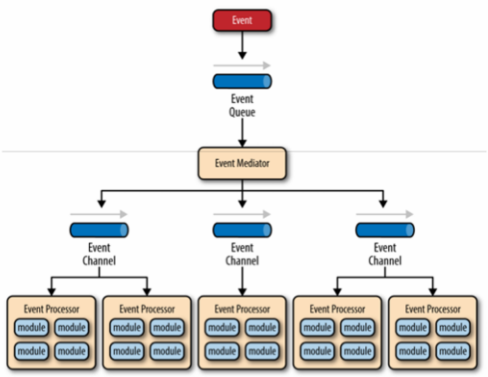
事件驱动架构之中介拓扑结构

体系结构之实现



依照上文图片，简单实现代码如下：

定义 Event：

```java
public class Event {
    private String name;
    private Object data;

    public Event(String name, Object data) {
        this.name = name;
        this.data = data;
    }

    public String getName() {
        return name;
    }

    public Object getData() {
        return data;
    }
}
```

定义 EventListener 接口，用于监听和处理事件：

```java
public interface EventListener {
    void onEvent(Event event);
}
```

实现 EventManager 类，它作为事件中介，负责注册监听器、触发事件等：

```java
import java.util.ArrayList;
import java.util.List;

public class EventManager {
    private List<EventListener> listeners = new ArrayList<>();

    // 注册监听器
    public void registerListener(EventListener listener) {
        listeners.add(listener);
    }

    // 移除监听器
    public void unregisterListener(EventListener listener) {
        listeners.remove(listener);
    }

    // 触发事件
    public void triggerEvent(Event event) {
        for (EventListener listener : listeners) {
            listener.onEvent(event);
        }
    }
}
```

创建一个简单的示例来演示如何使用这些类：

```java
public class EventDemo {
    public static void main(String[] args) {
        EventManager eventManager = new EventManager();

        // 创建监听器并注册
        EventListener listener1 = event -> {
            System.out.println("Listener 1 received event: " + event.getName());
        };
        eventManager.registerListener(listener1);

        // 创建并触发一个事件
        Event event = new Event("myEvent", "Some data");
        eventManager.triggerEvent(event);

        // 移除监听器（如果需要）
        eventManager.unregisterListener(listener1);
    }
}
```

场景构想之 Mud 游戏项目

mud 游戏以及衍生的单机类文字游戏就是典型的以用户事件驱动为主要结构的软件项目。使用事件驱动架构为用户事件进行实时交互和反馈。

我们以武侠 Mud 项目为例，作为玩家，其交易系统就是最典型的以事件驱动架构，包括货币交易、物品买卖与结算等等。除此之外，玩家在进行游玩时，进行状态改变，所对应的事件就由服务器交给不同的事件处理器处理，比如说，玩家战斗时，造成的伤害即视为一个事件，由服务器将此事件交于专门伤害系统处理，而非由金融系统进行计算。

ai 功能的实现：作为对话功能，引入 AI 作为 NPC 系统，也是一个可行的办法。

与其他网游不同的是，mud 游戏的用户所有操作均由指令的形式提交给服务器，只需要简单包装就能作为一个事件。所以采用事件驱动架构效果最佳。

技术实现概述

客户端：以 JavaFX 为展示页面的 Java 程序。理由：团队熟悉相关控件，学习成本低，且 MUD 游戏无需复杂界面，满足需求。

后端：以 SSM 框架为核心。理由：学习成本低，框架应用广泛。

数据库：MySQL。