分布式对象RMI实验报告

学号：13331158 林义涵

agendaService 接口设计

为了使分布式议程服务具有可扩展性，AgendaServiceInterface 接口采用抽象工厂模式，AgendaServiceInterface 接口继承java.rmi.Remote 类，并声明抽象方法 excuteCommand()。

```
// AgendaService接口 package agenda.service;
import java.io.IOException;
import org.jdom.JDOMException;
import agenda.exception.AgendaException;
public interface AgendaServiceInterface extends java.rmi.Remote {
    public abstract String[] excuteCommand(String command)
        throws AgendaException, java.rmi.RemoteException, IOException, JDOMException;
}
```

agendaService 服务端设计

为了使服务端业务处理层具有可扩展性，在服务端架构设计时采用抽象工厂模式，即 Command 包的各种命令类，都继承同一个命令接口 CmdFactory。

```
// 命令工厂接口
package agenda.command;
public interface CmdFactory {
 // 创建命令对象
  public Command createCommand();
}
```

Command是命令接口，其中 parse()方 法用于检查命令参考列表，excute()方法用于执行命令。

CommandFactory是命令工厂类，私有 构造函数 CommandFactory()实现单实例模式。getInstance()方法返回命令工厂的唯一实例的引用，createCommand()方法根据用户请求命令名，创建命令实例。例如请求的命令是 “register”，则由命令工厂的方法 createCommand()创建 RegisterCommand类的实例对象。