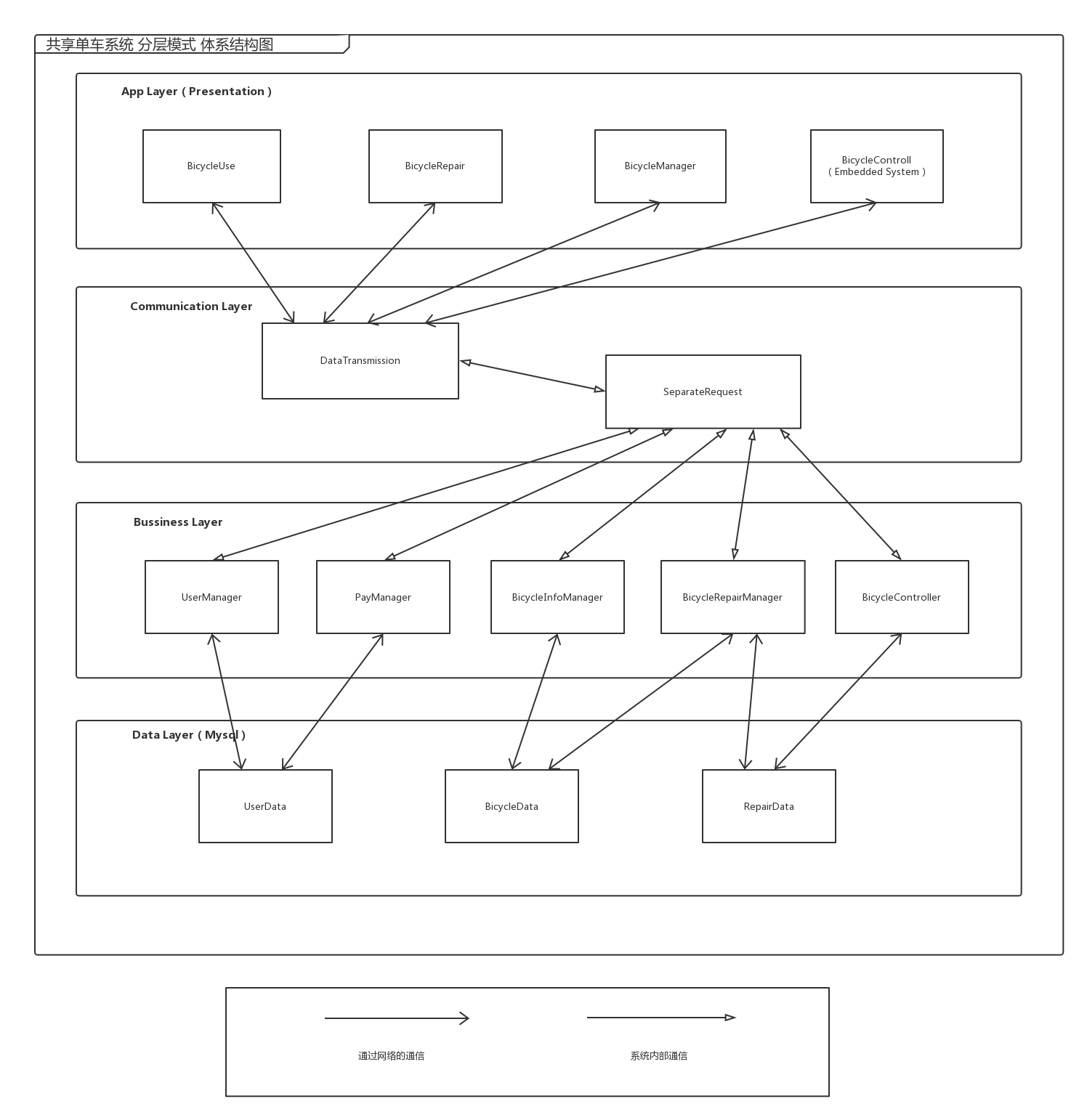
考虑到系统需求，我们在常用的架构选项中选择了可应用的一些架构模式，如layered模式、client-server模式、broker模式以及 P2P模式。

其中各个模式的选择理由如下：

|  |  |
| --- | --- |
| 架构模式 | 选择理由 |
| layered | 由于共享单车系统涉及不同平台交互，网络通信，数据存储处理等事物，是一个比较复杂的项目，使用分层结构，首先可以分离以上各个关键技术的实现，降低了开发的复杂度，其次将界面逻辑数据分离并增加专门的通信层可以增加系统的安全性，至于性能问题，由于系统分层不是太多，对系统的效率影响不是太大，应该可以通过使用某些新技术来改进。 |
| client-server | 由于共享单车系统需要远程传送数据，可以应用客户-服务器模式，各客户端通过网络向服务器端发送请求，服务器端处理请求并将处理结果返回给客户端。由于系统要处理高并发性请求，可以采用分布式数据库。同时，在客户端/服务器端开发过程中，也可以应用分层模式来降低复杂度。 |
| broker | 在传统C/S模式下进行完善，可直接使用Server块与下层的数据交互，Client-Server以及Client-Client之间的通信可以用Broker来定位，增加了系统的可扩展性。 |
| P2P | 在传统C/S模式下进行完善，考虑到系统具有高并发性的要求，采用P2P模式可以减少服务器负担。 |

各个模式的架构图及说明如下：

**layered：**



**各层描述**

App Layer（Presentation）：运行在手机上的app，或者是单车上的控制系统；具有和Communication Layer层通信的功能。

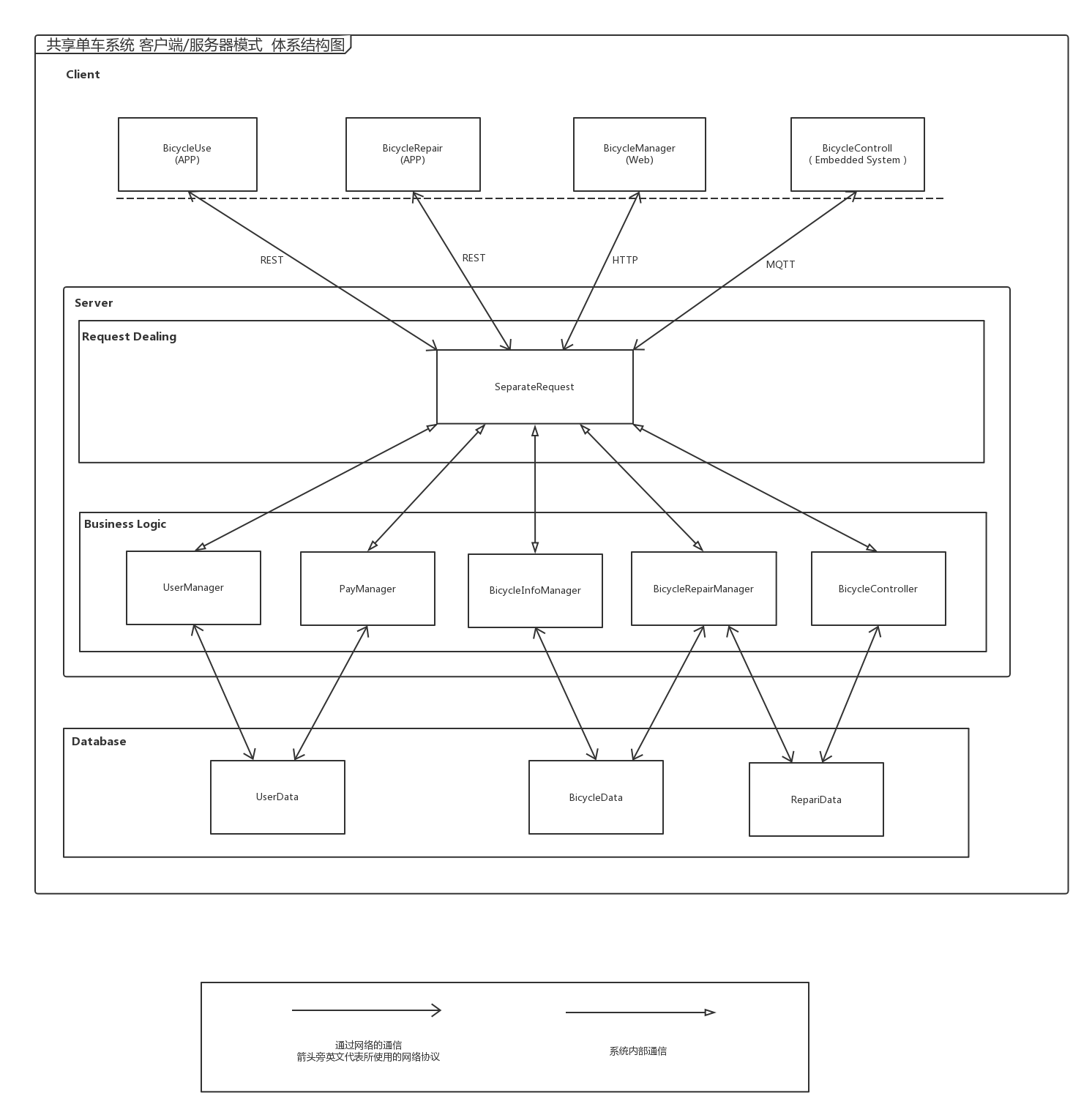
Communication Layer：负责解析来自App Layer不同角色的请求并将请求发送到Business Layer处理，或者将来自Business Layer的信息发送到App Layer的不同角色，也就是起到Business Layer和App Layer层的通信作用。

Business Layer：处理来自Communication Layer的信息，并返回处理结果，其数据来自底Data Layer。

Data Layer：负责使用持久性的技术存储数据的层，向Business Layer提供有关数据的各种操作，内部使用数据库如Mysql。

其中 Communication Layer 和 Business Layer 一般位于同一机器或平台。

**client/server:**

**各部分描述：**

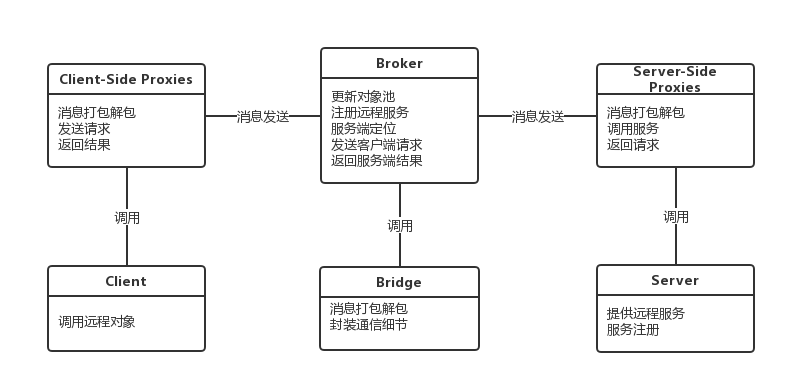
Client：用户端。用户端包括手机app（android、ios系统均有），web端和单车上的嵌入式系统。不同终端通过不同的网络协议同服务器端进行通讯。

Server：应用服务器端。应用服务器端主要分为两层：(1)RequestDealing负责将来自客户端的请求分配到Business Logic中不同部分处理，或者将Bussiness Logic返回的信息发送回客户端。(2)Business Logic：接受RequestDealing传来的客户端请求，从数据库中取得数据进行处理，并将处理结果返回给RequestDealing。

Database：数据库服务器端，负责和应用服务器端交互，进行数据的访问和修改。内部采用数据库，如MySQL。

本项目中，由于考虑到高并发性和高负载性，因此应用服务器与数据库服务器部署在不同机器中以提高性能。

**broker:**



在原有的C/S系统构造上，为了对共享单车系统远程对象服务组件进行解耦，封装不同系统应用间的通信，便于开发，进一步使用了Broker模式。该结构模式主要有六个组成部分：Clients，Servers，Client proxies，Server proxies，Broker，Bridge

Client：单车系统客户端通过Broker发送请求给服务端，通过Broker接受服务器端返回的结果和异常，具体结构见C/S模式Client块

Server：单车系统服务器端提供各种服务，包括单车信息分布，距离计算，用户查询等功能，具体结构见C/S模式Server块

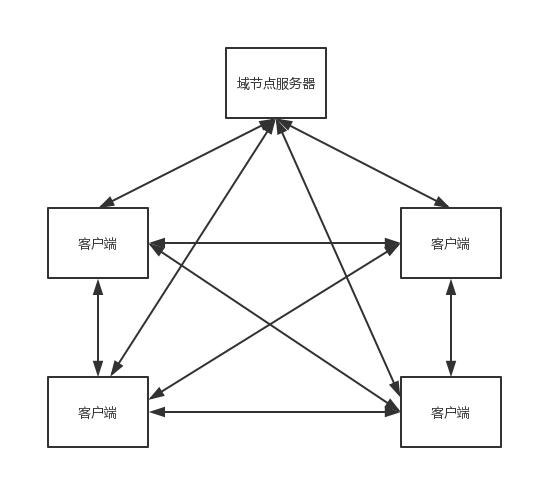
Broker：Broker在这里扮演的实际上是消息的传送者的角色。它把客户端的请求发送给服务端，服务端返回的结果和异常发送给客户端。由于Broker使得服务端对所有应用单车系统的用户提供本地的透明性，Broker能够唯一定位服务端标识。每一个客户端和服务端都有相应的一个Broker，每当请求发生时，总是单车系统客户端的Broker定位到服务端Broker的路由，然后再发送请求

Bridge：Bridge位于客户端和服务端的两个Broker之间，负责封装和隐藏两段之间的通信细节以及封装不同网络之间通信的细节

Client Proxies：负责封装Client及其相应的Broker之间消息的发送，提供服务对象本地透明性

Server Proxies：提供Server及其Broker之间的透明性

**P2P:**



在原有架构基础上，为减轻高峰时期对节点服务器的高并发和高负载要求，增加类似P2P模式下的客户端对等通信。

节点服务器->客户端:车辆的控制，客户端请求的反馈结果

客户端->节点服务器:申请，损坏维修，定位信息

客户端->客户端:当一个新的客户端申请查看周围单车（<5KM）时，已经被使用的单车由单车使用者app端提供当前单车的位置，车速，目的地等信息，而无需经过节点服务器的传输。

可用单车仍需要节点服务器提供位置信息。此共享信息可由当前客户端共享到其他查询用户。最大的共享传输范围在域节点服务器服务域内。