# 中介者模式

## 简介

中介者模式（Mediator Pattern）是一种行为型设计模式，它通过引入一个中介者对象来封装一组对象之间的交互。中介者模式的目的是减少对象之间的直接通信，将其转移到中介者对象上，从而降低系统的耦合度。

在中介者模式中，对象之间不再直接相互通信，而是通过中介者对象来进行通信。这种方式可以使各个对象之间的关系更为灵活，减少耦合，使系统更易于维护和扩展。

因此，通过中介公司联系供货商进行供货的行为适合使用中介者模式。

## 场景

联系两家中介(瓜子二手车、汽车之家)，通过他们联系不同的供货商（特斯拉，NIO，小鹏汽车，比亚迪）进行供货。

## 实现描述

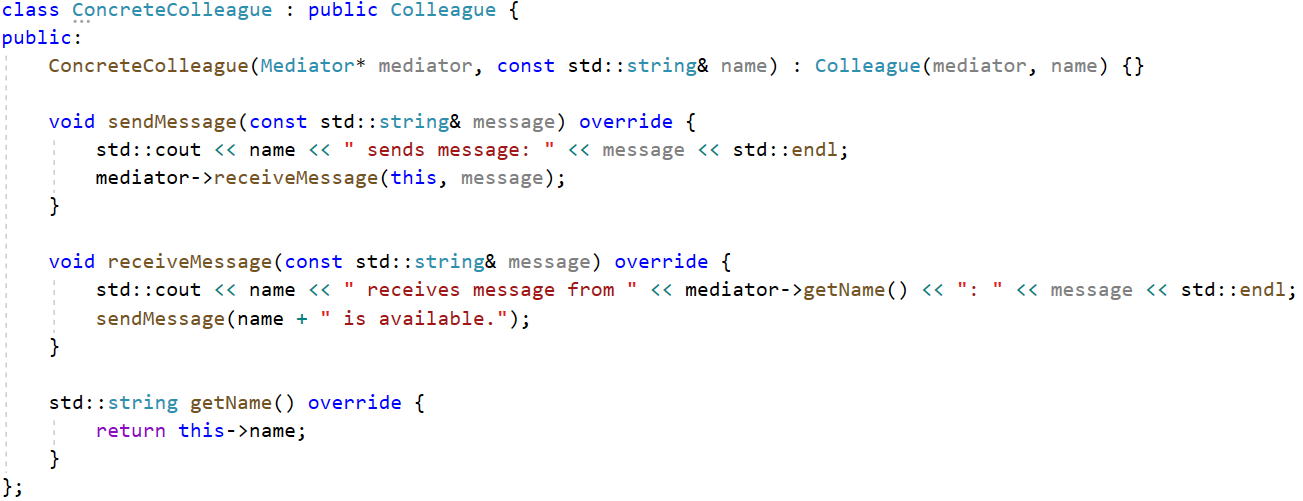
联系两家中介(瓜子二手车、汽车之家)，通过他们联系不同的供货商（特斯拉，NIO，小鹏汽车，比亚迪）进行供货。

首先定义抽象中介者类和同事类，分别负责中间商和供货商，供货商需要知道自己的中介者是谁：



然后实现具体中介者类和同事类，中介者存储自己负责的所有供货商表，除此之外还需要实现类之间的通信：

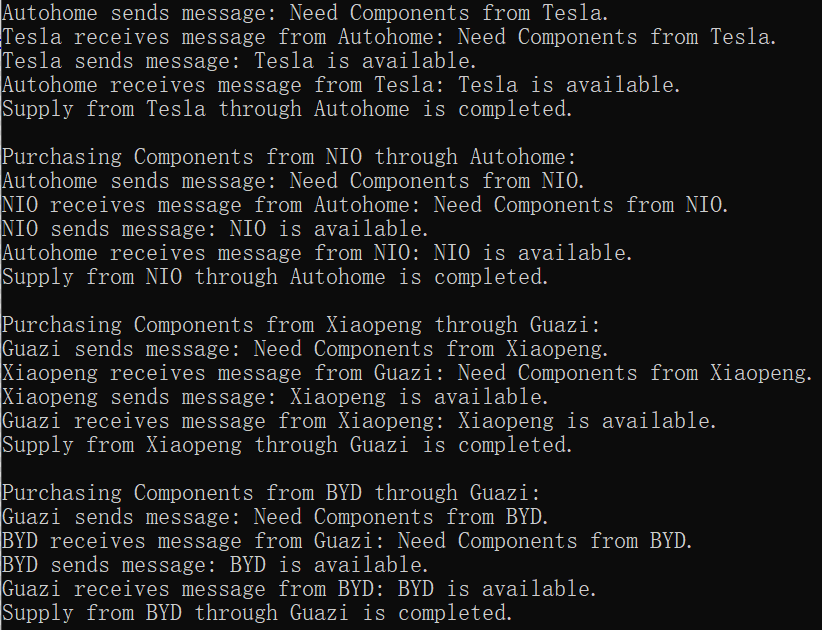




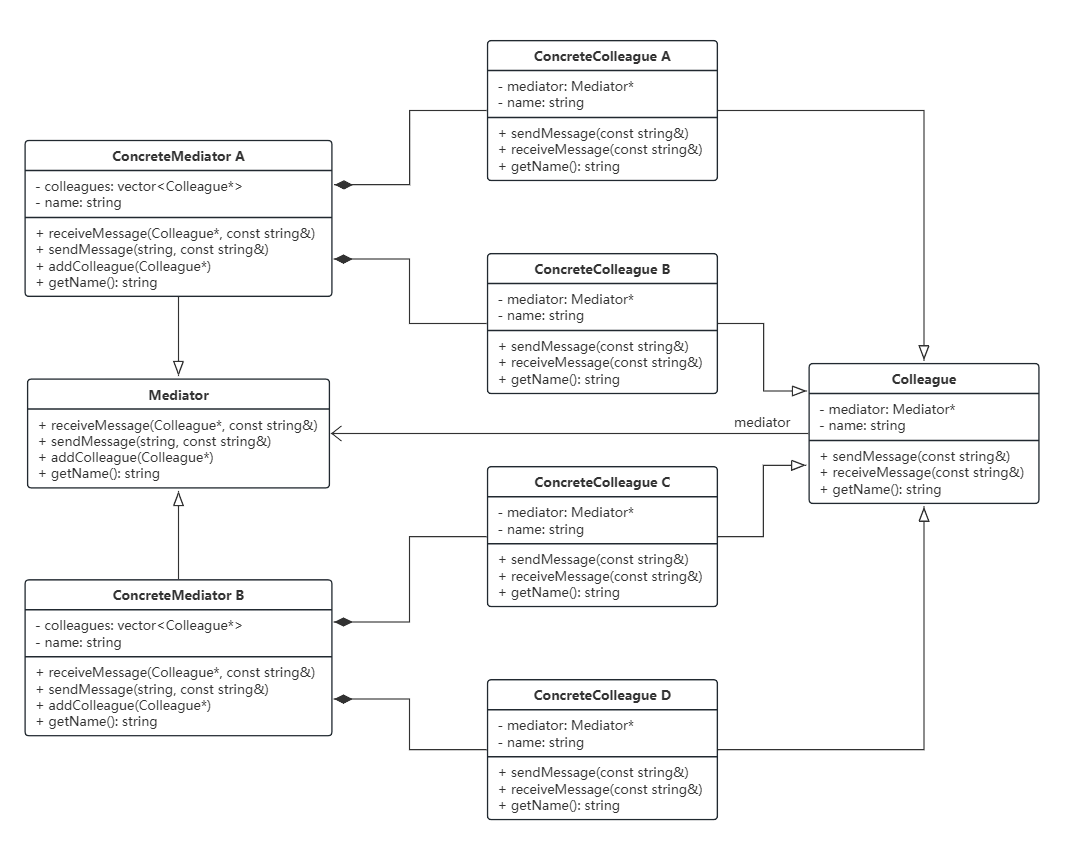
系统运行时，首先确定中介者、供货商以及其相互之间的关系，然后发送请求供货信息，消息通过中介者传递给供货商，再通过中介者返回消息：



最后输出如下，客户通过联系瓜子二手车这一中介者完成从小鹏汽车和比亚迪的供货；通过联系汽车之家这一中介者完成从特斯拉和蔚来的供货：

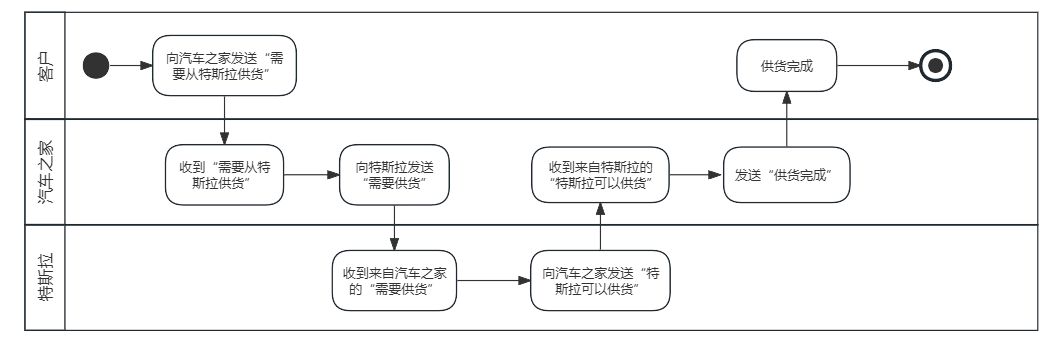


## 类图



## 流程图

以通过汽车之家向特斯拉请求供货为例，客户向特斯拉发送请求供货的信息和特斯拉完成供货的信息都需要经过汽车之家这一中介者：



## 代价分析

1. **复杂性增加：**
   * **成本：** 引入中介者对象会增加系统的复杂性。中介者需要处理多个同事对象之间的交互，可能导致中介者本身变得复杂，难以维护和理解。
   * **效益：** 中介者模式的复杂性可能在一些简单的系统中显得不划算。如果系统中对象之间的交互关系简单且稳定，引入中介者可能会显得繁琐。
2. **耦合度降低：**
   * **成本：** 中介者模式的主要优势是降低对象之间的直接通信，减少耦合。然而，这也意味着对象不能直接通信，而必须通过中介者，这可能导致系统的性能略微下降。
   * **效益：** 降低耦合度有助于系统的维护和扩展。在大型系统中，中介者模式可以使系统更具弹性，更容易适应变化。
3. **维护和扩展：**
   * **成本：** 中介者对象需要处理各个同事对象之间的关系，因此维护中介者可能会增加一些成本。
   * **效益：** 对于频繁变化的系统，中介者模式可以使修改和扩展更加容易。如果对象之间的关系需要调整，只需修改中介者而不是所有相关的对象。
4. **通信效率：**
   * **成本：** 由于所有通信都经过中介者，可能会导致一定的通信效率损失。
   * **效益：** 中介者模式可以实现更灵活的通信方式，有助于管理和控制对象之间的交互。在一些情况下，通信效率的损失可能是可以接受的。
5. **设计和开发时间：**
   * **成本：** 引入中介者模式可能需要额外的设计和开发时间，尤其是在系统初始阶段。
   * **效益：** 在长期来看，中介者模式可以减少系统的维护成本，并使系统更容易适应变化。在大型系统中，这种投资可能是值得的。

在选择是否使用中介者模式时，需要综合考虑系统的规模、变化频率、性能要求以及团队的开发经验等因素。在一些小型且相对稳定的系统中，引入中介者模式可能并不划算，而在大型、频繁变化的系统中，中介者模式可能成为一个有价值的设计选择。

汽车的供货商有很多，各供货商通常负责不同的部件，且很有可能多次发生变化，如果每次供货商发生变化时都要重写客户和供货商之间的方法，则会很复杂，因此采用中介者模式，将对供货商的方法封装在中介者中，可以很好地优化供货流程。

## 其他

1. 中介者模式可以进一步优化，例如通过引入消息队列来实现异步通信，提高系统的响应性能。
2. 可以考虑使用观察者模式，使中介者和同事对象之间的关系更加灵活，以适应系统的变化。
3. 在其他的实际应用中，需要根据具体情况选择是否使用中介者模式，确保它真正带来了系统设计和维护的便利。