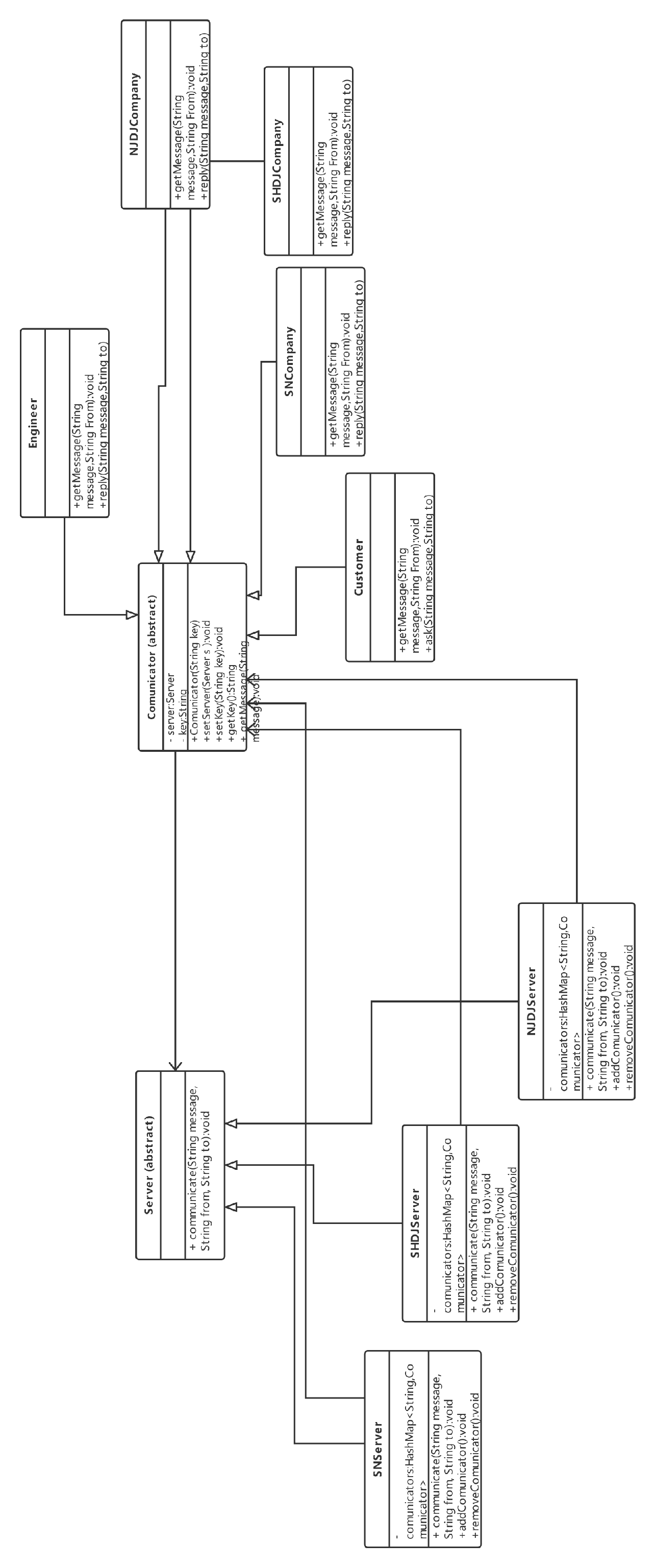
、



**public** **abstract** **class** Server {

**public** **abstract** **void** communicate(String message,String from,String to);

}

**import** java.util.HashMap;

**public** **class** SNServer **extends** Server {

**private** HashMap<String,Comunicator> comunicators = **new** HashMap<String,Comunicator>();

**public** **void** addComunicator(Comunicator c) {

**this**.comunicators.put(c.getKey(), c);

}

**public** **void** removeComunicator(String key) {

**this**.comunicators.remove(key);

}

@Override

**public** **void** communicate(String message, String from, String to) {

// **TODO** 自动生成的方法存根

**if**(**this**.comunicators.get(to)!=**null**) {

**this**.comunicators.get(to).getMessage(message,from);

}

}

}

**public** **class** SHDJServer **extends** Server {

**private** HashMap<String,Comunicator> comunicators = **new** HashMap<String,Comunicator>();

**public** **void** addComunicator(Comunicator c) {

**this**.comunicators.put(c.getKey(), c);

}

**public** **void** removeComunicator(String key) {

**this**.comunicators.remove(key);

}

@Override

**public** **void** communicate(String message, String from, String to) {

// **TODO** 自动生成的方法存根

**if**(**this**.comunicators.get(to)!=**null**) {

**this**.comunicators.get(to).getMessage(message,from);

}

}

}

**public** **class** NJDJServer **extends** Server {

**private** HashMap<String,Comunicator> comunicators = **new** HashMap<String,Comunicator>();

**public** **void** addComunicator(Comunicator c) {

**this**.comunicators.put(c.getKey(), c);

}

**public** **void** removeComunicator(String key) {

**this**.comunicators.remove(key);

}

@Override

**public** **void** communicate(String message, String from, String to) {

// **TODO** 自动生成的方法存根

**if**(**this**.comunicators.get(to)!=**null**) {

**this**.comunicators.get(to).getMessage(message,from);

}

}

}

**public** **class** Customer **extends** Comunicator {

**public** Customer (String key){

**super**(key);

}

@Override

**public** **void** getMessage(String message,String from) {

// **TODO** 自动生成的方法存根

System.***out***.println("消费者收到 "+from +"的回复： "+message);

}

**public** **void** ask(String message,String to) {

**this**.server.communicate(message, **this**.getKey(), to);

}

}

**public** **class** SNCompany **extends** Comunicator {

**public** SNCompany(String key) {

**super**(key);

// **TODO** 自动生成的构造函数存根

}

@Override

**public** **void** getMessage(String message, String from) {

// **TODO** 自动生成的方法存根

System.***out***.println("苏宁公司收到 "+from +"的询问： "+message);

}

**public** **void** replay(String message,String to) {

**this**.server.communicate(message, **this**.getKey(), to);

}

}

**public** **class** SHDJCompany **extends** Comunicator {

**public** SHDJCompany(String key) {

**super**(key);

// **TODO** 自动生成的构造函数存根

}

@Override

**public** **void** getMessage(String message, String from) {

// **TODO** 自动生成的方法存根

System.***out***.println("上海大金公司收到 "+from +"的消息： "+message);

}

**public** **void** replay(String message,String to) {

**this**.server.communicate(message, **this**.getKey(), to);

}

}

**public** **class** NJDJCompany **extends** Comunicator {

**public** NJDJCompany(String key) {

**super**(key);

// **TODO** 自动生成的构造函数存根

}

@Override

**public** **void** getMessage(String message, String from) {

// **TODO** 自动生成的方法存根

System.***out***.println("南京大金公司收到 "+from +"的消息： "+message);

}

**public** **void** replay(String message,String to) {

**this**.server.communicate(message, **this**.getKey(), to);

}

}

**public** **class** Engineer **extends** Comunicator {

**public** Engineer(String key) {

**super**(key);

// **TODO** 自动生成的构造函数存根

}

@Override

**public** **void** getMessage(String message,String from) {

// **TODO** 自动生成的方法存根

System.***out***.println("工程师收到 "+from +"的消息： "+message);

}

**public** **void** replay(String message,String to) {

**this**.server.communicate(message, **this**.getKey(), to);

}

}

**public** **class** test {

**public** test() {

// **TODO** 自动生成的构造函数存根

}

**public** **static** **void** main(String[] args) {

// **TODO** 自动生成的方法存根

Customer customer = **new** Customer("用户");

SNCompany snCompany = **new** SNCompany("苏宁公司");

SHDJCompany shdjCompany = **new** SHDJCompany("上海大金公司");

NJDJCompany njdjCompany = **new** NJDJCompany("南京大金公司");

Engineer engineer = **new** Engineer("工程师");

SNServer snServer = **new** SNServer(); //苏宁客服

SHDJServer shdjServer = **new** SHDJServer(); //上海大金公司客服

NJDJServer njdjServer = **new** NJDJServer(); //南京大金公司客服

//设置中介者

snCompany.setServer(snServer);

shdjCompany.setServer(shdjServer);

njdjCompany.setServer(njdjServer);

engineer.setServer(njdjServer);

System.***out***.println("1");

//设置交流者

snServer.addComunicator(customer);

snServer.addComunicator(snCompany);

shdjServer.addComunicator(customer);

shdjServer.addComunicator(shdjCompany);

njdjServer.addComunicator(customer);

njdjServer.addComunicator(njdjCompany);

njdjServer.addComunicator(engineer);

//流程

customer.setServer(snServer);

customer.ask("我上次在苏宁买的大金公司的空调安装有问题怎么办", "苏宁公司");

snCompany.replay("我们苏宁公司不负责售后，请联系上海大金公司客服", "用户");

customer.setServer(shdjServer);

customer.ask("我上次在苏宁买的贵公司的空调安装有问题怎么办,苏宁平台让我来找上海大金公司的客服", "上海大金公司");

shdjCompany.replay("我们问了相关部门，你买的空调是苏州大金工厂生产的，请联系南京大金公司客服", "用户");

customer.setServer(njdjServer);

customer.ask("我上次在苏宁买的贵公司的空调安装有问题，我问了上海大金公司客服，他让我来找南京大金公司，我自己查了一下用户手册里写了有一个部件，但是实际上确没有", "南京大金公司");

njdjCompany.replay("我们询问了相关部门,空调里是没有这个配件的", "用户");

customer.ask("但是用户手册上说是有的，这应该是你们的问题", "南京大金公司");

njdjCompany.replay("这个空调的配件好像停产了，我们让一个工程师到你家去看看吧", "用户");

njdjCompany.replay("去用户家修理一下空调", "工程师");

}

}

