**手游项目通信  
版本号：Ver3.1**

# 版本说明：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **版本号** | **日期** | **內容** | **作者** | **审核人** |
| **Ver1.0** | 2015-12-29 | 初稿 | 邓世伟 |  |
| **Ver2.0** | 2015-12-31 | 增加代码说明及使用例子 | 邓世伟 |  |
| **Ver3.0** | 2016-01-05 | 增加UML类图关系 | 邓世伟 |  |
| **Ver3.1** | 2016-01-08 | 增加系列图 | 邓世伟 |  |
|  |  |  |  |  |

# 请求通信

## 队列存储

1. 队列缓存各种用户操作，按操作的先后顺序来缓存。
2. 和后端约定相关参数及接口名称，根据约定来封装参数及接口。

## 数据打包

1. 把1个及以上的数据请求进行封装打包，需要和后端协商约定。
2. 数据使用SHA1或者MD5进行加密封装，需要和后端协商约定。
3. 每条数据都必须有唯一标识，发送及接收数据的时候会使用到。

## 发送数据

1. 检查当前是否有数据正在发送，根据状态来确定相关处理。
   1. 如果有数据在发送中，停止发送。
   2. 如果没有数据发送中，发送请求。
2. 发送一组数据后，启动定时器。用来检验请求来回时间。

## 定时器

1. 定时器一般5秒一个轮回，发送数据时，会被启动，收到请求回调后停止。
2. 当在设定时间内没有得到请求回调，启动重发机制，重发上一次请求，计数状态+1。
3. 计数设定最大上线，一般默认为3次，如果超出最大次数，抛出消息通知及提示警告，停止定时器。

## 重发

1. 在规定定时器时间内，重发一次当前没收到请求的包。
2. 在有效次数内，重发当前请求包。
3. 在提示警告后，玩家自动选择重发。

# 接收请求回调

## 解析数据

1. 根据返回的数据解析及分类。
2. 判断分析数据结果。
   1. 系统级错误，抛出消息通知，提示玩家。
   2. 普通逻辑错误，抛出消息通知，修改本地数据及更新显示。
3. 数据处理完成，抛出消息通知。

## 消息通知

1. 采用委托机制，通知处理结果。根据不同结果触发不同的委托。
2. 委托通知请求通信迭代。

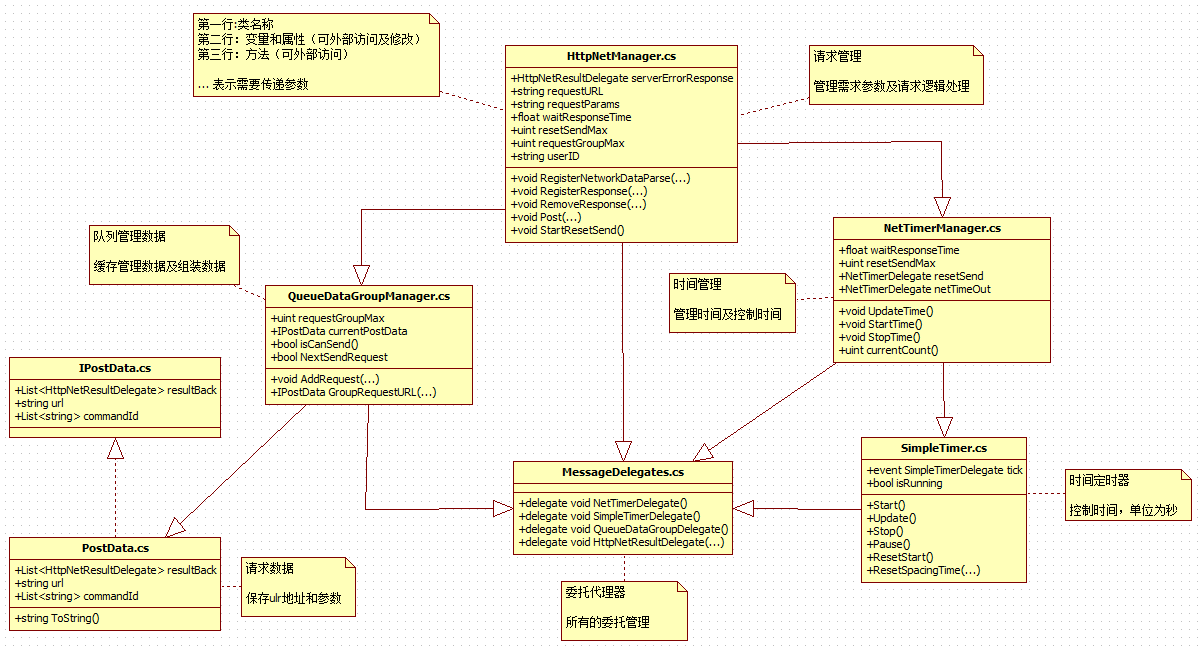
## 通信迭代

1. 触发通知，进入下一轮请求通信。
2. 回到起始请求通信节点，开始处理相关事务。

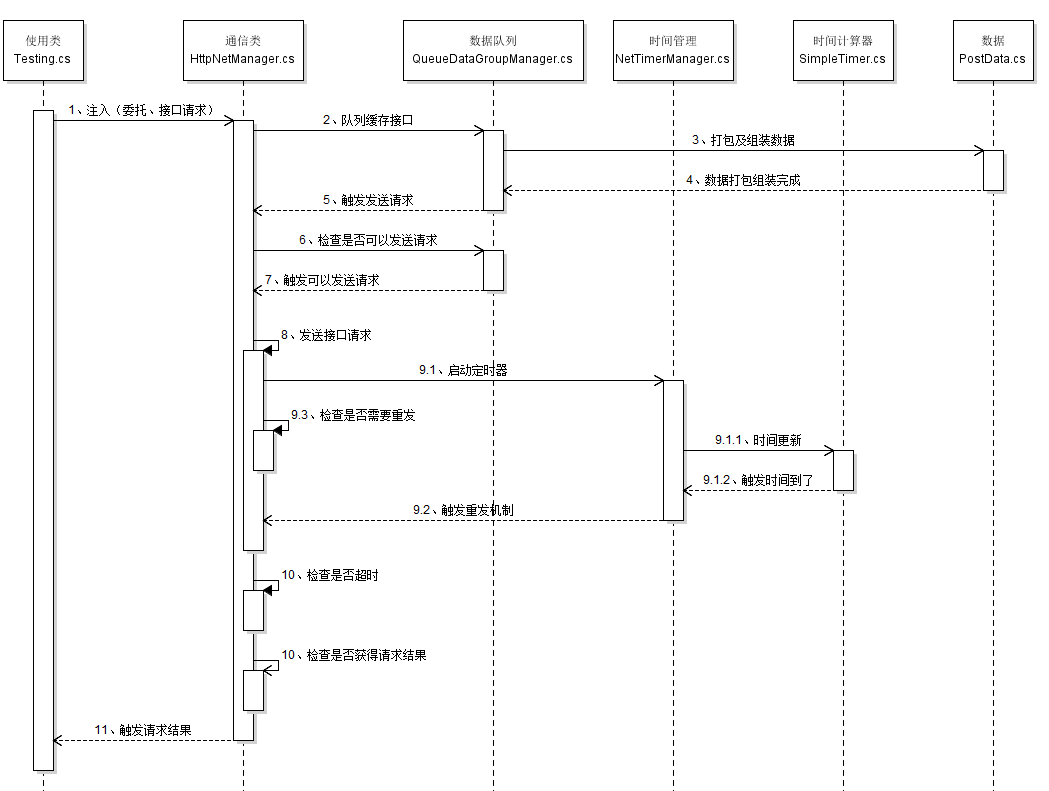
# 数据处理

1. 根据返回的数据进行Json转换。
2. 根据数据结构分类，缓存相应数据。
3. 管理及处理服务器错误提示码。
4. 缓存返回的数据，根据数据库表的格式缓存。
5. 对象字段通过反射类进行赋值。
6. 结果处理完成消息通知。

# UML类图关系



# 序列图



# 代码设计

## 通信类 (HttpNetManager.cs)

接收发送通信请求，处理实例化队列类及定时器类。管理通信地址及请求回调委托。处理服务器异常。

1. 接收通信请求。
2. 发起通信请求。
3. 接收通信返回结果。
4. 本次请求成功及失败的消息通知。

## 队列及组装数据类 (QueueDataGroupManager.cs)

缓存请求接口参数和消息委托，组装后端需要的接口及参数。

1. 根据后端需求组装接口及参数。
2. 队列缓存请求数据。
3. 检测是否有数据请求。
4. 缓存当前正在请求的接口及数据。
5. 以下为可访问及修改的公开属性及方法：

## 通信定时器类 (NetTimerManager.cs)

控制时间运转，管理重发次数及等待时长。

1. 计时器运转。
2. 重发计数处理。
3. 超出上限，抛出消息通知。
4. 以下为可访问及修改的公开属性及方法：

## 息通知委托类 (MessageDelegates.cs)

管理类之间的消息传递委托。

1. 管理以上3个类之间的消息传递。
2. 通知其他介入通信的消息传递。

## 普通定时器类 (SimpleTimer.cs)

管理时间的运行，时间单位为秒。

## 数据类 (PostData.cs)

请求参数的数据结构。

# 使用例子

以下为代码例子：

