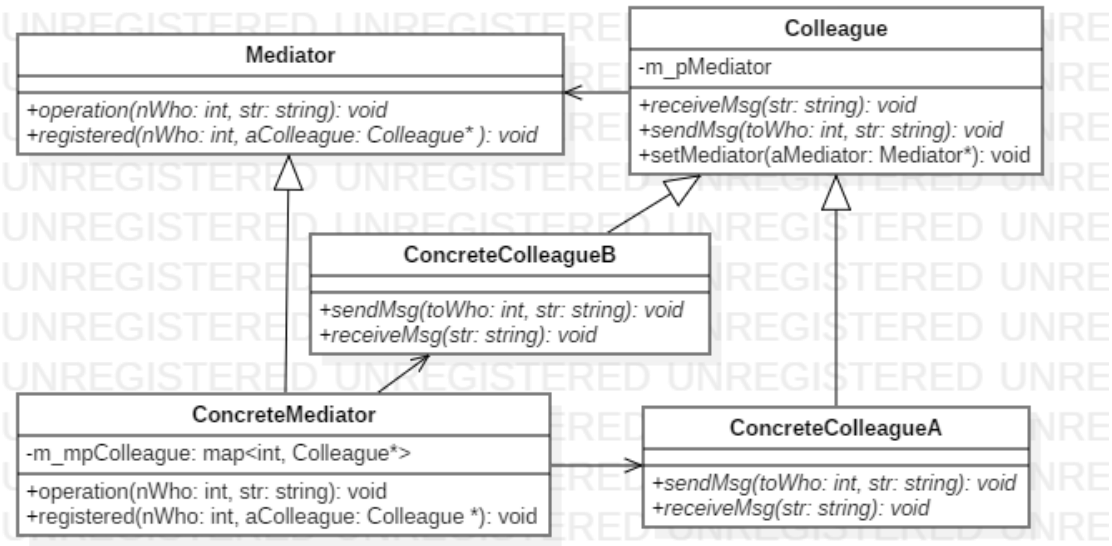
模式定义

中介者模式(Mediator Pattern)定义：用一个中介对象来封装一系列的对象交互，中介者使各对象不需要显式地相互引用，从而使其耦合松散，而且可以独立地改变它们之间的交互。中介者模式又称为调停者模式，它是一种对象行为型模式。

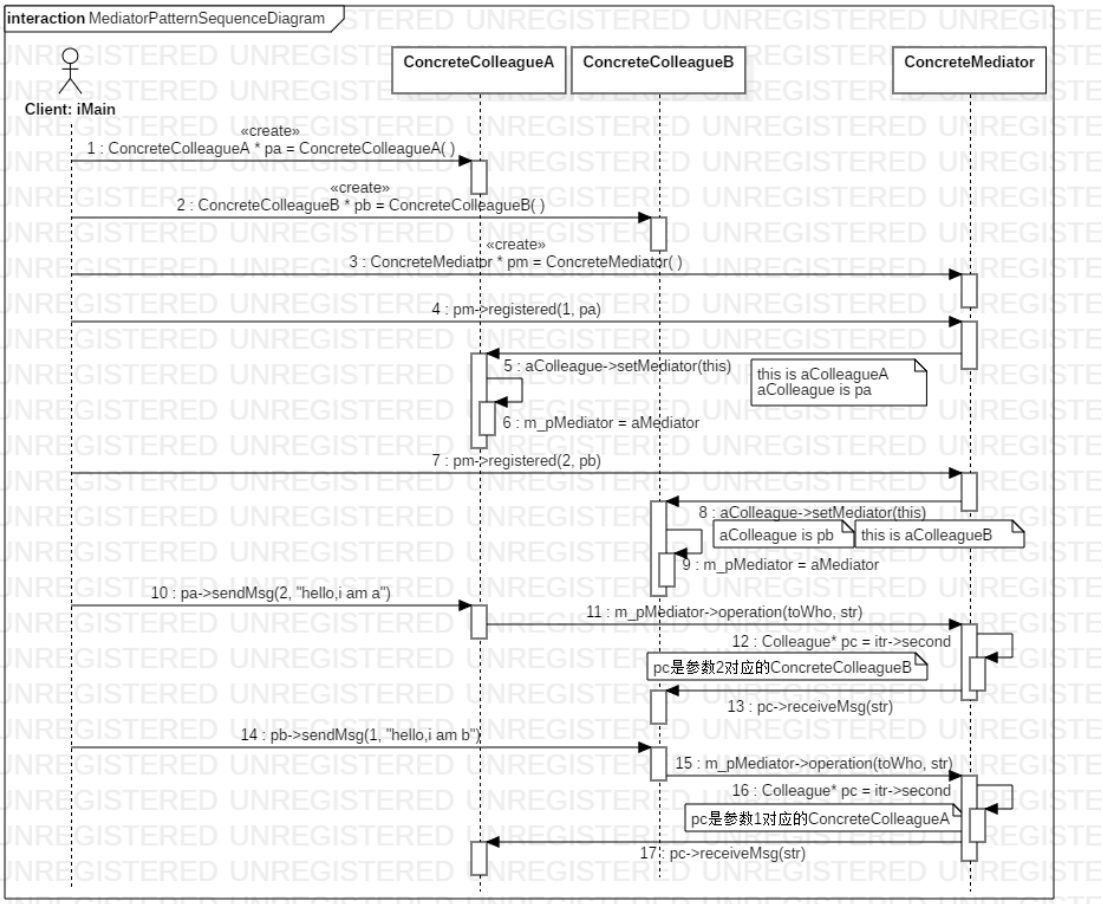
模式结构

中介者模式包含如下角色：

- Mediator: 抽象中介者
- ConcreteMediator: 具体中介者
- Colleague: 抽象同事类
- ConcreteColleague: 具体同事类



时序图



关键代码

```
int main()
{
    ConcreteColleagueA * pa = new ConcreteColleagueA();
    ConcreteColleagueB * pb = new ConcreteColleagueB();
    ConcreteMediator * pm = new ConcreteMediator();
    cout << "测试pm:  " << pm << endl;

    pm->registered(1, pa);
    pm->registered(2, pb);

    // sendmsg from a to b
    pa->sendMsg(2, "hello,i am a");
    // sendmsg from b to a
    pb->sendMsg(1, "hello,i am b");

    delete pa, pb, pm;
```

```cpp
void ConcreteMediator::registered(int nWho, Colleague * aColleague)
{
    map<int, Colleague*>::const_iterator itr = m_mpColleague.find(nWho);
    if (itr == m_mpColleague.end())
    {
        m_mpColleague.insert(make_pair(nWho, aColleague));
        //同时将中介类暴露给colleague
        cout << "测试:   " << this << endl;
        aColleague->setMediator(this);
    }
}
```
ConcreteColleagueA　　ConcreteMediator

```cpp
void ConcreteColleagueA::sendMsg(int toWho, string str)
{
    cout << "send msg from colleagueA,to:" << toWho << endl;
    m_pMediator->operation(toWho, str);
}
```
ConcreteMediator

```cpp
void ConcreteMediator::operation(int nWho, string str)
{
    map<int, Colleague*>::const_iterator itr = m_mpColleague.find(nWho);
    if (itr == m_mpColleague.end())
    {
        cout << "not found this colleague!" << endl;
        return;
    }
    Colleague* pc = itr->second;
    pc->receiveMsg(str);
}
```
根据注册时参数nWho对应的ConcreteColleagueA或ConcreteColleagueB

```cpp
void ConcreteColleagueB::receiveMsg(string str)
{
    2对应的是ConcreteColleagueB
    cout << "ConcreteColleagueB reveivemsg:" << str << endl;
}
```

测试结果

```cpp
int main()
{
    ConcreteColleagueA * pa = new ConcreteColleagueA();
    ConcreteColleagueB * pb = new ConcreteColleagueB();
    ConcreteMediator * pm = new ConcreteMediator();
    cout << "测试pm:   " << pm << endl;

    pm->registered(1, pa);
    pm->registered(2, pb);

    // sendmsg from a to b
    pa->sendMsg(2, "hello,i am a");
    // sendmsg from b to a
    pb->sendMsg(1, "hello,i am b");

    delete pa, pb, pm;


    system("pause");
```

```
D:\Design Pattern1\DesignPatterDemo\Behavioral Patter
测试pm:   00406960
测试:   00406960
测试:   00406960
send msg from colleagueA,to:2
ConcreteColleagueB reveivemsg:hello,i am a
send msg from colleagueB,to:1
ConcreteColleagueA reveivemsg:hello,i am b
请按任意键继续. . .
```