

软件复用

应用程序开发项目文档

—— Group5



目录

一、项目需求.....	3
二、需求分析.....	3
1. 系统用例图.....	3
2. 系统时序图.....	4
三、架构与功能.....	5
1. Server 服务端.....	5
2. Client 客户端.....	6
四、组件的选择和使用.....	6
1. PM 组件.....	6
2. License 组件.....	6
3. apache-activemq-5.10.0 消息中间件.....	7
五、使用说明.....	7
六、关于测试.....	7

一、项目需求

该项目要求我们在之前所做的组件开发和选择基础上 ,进行基于 Team 名称查询功能的应用程序开发。主要有以下功能：

1. 接收某同学姓名，然后返回所属 Team 的名称

消息协议/格式自定

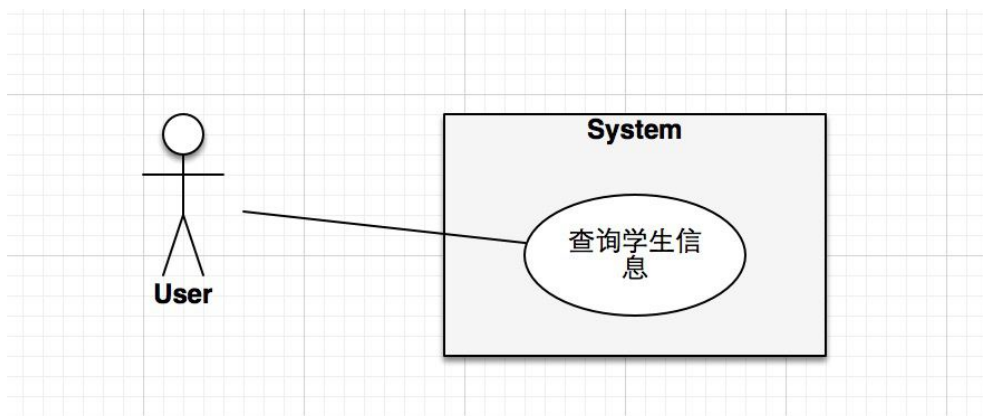
Team 配置：任意实现

Log4j 用于 Log 功能

2. 单元测试/功能测试 JUnit
3. 产品文档

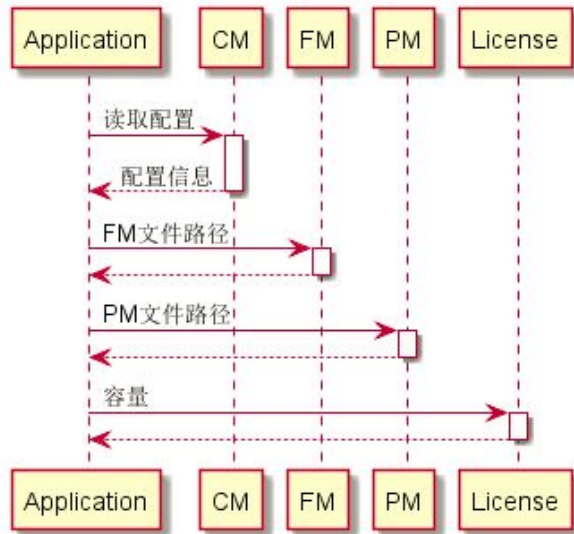
二、需求分析

1. 系统用例图

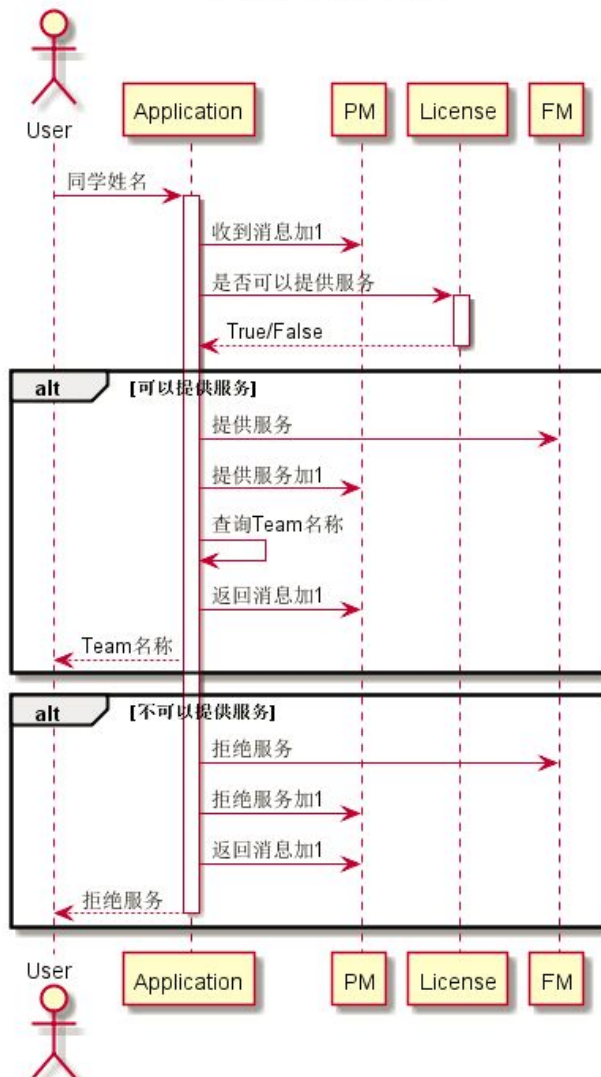


2. 系统时序图

Sequence Diagram: 启动



Sequence Diagram: 查询



三、架构与功能

该项目分为 Server 和 Client 两部分。

1. Server 服务端

Server 服务端主要提供对 Client 所传来的姓名所在组的查询功能。

Server 在启动时,经由 CM 组件从 Server 内部读取 project.properties 文件,按照以下格式得到 PM 和 FM 组件的输出路径,及学生信息的读入路径。并做相关初始化工作。

```
FM=src/file/FM_message.file  
PM=src/file/PM_message.file  
DataSource=src/file/name_group.txt
```

学生信息以如下格式给出在 txt 文件中。

```
吴逸菲=1  
黄徐欢=1  
李亚斯=1  
许铭湊=1  
王笑盈=2  
孙琳=2  
许舰=2  
李伟=2  
关晨=2  
胡圣托=3  
邓冰菲=2
```

此后 Server 借由 ActiveMQ 消息中间件,开启一个公有的消息队列。当有任何一个 Client 经由该公有消息队列向 Server 提出服务请求,会传入一个提出请求的当前时间。Server 收到请求后会首先会根据 License 的预设值判断是否能够提供服务。如不能,会返回拒绝服务的信息;如果可以服务,会以 Client 传入的时间作为唯一标示符另开启两个消息队列,分别作为 Server 对该 Client 的私有的传入和传出消息队列,并开启单独的线程对其进行服务。

此后,Server 每收到已经建立起私有消息队列的 Client 发来的姓名查询请求,会先根据 License 的预设值判断是否能够提供查询服务。如不能,会返回拒绝服务的信息;如果可

以服务，则会返回改姓名所在的组号，如果查无此人，则返回 Not Found 信息。

当 Server 收到来自某个 Client 的结束命令时，会关闭对该 Client 服务的线程，结束服务。

2. Client 客户端

Client 客户端在开启时会借由 ActiveMQ 消息中间件，通过 Server 的公有消息通道向 Server 传入当前时间，请求服务。如果收到 OK 的确认消息，即成功开启客户端，接收用户输入的姓名，通过 Server 建立的私有消息队列传给 Server 进行查询，并接收返回的结果。如果收到拒绝建立服务的消息或者在查询过程中收到 License 已经达到上限拒绝服务的消息，即自动关闭 Client。

当用户需要结束服务时，输入 Q 命令结束，Client 返回结束服务信息给 Server，同时自动关闭 Client。

四、组件的选择和使用

1. PM 组件

我们选择了第八组所开发的 PM 组件，用于

1. 接收应用程序的性能指标(指标名称,指标数值)
2. 每分钟自动生成性能报告(对每指标求和)
3. 性能报告输出到单独的性能文件,文件名包括性能报告时间

2. License 组件

我们同样选择了第八组所开发的 License 组件，用于

1. 每收到一个请求，记数加 1
2. 根据已经收到的消息数量和预设的 License 数值，判断是否可以继续提供服务

3. apache-activemq-5.10.0 消息中间件

我们使用了 apache-activemq-5.10.0 消息中间件，依其所提供的 Producer - Consumer 消息传递模式，将 Server 作为 Consumer，将每一个 Client 作为 Producer，实现二者之间消息的互通并与其他 Client 之间的消息独立。

五、使用说明

1. 找到“应用程序开发”目录，内含 Client、Server、Apache-activemq 等共四个文件夹。

2. 在 Server 端，将 Server 目录下的 src 导入到 IntelliJ 的 Java 项目中，将 Apache-activemq 文件夹中的 zip 文件解压缩，将其中的 activemq-all-5.10.0.jar 包 Add Library 到项目中。在命令行中，运行 apache-activemq-5.10.0\bin\activemq-admin 文件并输入 start 命令（需要注意选择是 Windows 还是 MacOS 的操作系统）。成功开启 ActiveMQ 消息服务，运行项目，开启 Server。

3. 在 Client 端，将 Client 目录下的 src 导入 Eclipse 的 Java 项目中，同样将 activemq-all-5.10.0.jar 包 Add Library 到项目中。运行项目，启动 Client。在命令行窗口中输入学生姓名后回车，即可得到结果。若服务次数已经达到上限，则返回失败信息，自动关闭 Client。如需退出客户端，用户输入 Q 命令。

六、关于测试

我们使用 JUnit 对 Server 程序进行了测试，分别对 MyServer、MyProducer、MyConsumer 类，针对服务端的启动、服务端的关闭、消息的传递、服务端查询功能的具体实现进行了测试。