東南大學

软件设计与体系结构课

实验报告

题　目 **自动驾驶系统**

软 件 学 院 院（系） **软 件 工 程** 专业

学生 71119101 杨状凌 71119102 王博杰

71119103 许润 71119104 包亦成 71119138 王骏

指导教师 廖　力

起止日期 2021.12.01-2021.12.23

设计地点 **南京**

目录

0.课题概述

0.1 选题背景及意义

机动车的发展和普及，使得人们出行更加高效便捷。

但随着车辆行业的膨胀，以及私家车的平民化，车辆数量暴增，使得道路情况复杂度日益上升，交通形势也更加严峻。大到城市，道路负荷将会严重超载，使得交通系统损伤甚至瘫痪；小到车辆，错综复杂的行车情况，使得驾驶人精力消耗严重，使得可驾驶时间缩短，且容易导致疲劳驾驶，甚至酿成交通事故的悲剧。

所以，目前急需一个能够根据交通情况、路况智能调控的自动驾驶系统，它能够根据卫星定位把握交通信息，做出合理路径规划，预防恶性拥堵；也能够根据车周可捕获的道路信息，做出智能判断和预测，做出行驶行为决策，为驾驶人节省精力。

0.2 实验的主要工作

对自动驾驶系统的功能需求进行分析，明确本系统核心业务流程，定义关键质量属性和质量场景；选用两种架构风格对系统进行设计，分别是 C/S 架构和面向服务架构，包括每种架构风格对应的“4+1”视图、风格的选择决策、具体落实方案，并使用基于场景的架构评估过程，遵循标准步骤实施评估并记录。

0.3 实验报告的组织结构

本报告主要分为以下几个部分：

第一章是自动驾驶系统的需求分析，按照总分总结构介绍系统的需求。

第二章是自动驾驶系统的架构设计一，利用 C/S 架构进行设计和分析。

第三章是自动驾驶系统的架构设计二，利用 SOA 架构进行设计和分析。

第四章是架构对比，主要从特性上说明两种体系的分析结果。

第五章是架构评估，利用 ATAM 方法逐步分析。

第六章是开发计划，主要根据设计方案安排项目开发计划和管理方案。

第七章是报告总结，总结反思实验中的收获与心得，以及自己对架构的一些新的认识。

1.系统需求分析

1.1 总体需求概述

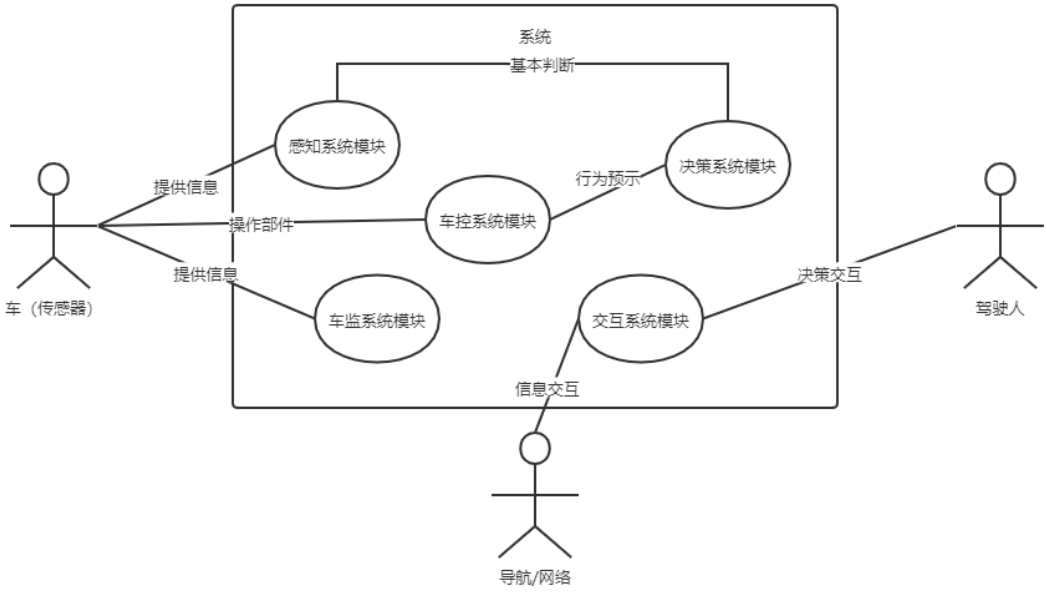
本课题的目的是开发一个自动驾驶系统。

自动驾驶系统主要通过对于周边环境以及车身情况，与外界达成实时信息交互，并进行接下来的行为决策，最后针对决策等信息对车辆进行方向、速度以及灯光等状态的调节，完成整个业务流程。

所以我们对于自动驾驶系统的划分，主要分为感知模块、决策模块、车控模块、车监模块和交互模块等五个模块，它们主要由相应的子系统模块组成：

- **感知系统模块**完成对车辆周围环境的感知识别和初步判断，自动驾驶使用了各种传感器，包括：摄像头、毫米波雷达、激光雷达、超声波雷达、GPS 等，每种感知技术相辅相成，使得车辆达到自主感知。
- **决策系统模块**主要是进行行为决策，包括对接下来可能发生情况的准确预测，以及下一步行动的判断和规划，选择合理路径等，会对其他模块发送行为预示。
- **车控系统模块**控制了车的油门、刹车、空调、灯光等，完成了对一般驾驶人会用到的各部件的操作，结合行为预示，将操作组合，完成变道超车等完整动作。
- **车监系统模块**主要针对于对车身各部件的监测，包括电路情况、燃料情况和发动机情况等，及时提醒驾驶人车身存在的问题，确保自动驾驶系统运行的正常条件。
- **交互系统模块**完成了自动驾驶系统与信息或卫星导航或他车系统交互的行为，主要是将自动驾驶系统形成网络互联，为服务提供更多的远程信息支撑；并且包含与驾驶人的交互，即某些信息需要驾驶人处理。

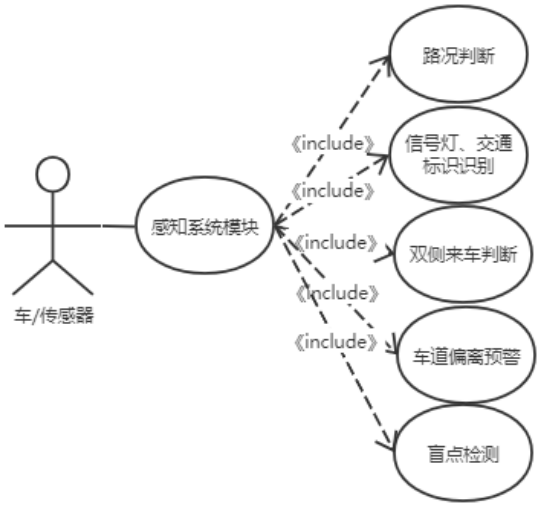
总体用例图如下：



1.2 感知系统模块

感知系统模块的功能主要是通过摄像头、雷达等各种传感器来对车辆周围环境的感知识别和初步判断，实现路口判断、标识识别、来车判断、偏离检测、盲点检测等功能。

具体用例图如下：



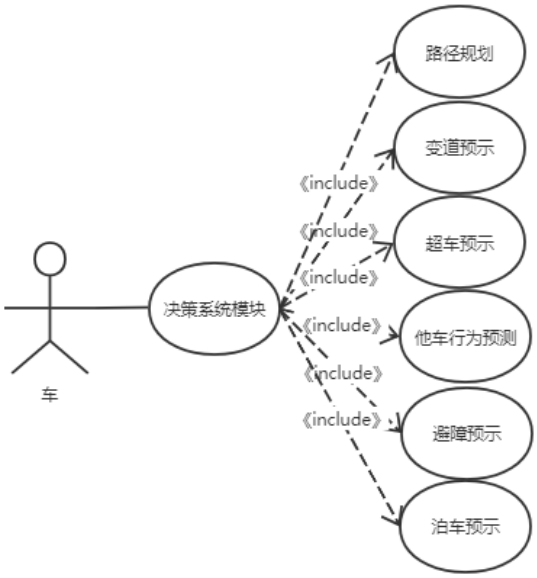
具体用例说明如下：

用例名称	外层包含模块	角色	用例功能
路况判断	感知系统模块	车/传感器	感知系统监控当前的道路是否存在施工、是否有车辆紧急停靠、雨天雾天的可见性、外界亮度等，来获取路况情况，来对车辆的行驶决策加以约束。
信号灯、交通标识识别	感知系统模块	车/传感器	感知系统监控道路两侧的交通标识和红绿指示灯的信息，来获取当前的交通状况。
双侧来车判断	感知系统模块	车/传感器	感知系统监控左后方、右后方的来车状况，判断是否有车辆从后方接近本车。
车道偏离预警	感知系统模块	车/传感器	机动车将根据导航系统精准定位，结合对路宽的检测，判断车道宽度，对可能存在的并道行驶情况报警。
盲点检测	感知系统模块	车/传感器	系统将在驾驶人一般视野盲点区域，根据传感器进行事物检测，并实时报警异常。

1.3 决策系统模块

决策系统模块的功能是进行行为决策，包括对情况的准确预测以及下一步行动的判断和规划，例如路径规划、变道预示、超车预示、他车行为预测、避障预示等等。

具体用例图如下：

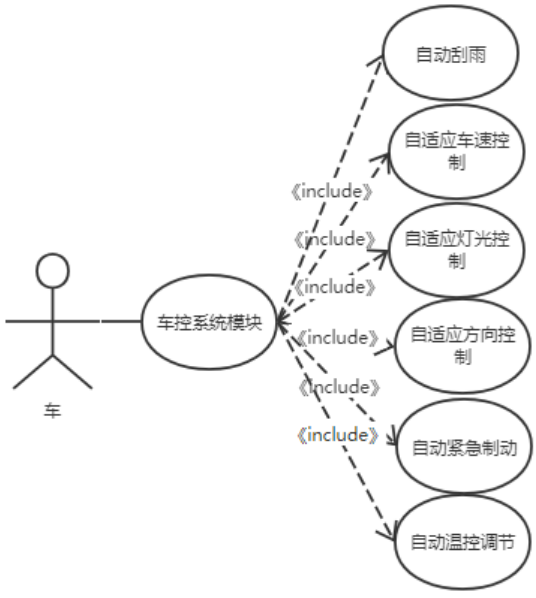


具体用例说明如下：

用例名称	外层包含模块	角色	用例功能
路径规划	决策系统模块	车	决策系统根据感知系统的一系列参数以及目的地，进行较为合理的路径规划。
变道预示	决策系统模块	车	决策系统根据当前的车辆状况和本车的行驶规划（直行或转弯）来进行变道操作。
超车预示	决策系统模块	车	决策系统根据当前的车辆状况，并依据驶往目的地的紧急程度，判断是否要进行超车操作来达到更快的行驶效果。
他车行为预测	决策系统模块	车	系统将判断前方车辆的图像角度变化以及灯光变化，预测其之后的驾驶行为，比如转弯或刹车等。
避障预示	决策系统模块	车	机动车检测到前方存在的障碍物，并同时检测其左右方路面情况，进行避障预示。
泊车预示	决策系统模块	车	系统根据地面所画的停车位进行自动泊车的预示。

1.4 车控系统模块

车控系统模块的主要功能是控制车的油门、刹车、空调、灯光等，完成了对一般驾驶人会用到的各部件的操作，包括自动刮雨、自适应车速控制、自适应灯光控制、自适应方向设计、紧急制动、自动温控调节等等。具体用例图如下：

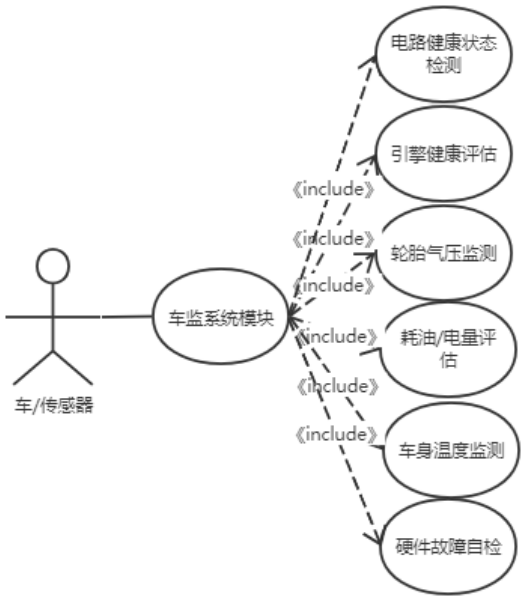


具体用例说明如下：

用例名称	外层包含模块	角色	用例功能
自适应车速控制	车控系统模块	车	车控系统根据道路的状况、交通标识等信息，控制本车的车速。
自适应灯光控制	车控系统模块	车	车控系统根据外界亮度、可见性等信息，进行车载灯光的自适应控制。
自适应方向控制	车控系统模块	车	车控系统将根据决策调节当前行驶方向。
自动紧急制动	车控系统模块	车	机动车对前方的人和车保持判断，并对其进行距离检测，如果距离过近，将采用紧急制动。
自动温控调节	车控系统模块	车	根据车舱内温度，调节适合档位的空调风速。
自动刮雨	车控系统模块	车	根据车外雨水情况，自动开启雨刮器，为驾驶人提供视野。

1.5 车监系统模块

车监系统模块的主要功能是完成对车身各部件的监测，及时提醒驾驶人车身存在的问题，确保自动驾驶系统运行的正常条件。包括电路健康状态检测、引擎健康评估、轮胎气压检测、车身温度检测等等。具体用例图如下：

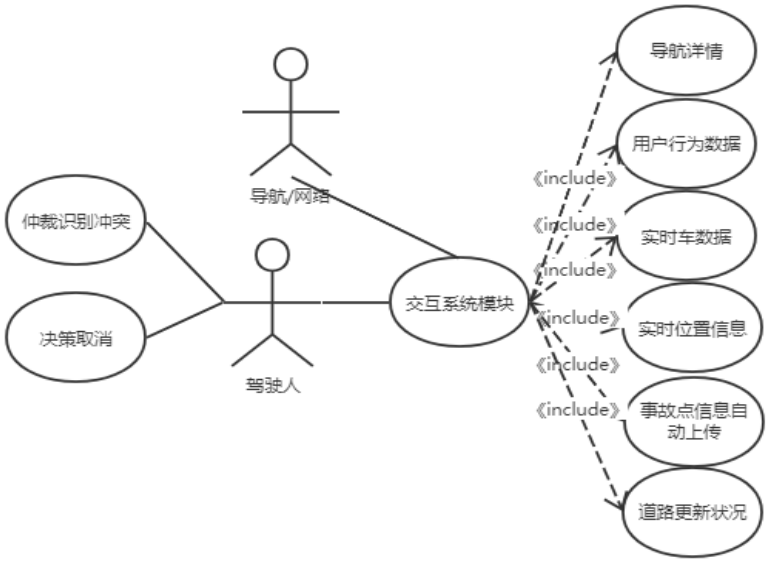


具体用例说明如下：

用例名称	外层包含模块	角色	用例功能
电路健康状态监测	车监系统模块	车/传感器	针对于机动车的电路系统进行故障监测和电力监测。
引擎健康评估	车监系统模块	车/传感器	针对于机动车的引擎转速以及温度进行定时测评。
轮胎气压监测	车监系统模块	车/传感器	针对于机动车的四个轮胎进行气压变化敏感性监测。
耗油/电量评估	车监系统模块	车/传感器	根据实时监测的机动车剩余能源，估算之后可顺利驾驶的最低路程公里数。
车身温度监测	车监系统模块	车/传感器	实时监测机动车引擎、车舱和油箱等部位的温度值，报警异常温度变化曲线。
硬件故障自检	车监系统模块	车/传感器	实时判断传感器、雷达和激光等设备电路状态是否正常，判断设备是否正在运行。

1.6 交互系统模块

交互系统模块的主要功能是将自动驾驶系统形成网络互联，为服务提供更多的远程信息支撑；并且包含与驾驶人的交互，即某些信息需要驾驶人处理。主要包括导航详情、用户行为数据、实时车数据、实时位置信息、事故点信息自动上传等等功能。具体用例图如下：



具体用例说明如下：

用例名称	外层包含模块	角色	用例功能
导航详情	交互系统模块	驾驶人、导航/网络	车辆实时从云端获取最新的导航数据。
用户行为数据	交互系统模块	驾驶人、导航/网络	搭载自动驾驶功能的车辆对当前处于人为驾驶的情况下的一系列行为操作进行记录，并存储至云端进行大数据分析。
实时车数据	交互系统模块	驾驶人、导航/网络	车辆实时共享车辆的车况信息，来对车辆的行驶状况进行安全监控。
实时位置信息	交互系统模块	驾驶人、导航/网络	车辆实时共享 GPS 定位信息，来达到对于失踪（对于定位信息交互失败的车辆）的车辆的快速寻回。
事故点信息自动上传	交互系统模块	驾驶人、导航/网络	车辆在经过事故发生点时，将具体的坐标记录并发送至云端，提醒使用导航的其他车辆。
道路更新状况	交互系统模块	驾驶人、导航/网络	车辆在经过与导航中道路状况不符的路段时，将所得到的道路信息上传至云端，更新数据。
仲裁识别冲突	------	驾驶人	当系统的多个决策或判断出现冲突时，系统将警告驾驶人进行人工仲裁决策。
决策取消	------	驾驶人	驾驶人可以选择取消当前自动驾驶系统的自动决策和判断；可选择屏蔽决策和判断，转为手动驾驶。

1.7 本章小结

本章节主要介绍了自动驾驶系统的需求，主要将其分为了感知、决策、车控、车监以及交互等五个模块，通过绘制用例图表现各模块间的关系，然后分别对各用例进行了详细的列表说明。

2.系统设计方案一

2.1 架构选型

C/S 架构实现了客户机与服务器的分离，使得组件分别运行在不同的计算机上，采用一种分布式的数据处理方式，再考虑到自动驾驶系统本身就需要在车辆上安装较多传感器、雷达等设备，因此采用 C/S 架构可以将大规模的业务逻辑以及数据处理分布到多个计算机中，从而降低整体的系统开销。

本系统采用三层 C/S 架构，将原本两层结构中客户机的业务逻辑层抽离出来作为单独的一层，客户端只保留表示层，各层之间通过约定好的接口相互调用。采用这种三层的 C/S 架构，优势为：高内聚低耦合、标准定义、逻辑复用、分散关注。高内聚低耦合降低了层与层之间的依赖性，提高了复用性。同时，明确了开发人员的分工，提高了软件项目的开发速度。但同时也必须考虑各层之间的通信效率，即使每一层的配置都很高，但通信效率的低下会降低系统的整体性能。

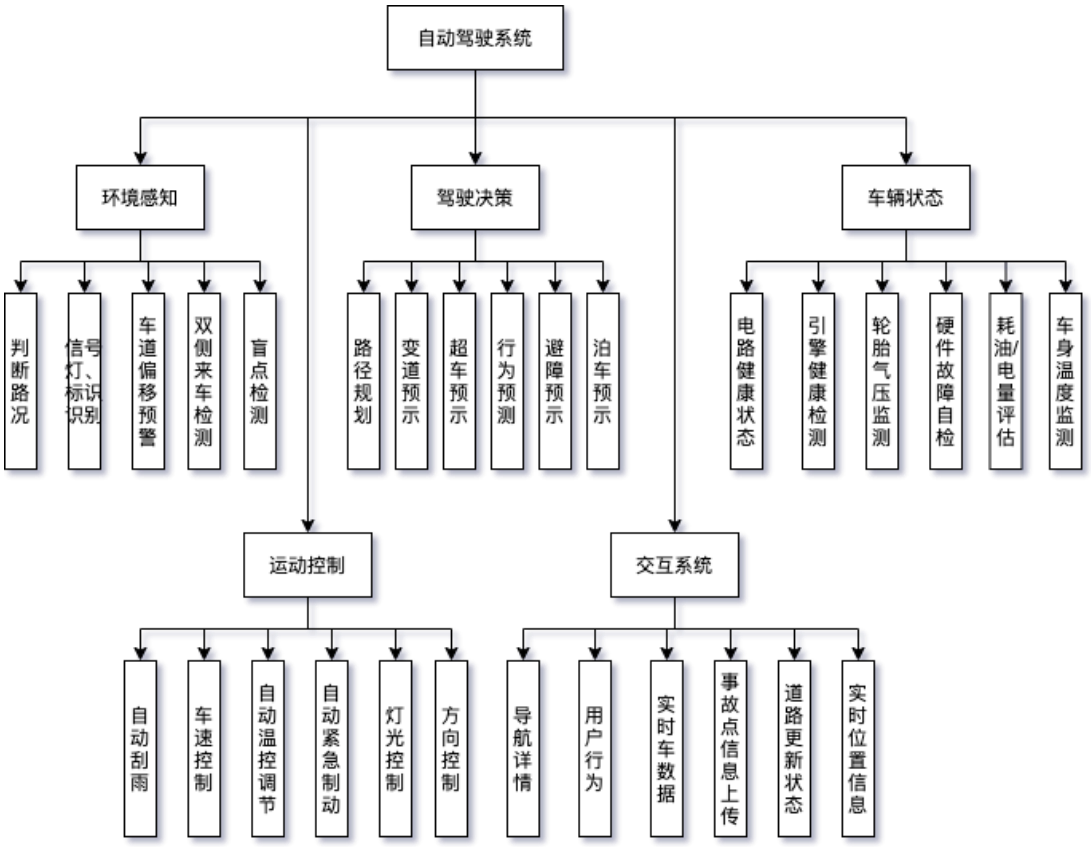
2.2 系统的功能划分

本系统将功能划分为 5 个模块，分别为环境感知模块，驾驶决策模块，车辆状态模块，运动控制模块，交互系统模块。

- **环境感知模块**负责利用传感器来获取外界信息，判断路况，查看信号灯，进行车道的偏移检测，以及盲点检测
- **驾驶决策模块**主要负责规划最优路径，同时在行驶过程中进行变道，超车，避障，泊车的决策，同时合理判断其他车辆的行为
- **车辆状态模块**主要负责电路健康状况检查、引擎健康检查、轮胎气压监测、硬件

故障自检、耗油/电量评估、车身温度监测等功能，保障车辆安全

- **运动控制模块**主要是实现驾驶过程中的决策，可以执行自动刮雨，车速控制，温度调节，灯光控制，方向控制；若遇到紧急状况，会采取紧急制动措施
- **交互系统模块**主要负责实现与云端、服务器、驾驶人的交互，进行导航详情、用户行为、实时车数据、事故点信息上传、道路更新状态、实时位置信息的管理

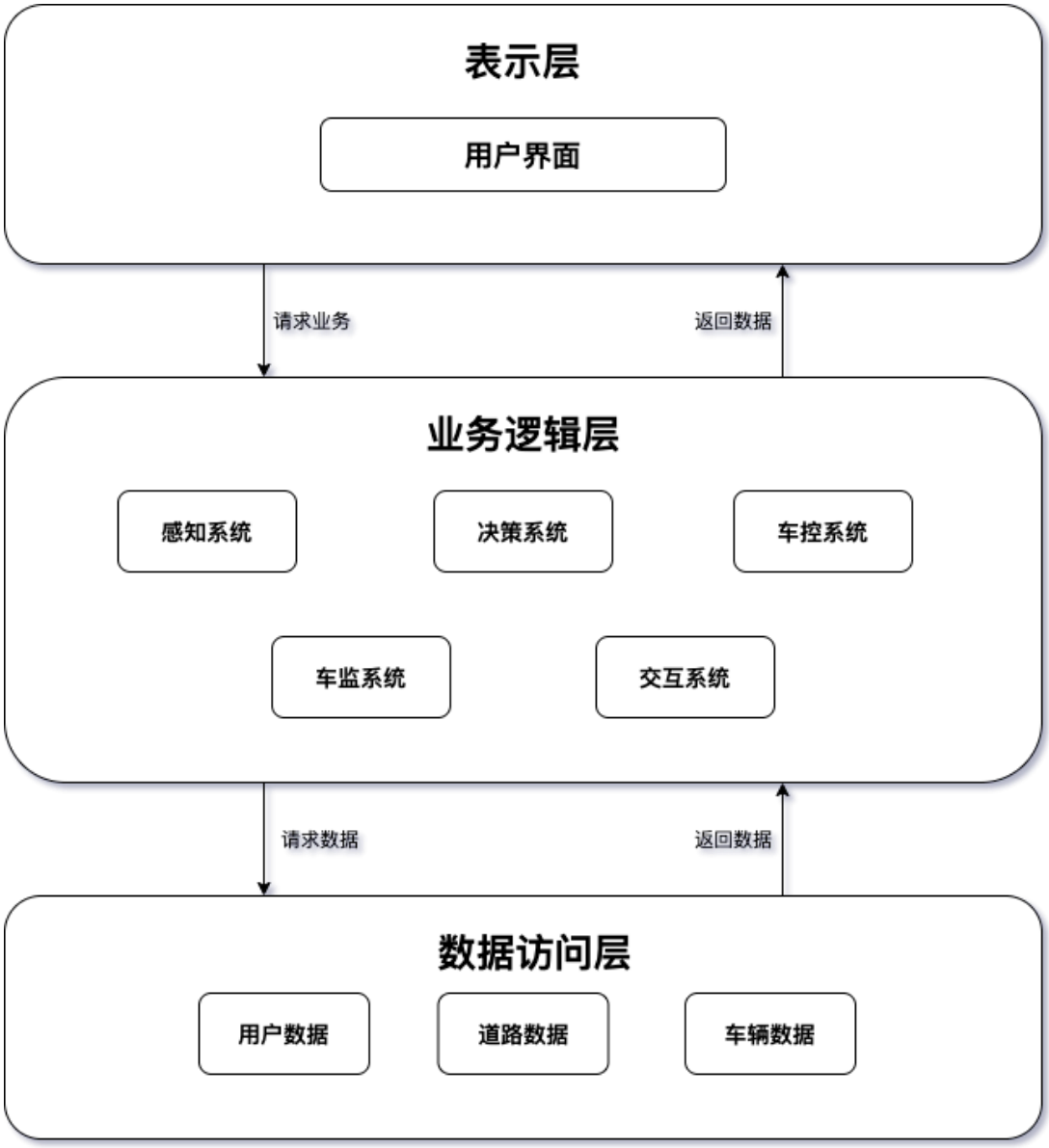


2.3 体系结构设计

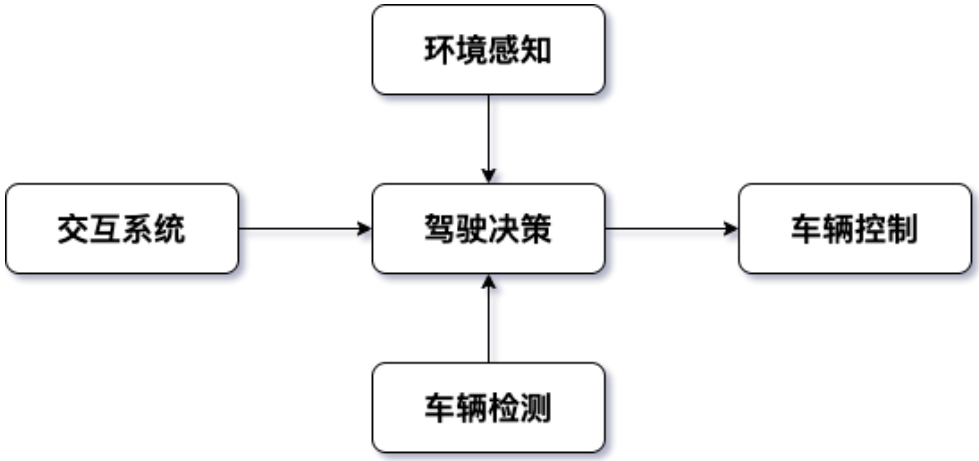
- 架构图
  - 本方案采用三层 C/S 架构，分为表示层、业务逻辑层、数据访问层
  - 表示层主要完成用户与应用程序之间的交互，包括接受用户输入的数据，显示应用的输出结果。在此层中，通常不包括系统的业务逻辑，逻辑代码仅与

界面元素有关。如地图导航、路况显示等。

- 业务逻辑层负责实现应用/业务的主要功能，当客户端需要进行数据访问和其他操作时，向业务逻辑层发送请求，由业务逻辑层向数据库发送请求完成数据查询等，并在客户端和服务端之间传输数据。包括感知系统、决策系统、车控系统、车监系统、交互系统。
- 数据访问层是系统的持久层，负责对数据库的读写及管理，主要是一些用户数据、道路数据和车辆数据。


表示层
用户界面
请求业务
返回数据
业务逻辑层
感知系统
决策系统
车控系统
车监系统
交互系统
请求数据
返回数据
数据访问层
用户数据
道路数据
车辆数据

- 逻辑视图


环境感知
交互系统
驾驶决策
车辆控制
车辆检测

2.3.1 组件

本系统由表示层、业务逻辑层、数据访问层组成的三层 C/S 架构体系实现。表示层主要负责反馈给用户视觉信息，让用户了解当前车辆状况和行驶情况；业务逻辑层主要负责从数据访问层中获取数据并加以处理，满足特定需求；数据访问层主要负责收集车辆各传感器、摄像头和云端的数据信息。

表示层

组件名	功能	数据请求	返回结果
MainWidget	显示一些车辆的基本信息和决策消息	请求各模块的处理后的车辆信息、决策信息和处理成功与否的结果	界面展示

业务逻辑层

组件名（业务类）	功能	数据请求	返回结果
InformationGet	路况判断 roadInformation	请求摄像头等感知器获取的道路的诸如图片、红外等数据	路面状况结果
	信号灯、交通标识识别 trafficInformation	摄像头捕捉到的关于交通标识和信号灯的图像	信号灯的状态和当前道路的交通提示信息
	双侧来车判断 comingCar Information	摄像头捕捉的本车左右两侧后方的图像和与左右两侧后方来车之间的超声波信息	左右两侧后方是否来车及来车时的距离信息
	车道偏离预警 offWayWarning	当前车辆位置和预计的车辆位置	车辆是否偏离
	盲点检测 checkBlind	摄像头捕捉车载人员视野盲点的头像	是否有异常状况
PlanDecision	路径规划 wayPlan	路况信息	所规划的路径
	变道预示 playChangeRoad	附近的车辆状况捕捉图和行驶规划	是否进行变道

	超车预示 playPassAhead	附近的车辆状况和预计的达到时间	是否进行超车
	他车行为预测 predictOthers	周围的车辆的一段时间的捕捉图片	他车的行为预测结果
	避障预示 playAvoidBar	车辆周围的图片捕捉结果	合理的避障流程
	泊车预示 playPark	地面的停车线捕捉信息与周围车辆信息	合理的泊车流程
CarController	自适应车速控制 autoSpeed	道路状况、交通信息	车速信息
	自适应灯光控制 autoLight	外界亮度、可见性	灯光亮度
	自适应方向控制 autoDirect	决策信息	方向控制结果
	自动紧急制动 autoStop	与前方的人和车的距离信息	是否紧急制动
	自动温控调节 autoTemperature	车舱温度情况和车载人员的状况	温度调节信息
	自动刮雨 autoClearRain	车外雨水状况	是否开启刮雨器
Monitor	电路健康状态监测 batteryMonitor	电路监测信息	电路状况
	引擎健康评估 engineAssess	引擎监测信息	引擎状况
	轮胎气压监测 wheelMonitor	轮胎气压监测信息	轮胎气压状况
	耗油/电量评估 energyAssess	耗油/电量的统计信息	耗油/电量
	车身温度监测 temperature Monitor	车辆关键部分的温度监测数据	温度状况
	硬件故障自检 hardwareChecking	硬件信息	硬件状况
Interaction	导航详情 navigation	获取最新导航数据	更新本地导航信息
	用户行为数据	捕捉人为驾驶时的操	驾驶员的行为数据

	driverAction	作记录影像	
	实时车数据 realtimeCarInfo	获取车辆当前的各监测器的监测数据	车辆状况
	实时位置信息 realtimePosition	获取 GPS 定位信息	向云端传输位置信息
	事故点信息自动上传 accidentUpdate	获取事故的定位信息	向云端传输事故位置信息
	道路更新状况 roadUpdate	获取道路的实际状况	向云端传输道路状况
	仲裁识别冲突 arbitration	冲突信息	提示仲裁信息
	决策取消 cancelPlan	决策信息	提示是否取消该决策

数据访问层

组件名	功能	数据请求	返回结果
CameraInformation	获取各个摄像头的捕捉图像	图片信息	经过灰度等处理后的图像信息
GPSInformation	获取定位信息	车辆坐标	车辆坐标
RoadInformation	获取到达目的地之间的各个道路的路况信息	路况信息	路况信息
CarInformation	获取车辆各部件的状况	车辆状况监测器的结果	车辆状况
DriverInformation	获取驾驶员的行为数据	行为数据	驾驶员的行为习惯
SensorInformation	获取诸如红外传感器之类信息	诸如红外、超声波等信息	数据分析结果

2.3.2 连接件

连接件	所承载的数据
Photo	摄像头捕捉的图像信息
GPSInfo	定位信息
RoadInfo	到达目的地之间的各个道路的路况信息

CarInfo	车辆各部件的状况
SensorInfo	诸如红外、亮度、超声波传感器之类信息
Decision	系统所作出的决策判断结果

2.3.3 约束

模块	操作	约束条件
感知模块	启用传感器	传感器自检通过且未被占用
	数据处理	数据流式处理，当处理程序空闲时可以使用
决策模块	路径规划	目的地在地图数据库中，且可以到达
	变道预示	前方车道空闲，后方 40 米没有车辆
	超车预示	前方有空闲车道，且前方车辆速度不大于 50km/h
	泊车预示	车辆后方有空闲车位，且空闲区域满足车辆大小的 1.3 倍
车控模块	自动刮雨	车速低于 60km/h，或车辆处于无人驾驶状态
	自适应方向控制	方向变更前后不影响周围车辆
	自动紧急制动	汽车有紧急制动需求，后方 60m 没有车辆
交互模块	决策取消	用户确认取消，且不构成驾驶威胁
	事故点信息自动上传	经过事故点，且其他车辆也标注此点为事故点
	用户行为数据上传	用户通过隐私策略，数据传输安全

2.4 体系结构实现方案

重要部分的流程图实例：

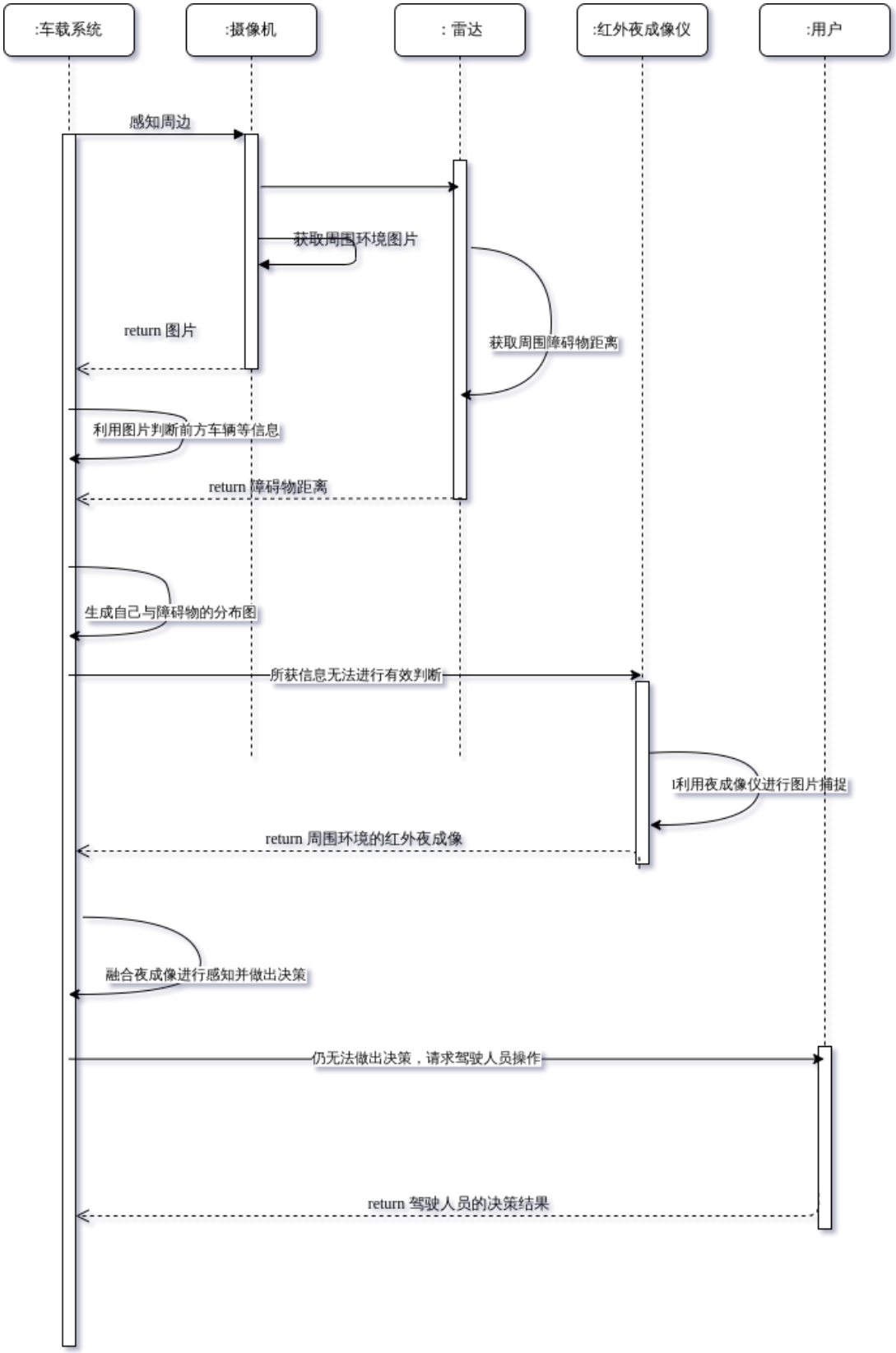
- 感知模块流程图

  a. 感知模块由车载系统最先对摄像机发出请求以获取周围环境图像，同时向雷达发出请求来定位周围的障碍物

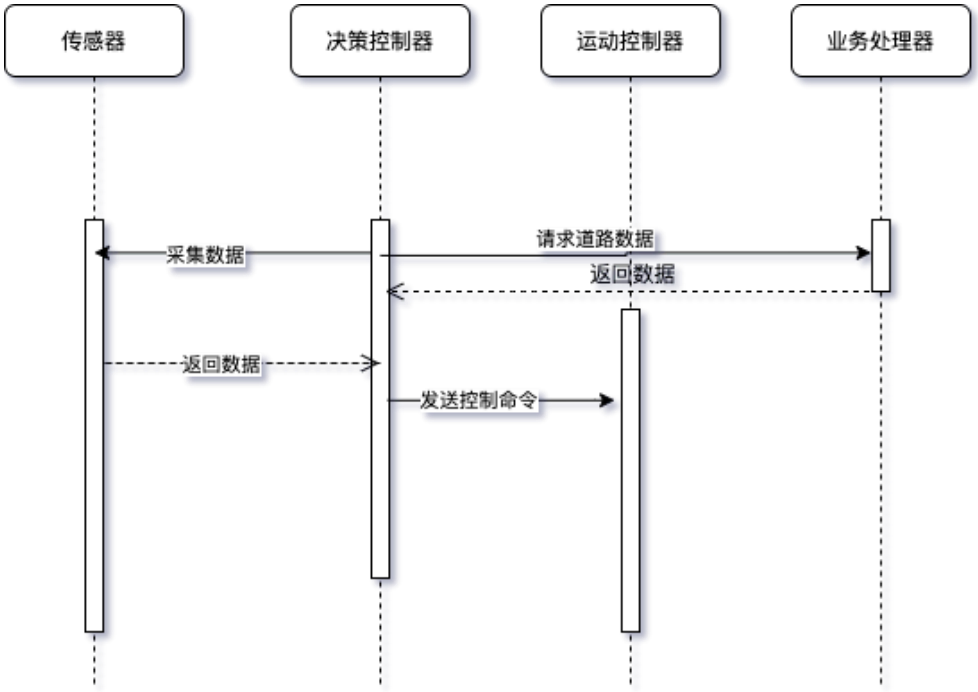
  b. 在周围光线，能见度都良好的状况下，摄像机返回的照片将满足车载系统对于周围环境的判断，再结合雷达返回的障碍物距离，推断出周围环境状态实现感知

  c. 若利用上述两个传感器无法对周围环境进行有效判断，车载系统会向更高级的传感器发送请求，以获取更多信息，再结合这些信息做推测

  d. 若所有传感器返回的数据都无法满足车载系统对于感知的需求，这时车载系统将对用户发出警告，让用户参与感知与决策

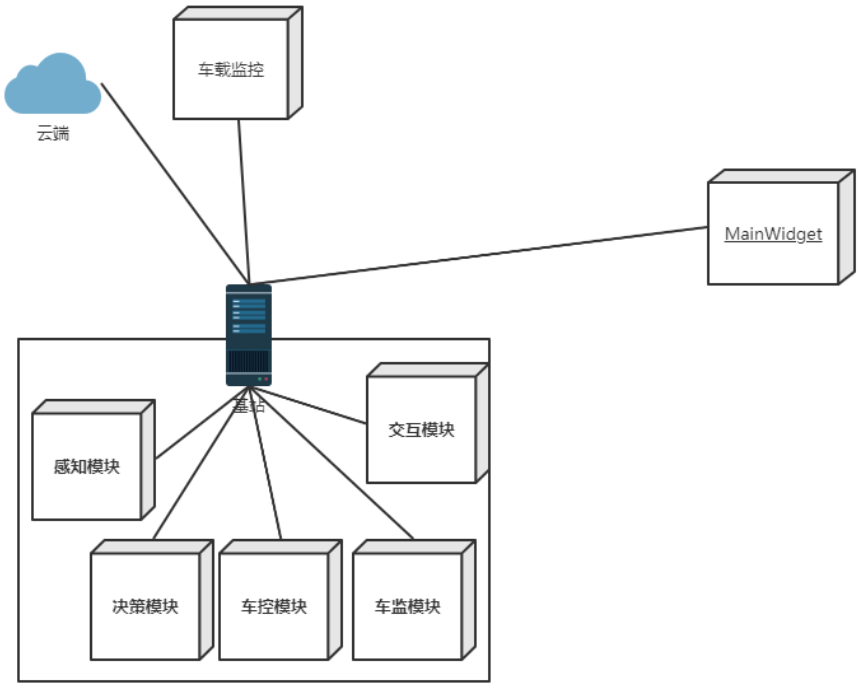

:车载系统
:摄像机
: 雷达
:红外夜成像仪
:用户
感知周边
获取周围环境图片
return 图片
获取周围障碍物距离
利用图片判断前方车辆等信息
return 障碍物距离
生成自己与障碍物的分布图
所获信息无法进行有效判断
利用夜成像仪进行图片捕捉
return 周围环境的红外夜成像
融合夜成像进行感知并做出决策
仍无法做出决策，请求驾驶人员操作
return 驾驶人员的决策结果

- 各模块数据交互流程图
  - 主要是由决策控制器起主导作用，实时读取传感器的数据，并通过计算得出当前对车辆控制的一系列决策，并实时将决策数据发送给车辆的各控制器，包括运动控制、灯光控制等；
  - 同时，决策控制器还需要在有需求的时候去请求业务处理层的数据，比如要更新地图信息、道路信息等。



2.5 体系结构部署

本次 C/S 架构实现的部署结构为：将展示结构部署在车载系统中，部分的数据访问模块也应当部署在车辆上（获取车辆数据），将大规模的业务实现模块部署在各个基站中，以获取较高的处理能力。



2.6 本章小结

C/S 架构风格实现的自动驾驶系统，将表示层和数据访问层部署在车辆上、业务逻辑层部署在分布在各地的服务基站上，通过网络高速传递数据，利用基站的强大的数据处理能力快速进行业务功能的实现，争取在低时延内规划出合理的决策，为自动驾驶保驾护航。

C/S 架构的重点在于业务逻辑的设立，合理的处理需求分析的结果如何转化至业务逻辑的设计成为了重中之重。目前看来，本次实验中将需求分析中的各模块对应实现为业务类是一种较为合理的方案。

3.软件设计方案二

3.1 系统的体系风格选择

SOA 作为一种面向服务的架构，是一种设计思想和方法论。在 SOA 架构中，服务是最核心的抽象手段和系统最基础的描述单元。每个服务组件具备独立的功能，且可被复用；服务组件之间的接口遵循统一标准，可互相访问，可组合扩展。

SOA 架构应用于汽车自动驾驶系统架构下有以下优势：1、车辆功能软件实现软硬分离，组件功能可以脱离硬件平台任意部署移动，对于算力要求的复杂功能组件可向具备高带宽算力的域控制器移动。2、车辆功能可被大范围复用，一些使用频度很高且基础的功能可作为基础服务组件先被开发，由于服务组件的独立性以及接口的标准可访问，基础服务开发完成后可向服务中间件注册，并供所有 SOA 架构中的控制器订阅使用。

因此我们在第二种设计方案中选择 SOA 风格。

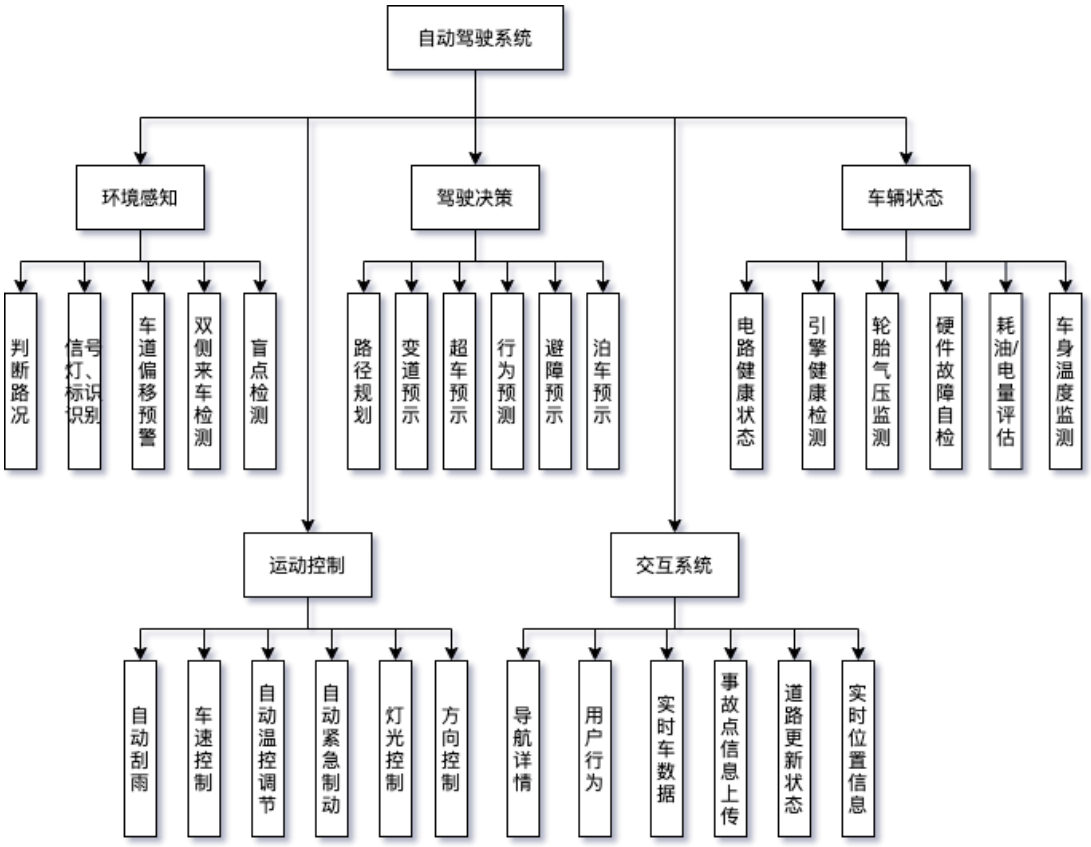
3.2 系统的功能划分

本系统将功能划分为 5 个模块，分别为环境感知模块，驾驶决策模块，车辆状态模块，运动控制模块，交互系统模块。

- **环境感知模块**负责利用传感器来获取外界信息，判断路况，查看信号灯，进行车道的偏移检测，以及盲点检测
- **驾驶决策模块**主要负责规划最优路径，同时在行驶过程中进行变道，超车，避障，泊车的决策，同时合理判断其他车辆的行为
- **车辆状态模块**主要负责电量健康状况、引擎健康检查、轮胎气压监测、硬件故障

自检、耗油/电量评估、车身温度监测的功能，保障车辆安全

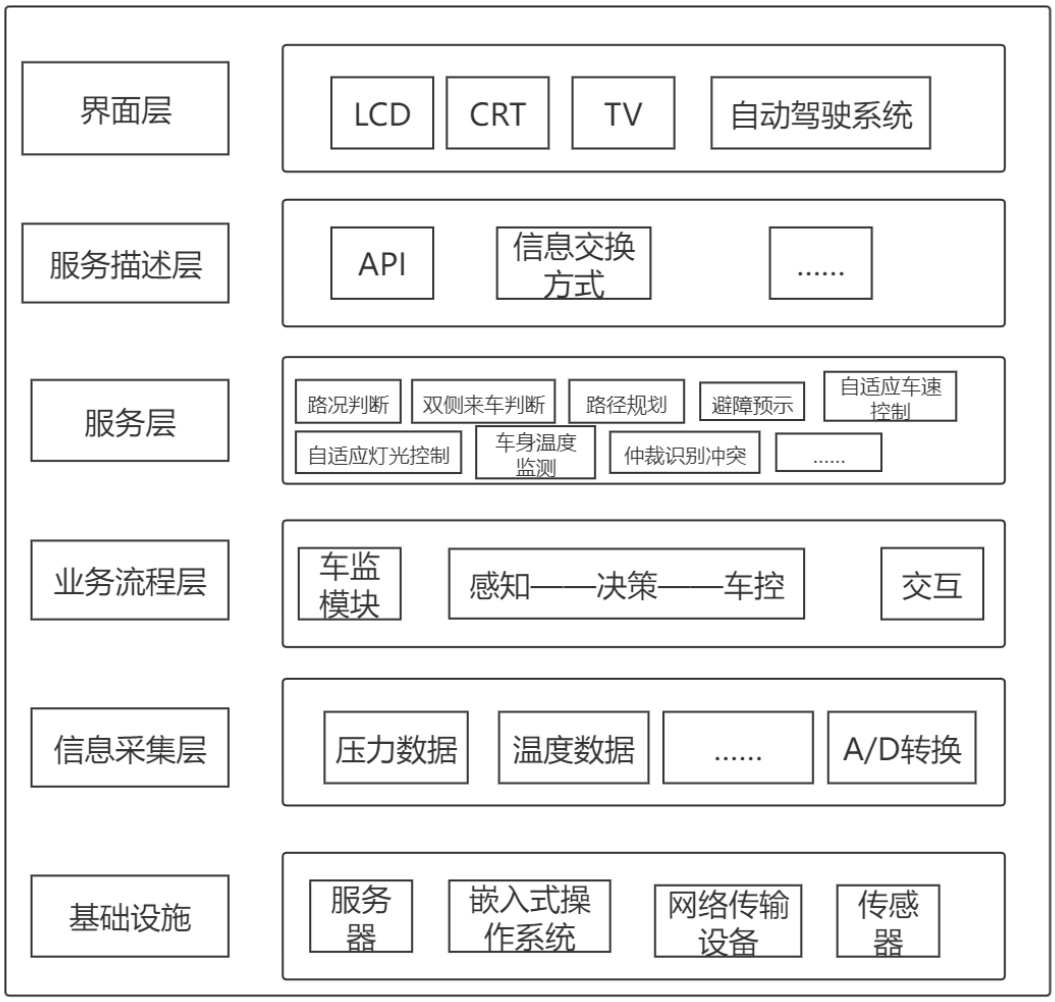
- **运动控制模块**主要是实现驾驶过程中的决策，可以执行自动刮雨，车速控制，温度调节，灯光控制，方向控制；若遇到紧急状况，会采取紧急制动措施
- **交互系统模块**主要负责实现与云端、服务器、驾驶人的交互，进行导航详情、用户行为、实时车数据、事故点信息上传、道路更新状态、实时位置信息的管理



3.3 体系结构设计

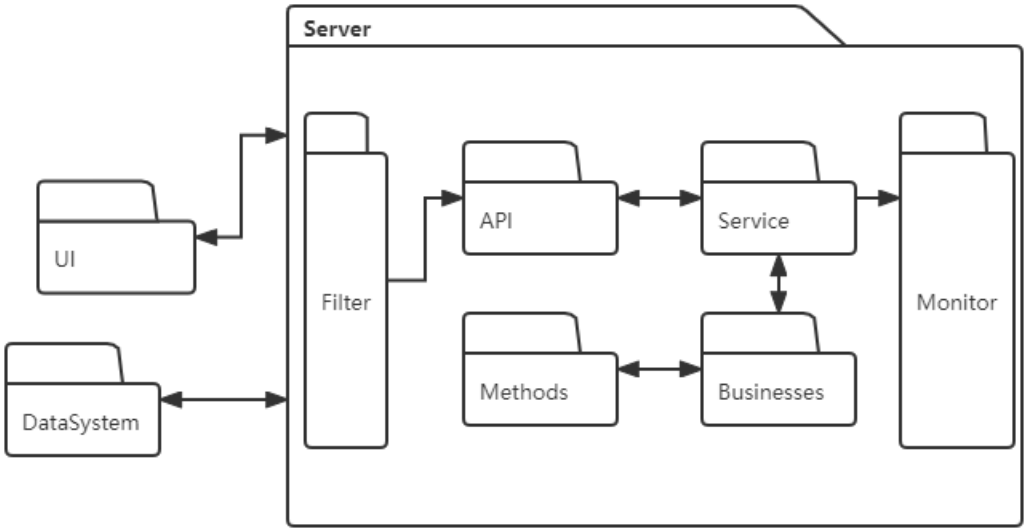
- **逻辑视图**

面向服务的架构除了界面层外，将接口以及信息交互归类为服务描述，且在面向服务的架构中，主体为各项服务，我们将服务与分析的用例类比，而将服务实现的串行流程归结为业务流程层，另外架构中也不免安排硬件设施对信息进行采集传输。



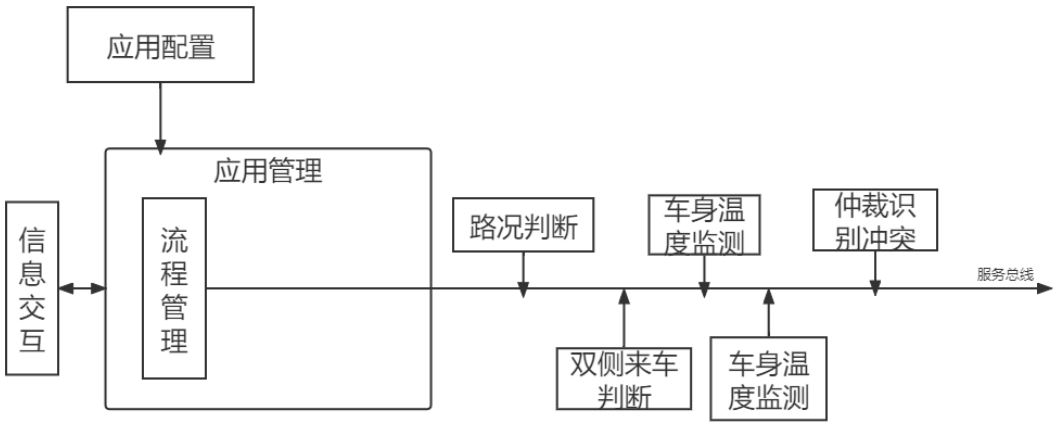
- **开发视图**

由于所有的开发都不免前端视图，此处我们便简单地用 ui 文件描述即可，途中数据系统主要负责对采集数据的预处理和打包等，而在服务器开发中，首先应对请求进行过滤，然后调用 API 实现服务的业务流程，所有操作中，数据都将受到监控。



- **流程示意**

本系统在面向服务的架构中，先对信息做出交互，然后根据服务的加入，在总线上显示各服务产生的数据，以及各个服务等待总线上的数据进行功能实行，整个流程在总线上得以完成。



3.3.0 服务划分

我们的系统中有许多偏底层功能都不需要自己实现，可以根据需要来通过服务接口进行实现，我们利用的服务主要分为云端服务和本地服务两种，对应于系统的云端"车联网"和对车辆本身的操作及信息获取。

编号	服务名	类别	说明
1	定位获取	云端服务	向导航系统服务器请求导航定位
2	驾驶人标准状态获取	云端服务	向云服务器请求正常驾驶人状态参数区间，以判断驾驶人状态是否正常
3	机动车标准状态获取	云端服务	向云服务器请求正常机动车部件参数区间，以判断机动车状态是否正常
4	数据上传	云端服务	将数据传送到指定数据库，作为分析基础
5	路径获取	云端服务	使用起始位置通过云服务获取最佳路径规划
6	能源剩余量获取	本地服务	向本地车载系统获取剩余油量或电量
7	温度获取	本地服务	向本地车载系统获取传感器检测的温度值
8	电路电压获取	本地服务	向本地车载系统获取传感器检测的关键电路电压值
9	轮胎气压获取	本地服务	向本地车载系统获取传感器检测的轮胎气压值
10	油门调控	本地服务	通过本地车载系统的接口，改变油门压起状态
11	速度获取	本地服务	向本地车载系统获取当前速度值
12	制动器调控	本地服务	通过本地车载系统改变制动器压起状态
13	灯光状态获取	本地服务	向本地车载系统获取当前灯光的开启关闭序列
14	灯光状态调控	本地服务	通过本地车载系统改变灯光系统的开闭状态
15	方向获取	本地服务	通过本地车载系统获取当前车舵旋转度数和罗盘检测的当前方向度数
16	方向调控	本地服务	通过本地车载系统改变车舵的旋转度数
17	空气循环调控	本地服务	通过本地车载系统切换内/外循环、制热/冷模式以及风速档位
18	温度获取	本地服务	向本地车载系统获取传感器检测的温度值
19	雨刮器调控	本地服务	通过本地车载系统切换雨刮器档位
20	车周距离检测获取	本地服务	通过本地车载系统获取遍布在车周身的距离传

			感器信息
21	交通标识识别	云端服务	实时上传图片至云端，获取标识识别结果
22	红绿灯指示识别	云端服务	实时上传图片至云端，获取红绿灯颜色识别结果
23	行为预测	云端服务	实时连续上传图片或视频至云端，获取其他人或车辆的行为预测结果

上表中，需要说明的是编号 21、22 以及 23 的服务，这些图像处理的服务，可以说是自动驾驶系统的核心，但我们如果使用云端服务进行的话，可能存在着传输速度和信道繁忙等限制，反而使得利用云端的便捷变为了困难。

另外，关于我们所定义的"决策服务"，其要求是获取感知信息，然后输出决策指令，这更偏向于是一个我们自主实现的"伪服务"，但也不能排除可以使用现成的服务接口，将通过本地服务获取的图片和传感器等信息上传至云端服务，然后获取决策指令，那么它也将同上成为一个可能将便捷变为困难的"真服务"。

3.3.1 组件

本系统包含 Sensor、Decision、Control、Monitoring、Interaction 五个组件 ，分别对应服务层的感知模块、决策模块、车控模块、车监控制、交互模块五个服务部分。

Sensor 组件：

类名	功能	数据请求	返回结果
roadSensor	感知路面是否在施工 getIfConstruction	摄像头采集当前道路图片	路面是否在施工
	感知路面是否有车辆紧急停靠	感应器感应是否有车辆停靠	路面是否有车辆停靠

	getIfParkedCar		
	感知路面可见性 getVisibility	传感器采集路面可见性	路面可见度
	感知路面亮度 getBrightness	光度传感器	路面亮度
trafficSensor	感知道路两侧交通标识 getSignal	摄像头采集的道路两侧的交通标识	交通标识类别
	感知道路前方红绿灯信息 getTraffic	摄像头采集的道路前方红绿灯信息	红绿灯颜色
comingcarSensor	感知双侧来车 getIfComingcar	传感器感应左右后方的来车状况	是否有车辆从后方接近本车
deviationSensor	感知车道偏离 getIfDeviation	获取导航系统精准定位以及当前路宽	是否偏离车道
pointSensor	感知盲点 getPoint	传感器检测驾驶人视野盲点区域	盲点是否安全

Decision 组件：

类名	功能	数据请求	返回结果
pathDecision	规划较为合理的路径 pathPlanning	感知系统的一系列参数以及目的地	行驶路径
changeLanesDecision	判断是否进行变道操作 ifChangeLanes	当前的车辆状况和本车行驶规划	是否进行变道操作
overtakeDecision	判断是否进行超车操作 ifOverTake	当前的车辆状况以及行驶的紧急程度	是否进行超车操作
otherCarsDecision	预测其他车辆的驾驶行为 drivingBehavior	其他车辆的图像变化以及灯光变化	其他车辆的驾驶行为
obstacleAvoidanceDecision	判断是否进行避障 ifObstacleAvoidance	传感器返回的路面情况	是否进行避障

parkingDecsion	判断是否需要泊车 ifParking	摄像头采集的停车位信息	是否需要泊车

Control 组件：

类名	功能	数据请求	返回结果
VelocityControl	调节车速 updateVelocity	决策指定的速度值	调节后的速度值
LightControl	调节灯光 updateLight	决策指定的灯光开关序列	调节后的灯光开关序列
DirecControl	调节方向 updateDirection	决策指定的方向角度	调节后的方向角度
BrakeControl	调节制动器 updateBrake	决策指定的速度值	调节后的速度值
AirControl	调节空调 updateAir	决策指定的空调档位	调节后的空调档位
WiperControl	调节雨刮器 updateWiper	决策指定的雨刮器档位	调节后的雨刮器档位

Monitoring 组件：

类名	功能	数据请求	返回结果
ComponentMonit	对机动车电路情况进行监测 checkCircuit	电路各点位的电压传感器参数	量化后的电路健康评估等级
	对机动车引擎情况进行检测 checkEngine	引擎周边的光电门转速测试参数	量化后的引擎健康评估等级
	对机动车轮胎情况进行检测 checkTyre	四个轮胎内壁的气压传感器参数	量化后的轮胎健康评估等级
SensorMonit	对机动车的硬件进行故障自检 checkSensor	所有传感器的参数	未工作的传感器名称及位置列表
TempeMonit	对机动车各部位进行	所有温度传感器的位	量化后的各部位信息

	温度监测 checkTempe	置和参数数据	和温度数据的列表
RemainMonit	对机动车剩余燃料可进行公里的估算 getRemainDis	剩余燃油量或电量	最低可行驶公里数

Interaction 组件：

类名	功能	数据请求	返回结果
NaviInter	从导航系统获取位置 getPosition	导航系统的实时位置	规格化后的浮点数位置坐标
	向导航更新道路情况 upRoad	导航通讯地址	是否成功的布尔值
CarInter	获取车辆组网内的车身正常信息 getStandard	车身部件参数范围	规格化后的二维数组，表示各个参数的标准范围
	上传车身信息到组网 upOwnMes	组网通讯地址	是否成功的布尔值
	上传实时位置信息 upOwnPosition	组网通讯地址	是否成功的布尔值
SystemInter	上传事故情况视频 upAccident	服务器通讯地址	是否成功的布尔值
	上传驾驶人行为数据 upDriver	服务器通讯地址	是否成功的布尔值
DriverInter	向驾驶人请求仲裁 getArbitration	系统界面	规格化后的驾驶人的仲裁选项
	监听驾驶人的决策取消请求 listenCancel	驾驶人的取消请求	是否成功的布尔值

3.3.2 连接件

连接件	功能说明	所承载的数据
Sensor2Decision	将感知的信息和初步判断传输给决策模块的服务	传感器和摄像头获取信息和初步识别信息
Monit2Decision	将车身监测信息传输给决策模块的服务	车辆各部件的监测传感器信息
Desicision2Control	将经决策处理后的信息传输给控制模块的服务	决策信息（速度值、灯光序列等）
Navi2System	将导航系统的信息传输给自动驾驶系统	车辆实时位置信息

3.3.3 约束（同方案一）

模块	操作	约束条件
感知模块	启用传感器	传感器自检通过且未被占用
	数据处理	数据流式处理，当处理程序空闲时可以使用
决策模块	路径规划	目的地在地图数据库中，且可以到达
	变道预示	前方车道空闲，后方 40 米没有车辆
	超车预示	前方有空闲车道，且前方车辆速度不大于 50km/h
	泊车预示	车辆后方有空闲车位，且空闲区域满足车辆大小的 1.3 倍
车控模块	自动刮雨	车速低于 60km/h，或车辆处于无人驾驶状态
	自适应方向控制	方向变更前后不影响周围车辆
	自动紧急制动	汽车有紧急制动需求，后方 60m 没有车辆

交互模块	决策取消	用户确认取消，且不构成驾驶威胁
	事故点信息自动上传	经过事故点，且其他车辆也标注此点为事故点
	用户行为数据上传	用户通过隐私策略，数据传输安全

3.4 体系结构实现方案

重要部分的流程图实例：

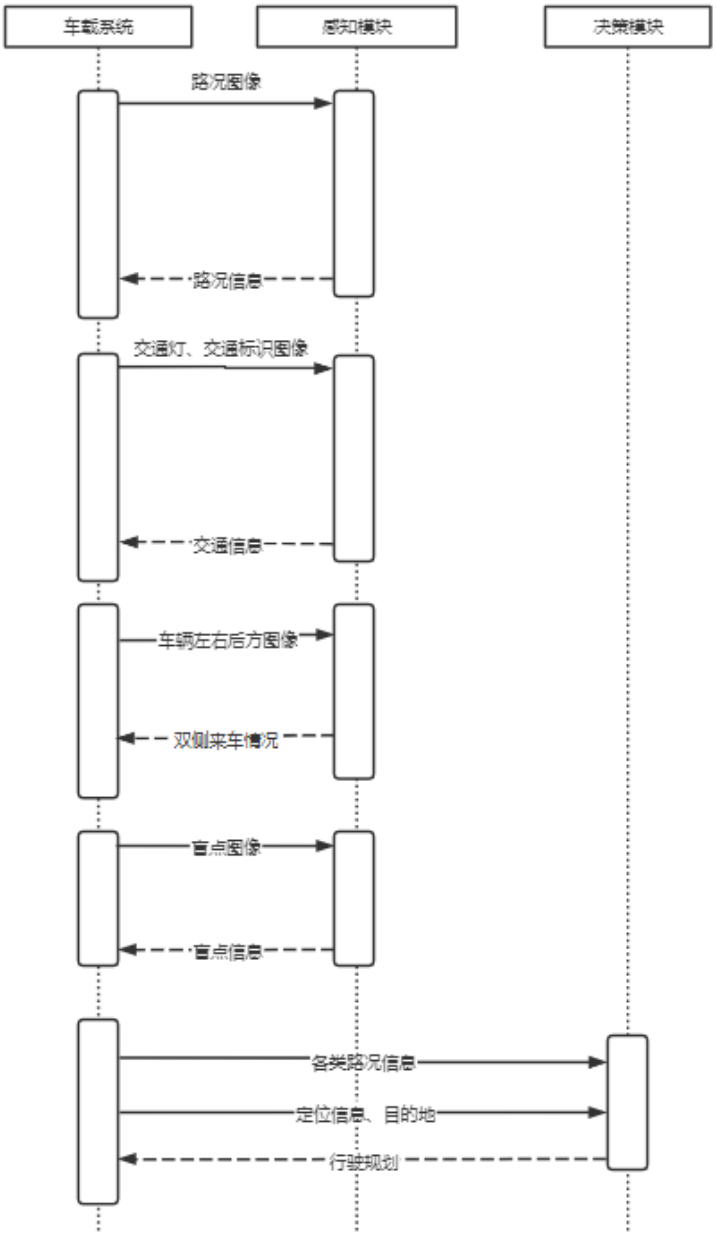
- 规划路线的流程图：

  a. 获取路况信息

  b. 获取交通状况信息

  c. 获取车辆附近的信息

  d. 规划路线



- 自动泊车的流程

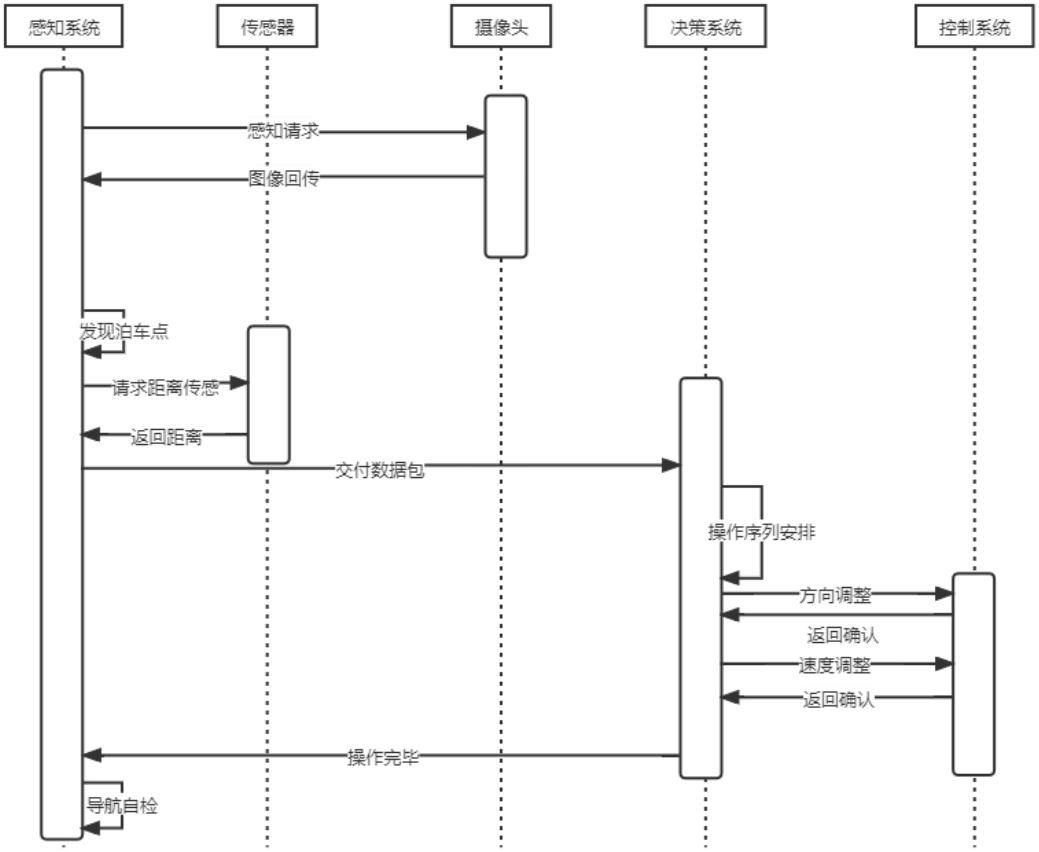
  a. 从摄像头感知泊车位

  b. 感知周边距离

  c. 根据数据决策操作

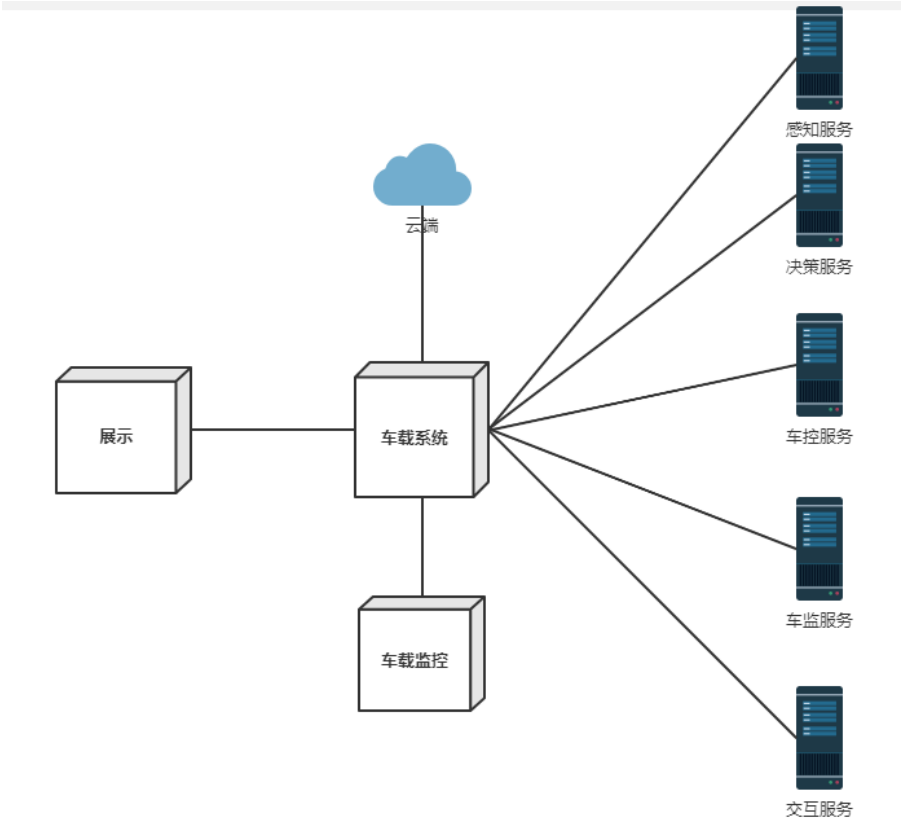
  d. 调整方向和速度

  e. 导航检查是否泊车正确



3.5 体系结构部署

SOA 风格强调的各个服务本身，汽车的车载系统通过调用各个服务来实现对应的功能。



3.6 本章小结

SOA 作为一种面向服务的架构，是一种设计思想和方法论。将 SOA 应用至自动驾驶系统，主要经历了如下流程：

- 需求转化为对应服务模块
- 各服务模块的具体化展开
- 定义车载系统和服务模块之间的关系，即如何使用服务模块来实现自动驾驶功能

SOA 将服务划分开，以便于分别实现与部署，有利于提高每个模块的服务效果，提升整体的实际效果。

4.软件架构对比

我们在本次体系架构中选择了 C/S 架构和面向服务的架构两种，在讨论其对比时，我们认为两者对于功能划分的情况比较类似，但其在组件定义中略有差距，并且由于 SOA 对于信息流动的要求不同，其连接件的设置有所差异，而在业务实现流程上，两者在信息流通上略有差异，大体流程相似，最后，面向服务的架构在部署上相对轻量。

以下将针对于两种架构，结合我们的系统分析其优劣。

4.1 方案一（C/S）分析

将 C/S 架构应用在自动驾驶系统中，首先需要汽车载有一个 PC 端，这样一来很多工作可以在汽车客户端中处理后再提交给服务器，所以汽车自动驾驶系统的响应速度快，减少了服务器的压力。同时，C/S 结构的管理信息系统具有较强的事务处理能力，能实现复杂的业务流程。并且自动驾驶系统的安全性能可以很容易保证，C/S 一般面向相对固定的用户群，程序更加注重流程，它可以对权限进行多层次校验，提供了更安全的存取模式，对信息安全的控制能力很强，因此很适合自动驾驶系统这种对数据保密要求高的软件。

但是该方案也有很多不足之处：

对于客户端来说，只能处理一些功能单一的业务。由于需要安装客户端，安装部署难，所以不易扩展。如果客户端使用的系统不同，就要针对这些系统分别编写程序。界面也缺乏通用性，且当业务更改时就需要更改界面，重新编写。最后，客户端需要专门的客户端程序，比较麻烦，针对点多不能够实现快速部署安装和配置。

对于服务器，用户数增多会出现通信拥堵，服务器响应速度慢等情况。当服务器难当大任时，只能将其废弃，使用更强大的服务器。而且用户是通过 ODBC 连接到数

据库的，且每个连接到数据库的用户都会保持一个 ODBC 连接，会一直占用中央服务器的资源，对服务器的要求很高。

同时，C/S 架构的成本问题也是需要考虑的问题，系统升级维护麻烦，需要更改大量程序，投入大量精力和金钱。业务扩展或变更时，需要更改程序，客户端的界面也需要重新更改。初次投入成本后不能一劳永逸，后期需要很多成本，比如服务器的更换，客户端的更变等。

4.2 方案二（SOA）分析

SOA 架构风格对于自动驾驶系统来说是一个非常合适的架构决策。最直接的因素有两个：① 自动驾驶系统中“多种传感器的感知和融合、复杂多变场景的规划决策、高实时要求的控制执行”涉及大量类型差异很大的计算，需要能被分解成不同的服务子系统，独立进化；②采用“发布/订阅”的通讯模式可以有效的降低各服务之间通讯的耦合性。这两点正好对应于 SOA 架构风格中的“客户-服务”风格和“基于事件/消息的发布订阅”风格两个架构约束。

SOA 应用在自动驾驶系统中，是一种从设计、开发、部署到管理离散逻辑单元（服务）模型。对于整个 SOA 开发过程而言，最主要的就是涉及从车辆特性、系统需求、系统开发、子系统开发、传感控制器总成开发。在 SOA 模型中，几乎所有功能都定义成了独立的服务。服务之间通过交互协调完成业务的整体逻辑。所有的服务通过服务总线或流程管理器来连接。这种松散耦合的架构使得各服务在交互过程中无需考虑双方的内部实现细节，以及部署在什么平台上。此外，便捷的云端访问，服务的精确封装，可有效地保证数据安全，使得车辆不再是信息孤岛，而是物联网中的一个节点，建立了真正的车 - 云通道。可以说 SOA 完美的符合自动驾驶系统。

但是 SOA 也存在一些不足，首先就是复杂性，SOA 服务的划分很困难，编排也并非那么轻松。这也是我们在设计方案二时遇到的最大问题，因为服务的划分本身就是一个系统化工程，然后还要从逻辑上证明这些服务能在一定上被重新编排成其它服务。就像是积木式玩具，其玩具厂商还是要思考思考设计哪些元素构件才能使玩具能构建出多样化的形状。其次，一个最大的问题是 SOA 的分布性质和 web 服务协议的开销，不可否认，任何分布式系统的执行速度都不如独立式系统，这完全是因为网络的制约作用造成的。当然，对于自动驾驶系统这类对实时性要求很高的软件来说，网络引起的延迟是无法容忍的，但同时也是无法避免的，唯一方法就是优化网络带宽从而尽量避免延迟问题。

4.3 方案决策

根据以上对两种架构的分析，我们从以下特性方面进行简要地对比：

- 重用性：SOA 的更多基础服务可以重用，且使用方式更简单
- 复杂性：C/S 的业务流程设计相对于 SOA 服务的划分和编排更简单
- 延迟：C/S 的业务相对集中，延迟略小于服务相对分散的 SOA
- 可扩展性：C/S 业务内耦合较 SOA 程度高，扩展性相对较差
- 安全性：C/S 能设置更多安全验证，安全性更好

结合这些性质的比较，我们认为复杂性的问题是可以解决的，且若真实地大范围落地自动驾驶系统，那么可以使用 5G 技术，甚至设置专门的基站进行辅助，届时延迟的影响将可忽略，而安全性方面，在各服务严格遵循隐私条约的情况下，该影响也将降低，所以，结合发展趋势和规范成熟的背景，我们选择面向服务的软件架构来实现自动驾驶系统。

5.体系结构评估

5.1ATAM 的参与人员

（1）评估小组：该小组是所评估构架的项目外部的小组。组成人员为包亦成，许润。

（2）项目决策者：这些人对开发项目具有发言权，并有权要求进行某些改变。组成人员为杨状凌。

（3）构架涉众： 涉众在构架中有个体既得利益，他们完成工作的能力与支持可修改性、安全性、高可靠性等特性的构架密切相关。组成人员为王博杰，许润，包亦成，王骏。

5.2 质量属性效用树

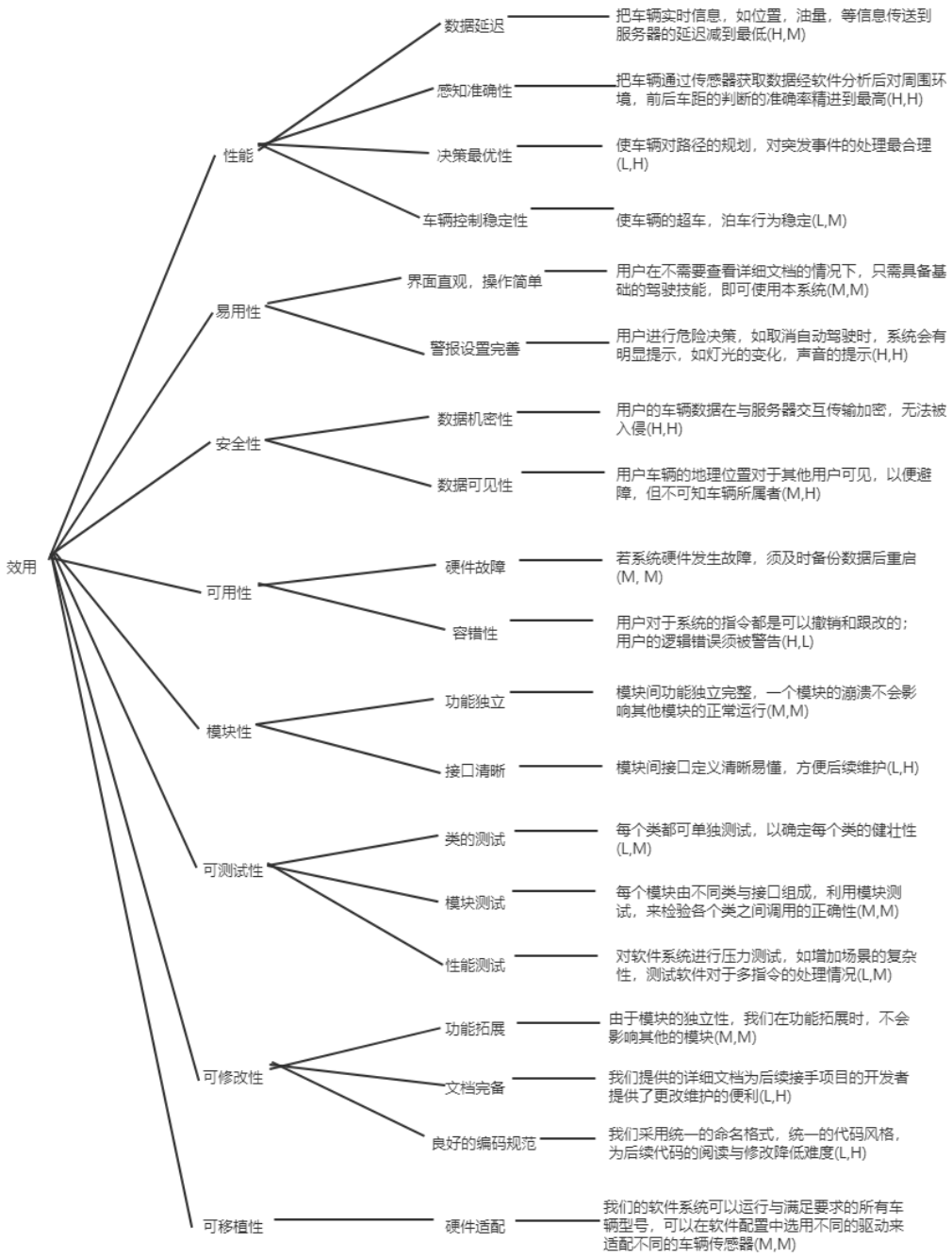
评估小组与项目决策者合作，共同确定出该系统的最重要的质量属性目标，并设置优先级，进行进一步的细化，该步指导其他的分析，这种方式将所有风险承担者和评估小组的精力集中到对系统的成功与否具有重要意义的体系结构的方面上。

质量属性	属性求精	场景编号	场景
性能	数据延迟	XN01	把车辆实时信息，如位置，油量，等信息传送到服务器的延迟减到最低(H,M)
	感知准确性	XN02	把车辆通过传感器获取数据经软件分析后对周围环境，前后车

			距的判断的准确率精进到最高(H,H)
	决策最优性	XN03	使车辆对路径的规划，对突发事件的处理最合理(L,H)
	车辆控制稳定性	XN04	使车辆的超车，泊车行为稳定(L,M)
易用性	界面直观，操作简单	YY01	用户在不需要查看详细文档的情况下，只需具备基础的驾驶技能，即可使用本系统(M,M)
	警报设置完善	YY02	用户进行危险决策，如取消自动驾驶时，系统会有明显提示，如灯光的变化，声音的提示(H,H)
安全性	数据机密性	AQ01	用户的车辆数据在与服务器交互传输加密，无法被入侵(H,H)
	数据的可见性	AQ02	用户车辆的地理位置对于其他用户可见，以便避障，但不可知车辆所属者(M,H)
可用性	硬件故障	KY01	若系统硬件发生故障，须及时备份数据后重启(M, M)
	容错性	KY02	用户对于系统的指令都是可以撤销和更改的；用户的逻辑错误须被警告(H,L)
模块性	功能独立	MK01	模块间功能独立完整，一个模块的崩溃不会影响其他模块的正常运行(M,M)

	接口清晰	MK02	模块间接口定义清晰易懂，方便后续维护(L,H)
可测试性	类的测试	KCS01	每个类都可单独测试，以确定每个类的健壮性(L,M)
	模块测试	KCS02	每个模块由不同类与接口组成，利用模块测试，来检验各个类之间调用的正确性(M,M)
	性能测试	KCS03	对软件系统进行压力测试，如增加场景的复杂性，测试软件对于多指令的处理情况(L,M)
可修改性	功能拓展	KXG01	由于模块的独立性，我们在功能拓展时，不会影响其他的模块(M,M)
	文档完备	KXG02	我们提供的详细文档为后续接手项目的开发者提供了更改维护的便利(L,H)
	良好的编码规范	KXG03	我们采用统一的命名格式，统一的代码风格，为后续代码的阅读与修改降低难度(L,H)
可移植性	硬件适配	KYZ01	我们的软件系统可以运行与满足要求的所有车辆型号，可以在软件配置中选用不同的驱动来适配不同的车辆传感器(M,M)

质量属性效用树如下所示：



效用树为我们提供了一种直接而有效地将系统的业务驱动因素转换为具体的质量属性场景的机制，该步骤的输出结果是对具体质量属性需求(以场景形式实现)的优先级的确定。

5.3 质量场景的架构分析

本节将对两个优先级最高（为(H,H)）的场景进行分析，即场景 XN02，YY02。下面根据实现质量属性所采用的战术对这些重要场景给出构架分析表格。

场景号：XN02	场景：感知场景复杂性增加			
属性	性能			
环境	雨雾天气，低能见度下，多车辆，距离紧密			
刺激	外部环境变化			
响应	及时改变传感器的触发情况			
构架决策	敏感点	权衡点	有风险决策	无风险决策
开启红外夜成像仪	S1			N1
启用毫米波雷达定位	S2			N2
采用雨雾天气图像处理技术	S3		R1	
推理	● 开启夜成像可以提高模糊低光场景下的图像质量 ● 利用毫米波雷达可以处理密集障碍物下的障碍物距离感知 ● 采用不同的图像处理技术，可以让传感器获得的图片信息更加直观具象，但选用错误的图像处理技术可能会引发负面效果，具有一定的决策风险			

场景号：YY02	场景：用户做出错误或危险指令时警报触发			
属性	易用性，容错性			
环境	汽车正常行驶			
刺激	用户做出危险操作			
响应	将用户指令暂存，发出警告			
构架决策	敏感点	权衡点	有风险决策	无风险决策
暂存用户指令		T1	R2	
触发音响警报		T2	R3	
触发视觉警告		T2		N3

推理	● 暂存用户指令，在某些情况下，车辆可能存在误检测状况，此时应在用户仔细确认指令安全后，调用暂存的指令进行执行；但存在用户故意损害车辆设备的风险 ● 触发音响警报，在用户行驶过程中，不可能一直盯着屏幕看，故需要通过音效来提示用户指令风险，但可能会干扰用户听到其他车辆的喇叭声 ● 通过灯光效果变化，用户节目弹窗来警告用户指令的风险

标注表

序号	内容
S1	夜视能力的加强会影响系统的识别性能
S2	增强雷达定位能力，能使位置信息更准确，提高系统性能
S3	引入雨雾图像处理技术，可能使得识别性能更好，但也可能相反
T1	指令缓存会同时影响系统的可用性，易用性与安全性，缓存越大，整体可追溯指令越多，安全性与易用性越强
T2	警告效果越好，车辆的易用性与安全性越高
R1	雨雾图像处理技术可以增强图像在雨雾天气的清晰度，但由于对原图进行处理，可能丢失一些关键信息，给决策带来风险
R2	暂存指令的指令提高了系统易用性，但同时暂存的指令大多为系统认定的具有风险的命令，用户再次调用可能对系统造成威胁
R3	音响警报为用户带来清晰的提醒，但同时也干扰了用户对外界喇叭等信息的收集，带来一定风险
N1	开启红外夜成像增加了数据源的获取途径，不会产生风险
N2	开启毫米波雷达增加了数据源获取途径，精确了周边小范围复杂环境的定位，不会产生风险
N3	视觉警报为用户带来直观的警告提醒，不干扰用户其他信息的收集，不产生风险

5.4 对系统架构的再分析

5.4.1 架构一: CS 风格

采用战术	敏感点(S)	有风险决策(R)
客户机与服务器的分离	提高数据处理速度，降低对车辆系统的算力需求	车辆进入信号差的地方，数据传输会受阻，形成决策高延迟的情况，此时应依赖与车主手动操作或调用车载系统自带的弱性能处理方案
高内聚低耦合策略	降低了层与层之间的依赖性，提高了复用性	
统一接口策略	提高了多人员开发时的效率，提高代码的健壮性	
分布式部署	将数据分布式存储，用户在不同地区访问服务器的速度都应当快速低延迟	

上表分别从性能战术，可测试性战术，可拓展性战术对架构进行了再分析

5.4.2 架构二:SOA 风格

采用战术	敏感点(S)	有风险决策(R)
功能独立	便于系统的升级，更新功能时不会影响其他功能的使用	
软硬分离策略	系统不依赖于固定的硬件，只需硬件做好适配即可运行服务	
高算力需求迁移至云端	提高车载服务对大数据处理的响应速度	当网络不佳时，若决策过度依赖云端，会造成决策错误或延迟
分布式部署	将数据分布式存储，用户在不同地区访问服务器的速度都应当快速低延迟	

上表分别从可拓展性战术，可移植性战术，性能战术对架构进行了再分析

5.5 评审结论

通过上面属性场景再分析可以看出，两种方案各有千秋。但他们都有共同的痛点，就是云计算的依赖，这种依赖的实时性不可避免地与当前时代下网络环境的不稳定性有着冲突。但总的来说，两种架构都达到了设计时预期的要求。

6.开发计划

6.1 过程安排

编号	阶段	工作内容	具体工作	负责人	截止时间	备注
1-1	项目启动	项目确定	分析项目内容	——	2021.12.01	
1-2		简要需求确立	简单分析项目需求，讨论是否可行	——	2021.12.01	
1-3		项目启动讨论	确定项目负责人以及模块负责人	杨状凌	2021.12.02	
1-4		项目实施计划讨论	组内讨论项目里程时间，以及初步划分项目模块	——	2021.12.02	
1-5		前期准备	确定需要的项目背景资料支持和技术支持，并对其进行检索查阅	——	2021.12.03	
2-1	需求分析	明确目标	对自动驾驶系统的相关角色进行定义	许润	2021.12.04	
2-2		总体需求	找出自动驾驶的需求方向	——	2021.12.04	
2-3		用例抽取	将总体需求进行解耦分析	——	2021.12.04	
2-4		用例描述	对每个用例模块进行简略分析	——	2021.12.05	
2-5		详细分析	对每个用例模块进行详细的展开讨论	——	2021.12.06	
2-6		总结确认	确认需求的完备性，补充缺失的需求	——	2021.12.06	
3-1	概要设计	约束明确	设计各类约束	王骏	2021.12.07	
3-2		模块分解	将实现模块进行功能模块的解耦划分	——	2021.12.07	
3-3		流程分析	分析每个业务模块的业务流程	——	2021.12.08	

3-4		实体设计	设计访问数据库的实体类	——	2021.12.09	
3-5		架构设计	建立架构模型	——	2021.12.10	
3-6		架构评估	分析架构模型	——	2021.12.10	
3-7		用户界面设计	设计具体的展示界面	——	2021.12.11	
3-8		概要设计总结	对之前的概要设计进行总结讨论	——	2021.12.12	
4-1	详细设计	界面详细设计	对展示界面进行详细的设计	杨状凌	2021.12.13	
4-2		数据访问设计	对实体类的属性进行定义	——	2021.12.14	
4-3		业务流程设计	详细展开业务流程	——	2021.12.16	
4-4		详细设计讨论	对之前的详细设计结果进行讨论分析	——	2021.12.18	
4-5		详细设计确立	确立较为合理的详细设计结果	——	2021.12.19	
4-6		详细设计总结	总结完善详细设计	——	2021.12.19	
5-1	代码开发	设备采购	采购想要使用的设备	杨状凌	2021.12.20	
5-2		环境搭建	搭建运行环境	王博杰	2021.12.20	
5-3		启动后端开发	开展后端开发的工作	——	2021.12.20	
5-4		接口编写	按照设计计划规范编写功能接口	——	2021.12.24	
5-5		接口记录	文档规范记录接口信息	——	2021.12.24	
5-6		启动前端开发	讨论决定前端开发计划和负责人	包亦成	2021.12.24	
5-7		界面编写	对项目各页面和功能进行界面开发	——	2021.12.28	
5-8		前后端组合	在前端中进行接口规范使用	——	2021.12.30	
6-1	测试部署	内部测试	组内项目将在内部测试人员手中试运行，并提供运行报告	王骏	2022.1.1	
6-2		客户测试	项目移交客户，提供短期试运行	——	2022.1.3	

6-3		问题集中	将测试问题集中，组内讨论解决方案	——	2022.1.4	
6-4		漏洞修复	组内针对反馈漏洞进行改法交流，并及时修改	——	2022.1.8	
6-5		系统上线	将初成型版本部署到服务器，进行限额上线运行测试	——	2022.1.9	
6-6		测试版试运行	为期半个月的市场测试	——	2022.1.9	
6-7		正式上线	试运行正常后，将进行市场化的正式上线	——	2022.1.10	
6-8		项目交付	完成项目	——	2022.1.10	

6.2 项目管理计划

编号	管理条目	管理对象	具体内容	备注
1	项目周推进	全组	每个工作周的结束，整合当周内容，完成对组内项目进行审核查验	
2	进度评估	全组	每天对项目完成度进行评估，及时对全组发出进度落后预警	
3	组织学习	全组	及时地发现组内知识缺失，并组织全体成员集中学习	
4	责任分担	全组	根据组内成员的特长，为其分配相应的直接负责内容	
5	支出明确	全组	所有项目内需开支，应该记录可查	
6	阶段讨论	阶段负责人	每完成一个开发阶段，由相关阶段负责人带领全组进行内容回顾和讨论	
7	记录明确	全组	各成员完成相关开发任务，都应有文字记录，便于工作量查询	
8	结题表彰	全组	项目结束后，将根据产品模块工作量以及组员各自的记录文档进行工作量总结，并给予相应表彰	

7.个人小结

组长：

71119101—杨状凌

经历了本次软件体系架构实验，我作为组长对整个项目进行了日程的管理和规划，并且自身也参与了对于其中一部分内容的报告。

对于体系架构的实际设计，首先我们将报告分出需求、方案一和方案二以及对比评估的大结构，并设立了每周一里程碑，完成一个大的模块，同时，我们也设立每周反思环节，即对于之前的工作，我们将在组内讨论其合理性，以及是否能与之后的内容协调起来，并且进行适量的更新；其次我们每周将当周任务细分，首先确立大致思路，然后由所有同学开始动工，结束后，进行整合，整合过程中将由全组仲裁冲突，形成一周的阶段成果；最后在整个工程中，我们坚持每次集中人员对体系架构进行报告撰写，使得有问题及时提出、有冲突及时讨论，如果只是线上，我想并没有这样的效果，并且在现场时，我们可以及时补充遗忘的知识，在共同监督下进行学习，提高专注度和进行现场学习反馈。

对于体系架构实验中的执行，我们选择了自动驾驶系统进行软件设计。我们在实验中，结合自身的驾驶经验以及网上的自动驾驶系统相关的材料，设立了多个驾驶情况，例如泊车、超车以及紧急制动等等，并且对于各个驾驶情况，我们从信息获取到决策执行，将整个系统分为了感知、决策、车控、车监和交互等五个模块，遵循了一个易于接受的事件处理流程；我们的系统也需要许多外界的软硬件支持，例如已经成型的卫星导航工程和数量繁多的光传感器、气压传感器以及摄像头等等，这是我们将软硬件结合设计的第一个工程。

我们将实验设计分为了 C/S 架构和面向服务的架构两种方式，前者是一种非常通用的架构形式，对于车载系统，其也可以认为是一个一个的客户端与数据处理的服务器相连接；而后者中，我们认为各个模块中，能将许多功能划分为一个个服务，这些服务可以独立开来，它们通过信息互连，完成一个个完整的业务流程。

之后我们将两种架构简单地对比了其异同点，并独立分析了两个方案各自的优缺点。

最后我们利用 ATAM 方法对我们的设计进行了评估，说明了其中的一些敏感点和决策点等特征，并且画出了质量属性效用树，为各个场景进行了简单的评估。

虽然我们最终并没有完成代码实现，但我们也给出了包括实现和测试在内的完整项目过程安排，体现出了我们的计划表。

本次实验让我更加熟悉了一个团队项目的计划安排的重要性，也只有多交流多讨论，才能发现各位想法的不同，更完善我们的实验内容。

组员：

71119102—王博杰

本次体系结构实验，我们小组选题为自动驾驶系统的应用设计。我们完成了项目的需求分析、两种体系风格的实现方案设计、方案的小型对比分析、体系结构评估。本人主要负责了需求分析部分、C/S 架构风格设计部分的实现。

我们的项目需求基本上可以划分为感知模块、决策模块、车控模块、车监模块和交互模块五类大模块。在分析需求时，难点在于如何将不同的功能需求划分到相应的功能模块下，实现需求的解耦操作，期间进行了讨论分析，经历了多次版本的迭代，最终确定了当前的划分结果。对于 C/S 软件体系结构选型的情况下对自动驾驶系统的设计，在确定组件时有些犹豫，再结合以前开发桌面端小程序的经验之后，确定了将每个功能需求模块实现为一个单独的业务类的决定，设计了上述的业务逻辑层的结构。

最后的体系结构评估部分，涉及到了两种不同的体系结构，我们尽量对需求进行分析，使其评估方案能够同时适用两种架构方案。其中的分析方案部分要进行较为细致的补充与分析，否则难以做到对体系结构的正确评估结果。

本次体系结构设计的实验，从课堂中走出来，以组队的形式来探讨分析，有利于我们对于课堂知识的再回顾和深入掌握，遗憾的是，由于时间原因，不能涉及到代码层面的实现，不能用代码来更深层次的检验设计的合理性，是一大遗憾。

71119103—许润

本次的软件体系结构实验二，我们组选择的是自动驾驶系统。相比于其他几个主题（如外卖点餐系统、多人在线游戏等等），其实这个主题是相对陌生的，因为没有相关的软件供我们参考分析。并且由于该系统是软硬件交互的，因此分析、实现起来也较为困难。

在设计的过程中，尤其是 SOA 风格，之前也没有接触过，只能边学边用。所以直至现在，我也不敢说自己的设计是正确的，有很多地方也是模棱两可，感觉自己的软件架构意识还是不强，组件以及连接件的判断也不是很清晰。不过这正是学习的过程，如今我感觉对体系架构有了初步的认识，希望自己在以后碰到类似的设计工作时，能够合理的运用学过的东西，逐渐培养起软件架构的意识。不能只放眼于具体的代码实现阶段，而是需要用一个类似于软件构架师的思维和角度去仔细思考和斟酌如何才能设计出一个体系结构相对合理，存储方式比较简洁的产品项目。

经过从软件工程导论到暑期实训，再到现在的软件体系结构的几次实验，我也学习并实践了不少架构风格（C/S，B/S，微服务，SOA 等等），对架构也有了一些新的认知，什么样的架构才是好架构呢？之前有听到过一句话，概括的很精辟。好的架构必须需要贴合业务，那么把业务技术演变成一个数学公式来表达可以理解为:2 个数字的和等于 10，求如何组合能得到最大的乘积。那不是 3*7，也不似乎 4*6，而是 5*5。所以架构不是生搬硬套，为了架构而架构。我们应避免通过个人的主观意愿来主导。比如自己觉得某个中间件好，就拿着锤子到处找钉子，这一敲下来，看着不错，但是带来的成本和风险被忽略了。可能有更好的解决方案，或者完全没必要在当下敲这一锤子下去。

71119104—包亦成

在这次的体系结构设计中，我们小组完成了对一个自动驾驶系统的两种体系结构设计，分别是C/S架构和面向服务的架构，完成了功能需求分析，两种架构风格的设计以及详细实现、两种架构风格的评估与对比，总结其优缺点，最后还列出了开发计划，总的来说经历了一个完整架构方案分析的过程，也学到了很多，当然也有很多不足。深刻认识到了架构设计的重要性，对整个系统的整体稳定性有着非常重要的地位。

在本次实验中，我主要负责的部分是初始需求的设计和细化，C/S架构的设计以及详细实现方案，还有体系结构评估。其实在最初的需求导出和精化的过程中，我们出现了一个比较大失误，因为我们并没有分清这个系统中的各个角色就去分析需求，尽管能出来一些需求，但这时的需求是十分散乱的。之后在老师的指引下我们迅速纠正了错误，之后也就顺利的完成了需求分析。在我主要负责的C/S架构设计中，一开始是比较纠结的，因为确实对这种自动驾驶系统之前没有很深的了解，因此也只能在查阅大量资料和结合自己以往在软件体系结构课程中的所学来完成架构设计，必定存在很多不足。设计的三层C/S架构也是较为不成熟的，但就这个过程来说，我体验了架构设计的流程，也为以后架构设计打了基础，更加有方向了。

尽管这次实验是比较仓促的，也只是进行了架构的设计，并没有落实到真实的开发中，但是因为是第一次的架构设计，因此受益还是很多的。

71119138—王骏

本次软件体系结构实验，承继上一次实验与教学内容。通过本次实验，我们小组协作，为自动驾驶系统创建了cs和soa架构。在这之前，我们结合上课内容，对这个系统进行分析，最终选择了这两个可以适应我们需求的架构。而最终的架构评估也验证了我们一开始的猜测是对的。而通过这次实验，我们也更加深刻地认识到软件架构对系统稳定性，对后续开发维护与升级有着举足轻重的作用。

本次实验中我个人主要参与最开始的需求分析研讨与分类。最基础的需求分析花费了我们很多时间，我们很小心地将需求进行分类，模块化。但在后面的架构设计中，我们依然经历了需求的变更。这种从需求发生的变更，让我们的进度一度有所停滞。另外，在之后的工作中，我负责了架构的评估。利用ATAM评估策略，我对cs架构和soa架构进行了分析。在评估后，我们很开心地看到我们的架构还是比较符合需求的。在性能方面，易用性，可拓展性，可维护性都有很好的表现。

通过这次实验，我受益匪浅。一方面锻炼了我团队协作的能力，与队友们合理的分工让团队的效率很高，每个人符合不同的模块，让我们最终的进展有了1+1>2的效果；另一方面，本次实验也让我将课堂中的知识应用到实际，从需求分析到架构评估，这样一套完整的实验让我更加了解软件开发设计的流程，也让我体会到软件不仅仅是代码，是可执行文件,更是文档，代码与数据结构的集合体。

附录：版本修改说明

1.需求分析修改

针对于第一章说明的所有的模块，将系统中的子模块重新定义为用例，并将具体功能展开为被各模块 include 的用例，重新定义每个用例图的 Actor 的具体角色是驾驶员、车（传感器）、导航/网络中一个或多个，同时对应修改具体用例图对应的表格。

2.新增 SOA 服务划分

针对于第三章说明的 SOA 架构，我们分出了新的子章节 3.3.0 来划分和说明我们的服务，服务分类为本地和云端服务，并在其中给出了一些我们的思考。

3.新增架构评估标注

我们在 5.3 节架构评估中补充了表格中例如 S1、N1 等标注，在其下列表解释了每个标号代表的意义。

4.新增决策陈述

我们在第四章对于架构的初步分析后，权衡优缺点，补充了对于选择架构决策的陈述，详情请见 4.3 章节。

5.文章完善

修改了如恶"行"拥堵、收"货"、依"耐"性、车"声"温度、部"属"，不"在"是、"跟"改、"锄"溃等错别字，和统一了全篇的一些名词例如车"检"统一为车"监"，并将段落语句修改地更加通顺，完善了文章的表述。