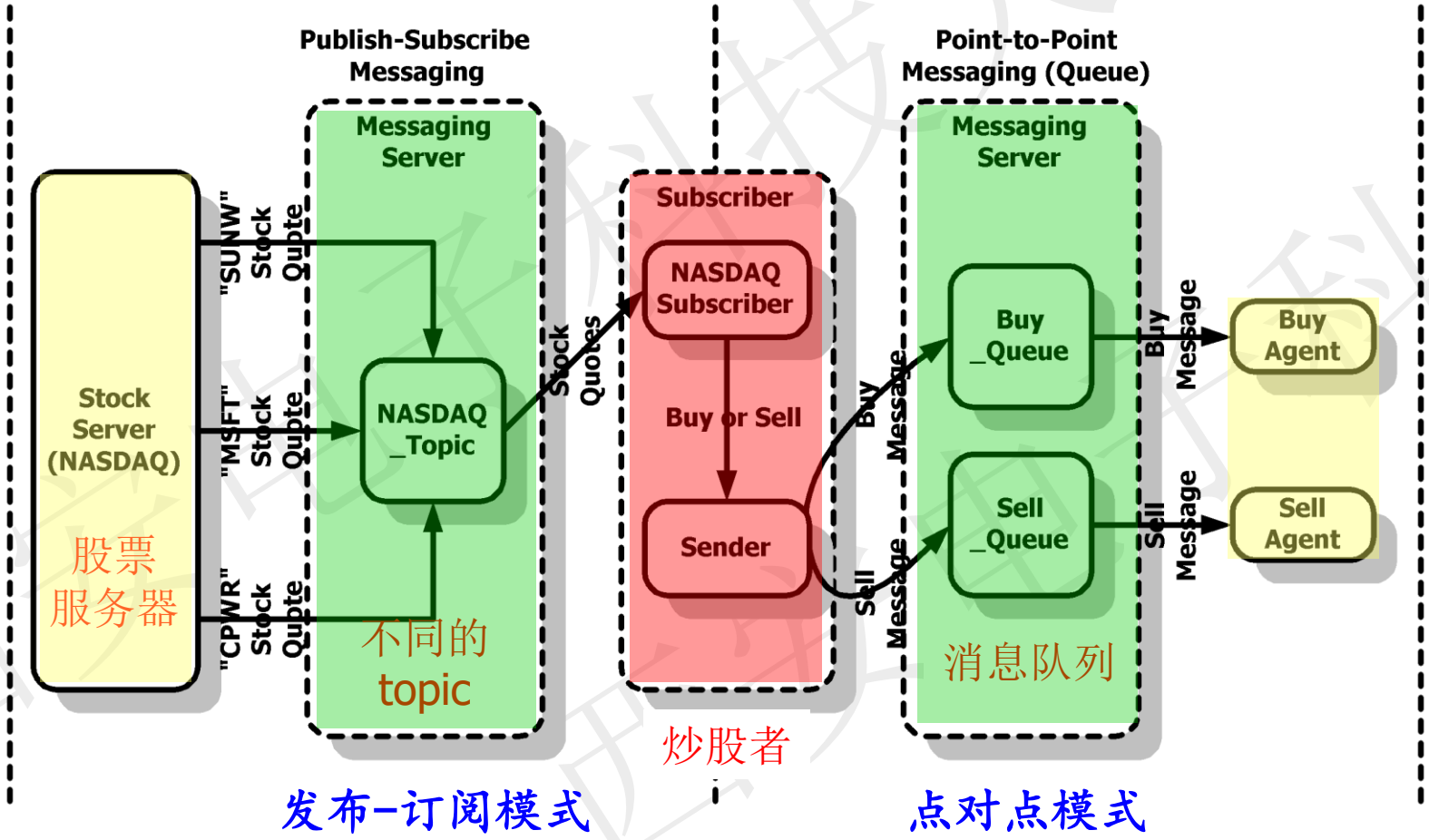
事件系统派遣机制-调度机制(1/2)

按照调度机制对事件系统的分类



事件系统派遣机制-调度机制(1/2)

Example：“股票交易平台”的事件调度机制设计



事件系统派遣机制-调度机制(1/2)

思考：“外卖平台”的事件调度机制设计?

商家接单：用户-骑手

事件派遣机制?

骑手接单：用户-商家

事件派遣机制?

西安电子科技大学
XIDIAN UNIVERSITY

外卖平台下单勾选“不需要餐具”选项

计算机科学与技术学院微信

课程讨论群

蔺一帅 讲师 硕导

邮箱：yslin@xidian.edu.cn

主页：https://web.xidian.edu.cn/yslin/