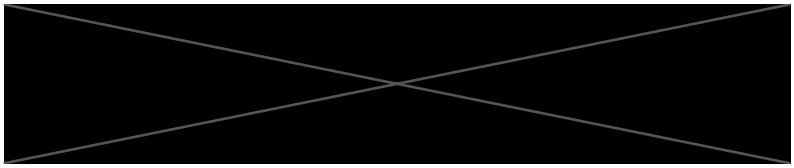
1040601 SWE.2 OTA车端软件架构设计

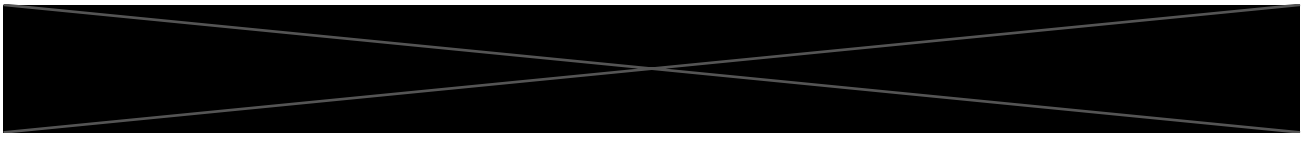


1 文档介绍 Introduction

1.1 范围说明 Scope

本文档是对J01 OTA 相关功能的概要设计 ， 为系统设计人员，开发人员，测试人员及其他相关人员提供背景、架构等方面的指导和说明。

1.2 参考文档 References



1.3 读者对象 Target Audience

项目经理，产品工程师，开发人员，测试人员及其他相关人员。

1.4. 术语和缩写 Definitions and Abbreviations

本章节对本文中使用到的术语定义和缩写进行解释。

缩写 (Abbreviation)	英文全称(Full Name)	中文描述（Chinese description）
OTA	Over-the-Air Technology	远程无线升级技术
FOTA	Firmware Over-The-Air	固件空中升级技术
ECU	Electronic Control Unit	电子控制器单元
TSP	Telematics Service Provider	远程通信服务供应商
HMI	Human Machine Interface	人机交互接口
FBL	Flash Bootloader	闪存引导加载程序
CDN	Content Delivery Network	内容分发网络服务器系统模块，用于管理升级包
OTA Server	Over the air server	OTA管理平台，负责车辆的版本管理，任务发布，升级结果展示等
MCU	Micro controller Unit	微控制单元，作为单独升级模块
APK	Android application package	OTA 人机交互Android应用程序包
UCM	Update Control Master	OTA控制主模块，用来负责OTA升级主流程控制管理
Lite	Update Control Lite	OTA 控制从模块，用来负责提供OTA升级所需的关键能力支持（如文件下载，车辆状态信息获取等）。
Lite2hmi	Update Control Lite to HMI	OTA 控制从模块，用来负责对接HMI界面展现相关内容的交互
UA	Updata Agent	差分升级模块，负责SOC的升级
OTA Client	Over the air client	OTA车端车云通信客户端，用于车云通信
DIM	Device Interface Manager	设备交互管理模块，用于管理需要被升级的零件
UIM	User Interaction Manager	用户交互管理模块，用于管理人机交互
DPM	Download Policy Manager	下载策略管理
IPM	Install Policy Manager	安装策略管理
DIM	Device Interface Manager	设备交互管理模块，用于管理需要被升级的零件
SM	Security Manager	提供必要的安全措施，包括数据加密、身份验证和完整性校验，以保护Lite设备免受未经授权的访问和篡改。
D2B	Device to Business Manager	需要宿主设备提供的一些接口来实现功能
UAM	Update Agent Manager	在宿主设备是智能件时，负责升级刷写操作和部分零件信息的采集工作
VSM	Vehicle Status Manager	在OTA过程中读取、设置和监控车辆状态，如车速、档位、电池电量等，以及设置整车的高压状态。
VDM	Vehicle Diagnostic Manager	适用于非智能件的刷写及车辆诊断，以及执行部分零件信息的采集工作

1.5 架构模型层次定义 Architecture Model and Layer Definition

参考C4模型的层次定义https://c4model.com/

L1 → L3 示例

L1 级： 系统上下文级别，描述系统间功能模块划分, 适用于《系统上下文》章节。

L2级： 容器图级别，容器是可单独运行/可部署的单元（例如，单独的进程空间） ）执行代码或存储数据。容器图显示了软件体系结构的高层结构以及如何在其间分配职责。它还显示了主要的技术选择以及容器之间的通信方式

L3级： 模块/组件级别， 描述模块与模块内部调用关系，前后端模块依赖关系，模块与模块层次关系， 模块与模块数据流/控制流关系，模块与模块间外部接口。适用于《软件组件设计》章节，和详细设计文档软件架构上下文引用。

L4级： 类关系级别， 描述模块/组件内部类关系（继承，组合，聚合等），数据结构， 数据库表单设计，接口详细设计， JSON字段定义等，适用于详细设计文档，

扩展级： 部署图，网络拓扑图，数据库结构

本概要设计只涉及到L1-L3级架构，和扩展级（如果有需要）

L3级与L4级设计推荐采用PlantUML 方式，以Code as UML形式后续可随代码仓库迁入/迁出 方便版本管理，设计与代码同步

2 架构设计 Architecture Design

2.1. 设计原则 Design principles

设计开发需遵循以下原则：

简明易懂： 作为首要的原则，架构设计需要保证任何具有软件开发经验的人员都能够快速的了解所设计的内容，并能快速的进行进一步的工作。

可扩展： 架构设计需要考虑后续的可扩展性。

可重用： 架构设计需要考虑应可重用性，各个模块及功能都需要在不进行大的改动下可以被集成到其他项目或应用中。

低耦合： 各个模块 及功能间的耦合度要尽量低.

2.2. 可复用模块设计 Reusable Module / Component Design

NA，首次开发设计。

2.2.1 公共库

FOTA软件开发依赖的第三方开源组件库依赖情况如下：

开源组件			许可证		大小
名称	版本	下载链接	类型	链接	库
openssl	1.1.1n	https://www.openssl.org/source/old/1.1.1/openssl-1.1.1n.tar.gz	GNU	https://github.com/openssl/openssl/blob/OpenSSL_1_1_1n/LICENSE	3.64MB（静态库）
curl	7.88.1	https://github.com/curl/curl/releases/download/curl-7_88_1/curl-7.88.1.tar.gz	MIT	https://github.com/curl/curl/tree/curl-7_88_1/LICENSES	10000KB（静态库）

leveldb	1.22	https://github.com/google/leveldb/archive/refs/tags/1.22.tar.gz	BSD	https://github.com/google/leveldb/blob/1.22/LICENSE	989KB（静态库）
iniparser	4.1	https://github.com/ndevilla/iniparser/archive/refs/tags/v4.1.tar.gz	MIT	https://github.com/ndevilla/iniparser/blob/v4.1/LICENSE	12KB（静态库）
EasyLogger	2.2.0	https://github.com/armink/EasyLogger/archive/refs/tags/2.2.0.tar.gz	MIT	https://github.com/armink/EasyLogger/blob/2.2.0/LICENSE	22KB（静态库）
cJSON	1.7.15	https://github.com/DaveGamble/cJSON/archive/refs/tags/v1.7.15.tar.gz	MIT	https://github.com/DaveGamble/cJSON/blob/v1.7.15/LICENSE	27KB（静态库）
zlib	1.2.11	https://github.com/madler/zlib/archive/refs/tags/v1.2.11.tar.gz	GNU	https://github.com/madler/zlib/blob/v1.2.11/README	115KB（静态库）

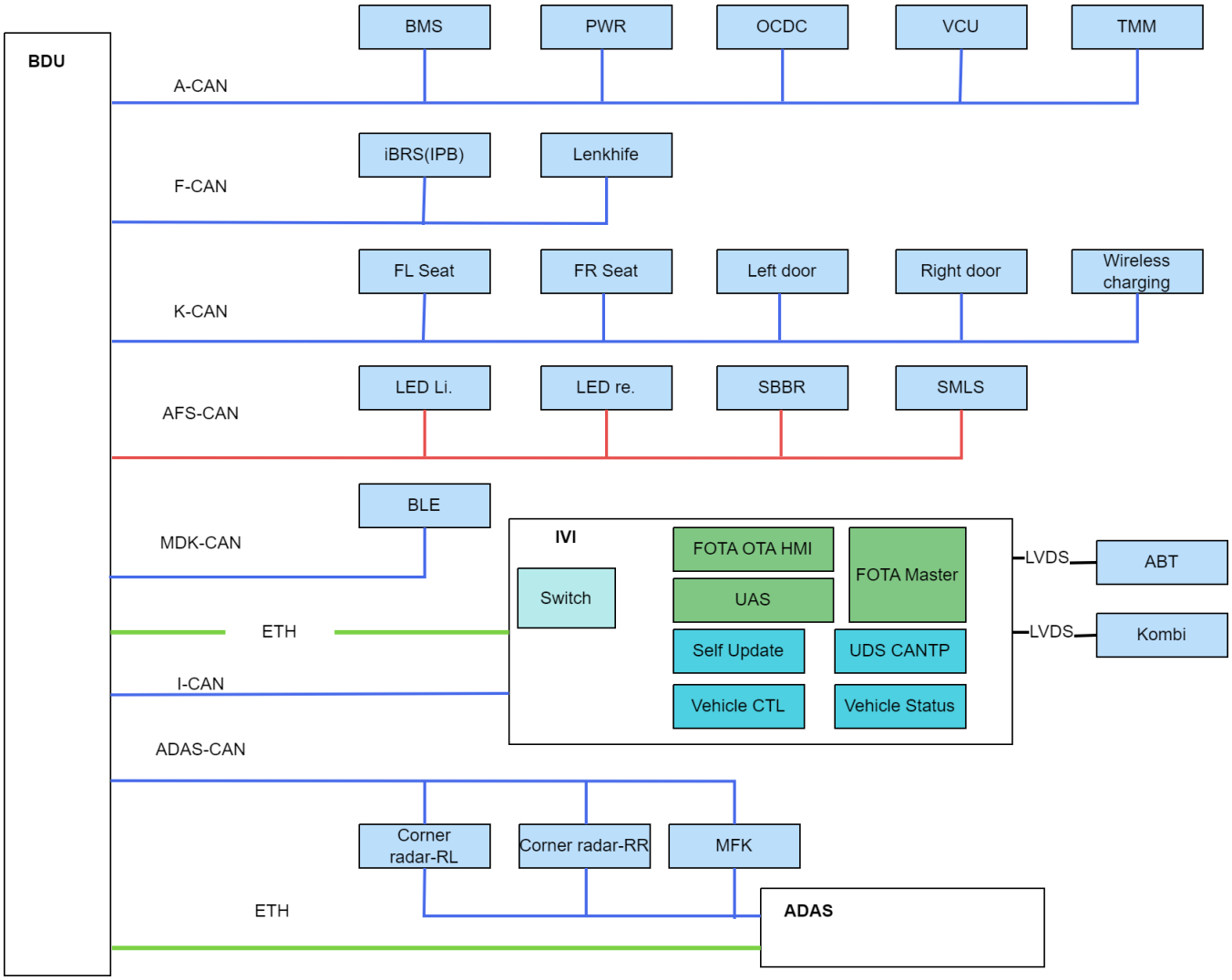
2.3 系统上下文 / System Context（粗粒度 L1级）

2.3.1 上下文框图 Context Diagram

系统上下文框图描述软件运行的上下文环境，框图需要包括上下文运行环境中与本软件模块相关的外围软件模块，并说明这些软件模块的功能、和本软件模块的关系等。典型的总体框图如图：

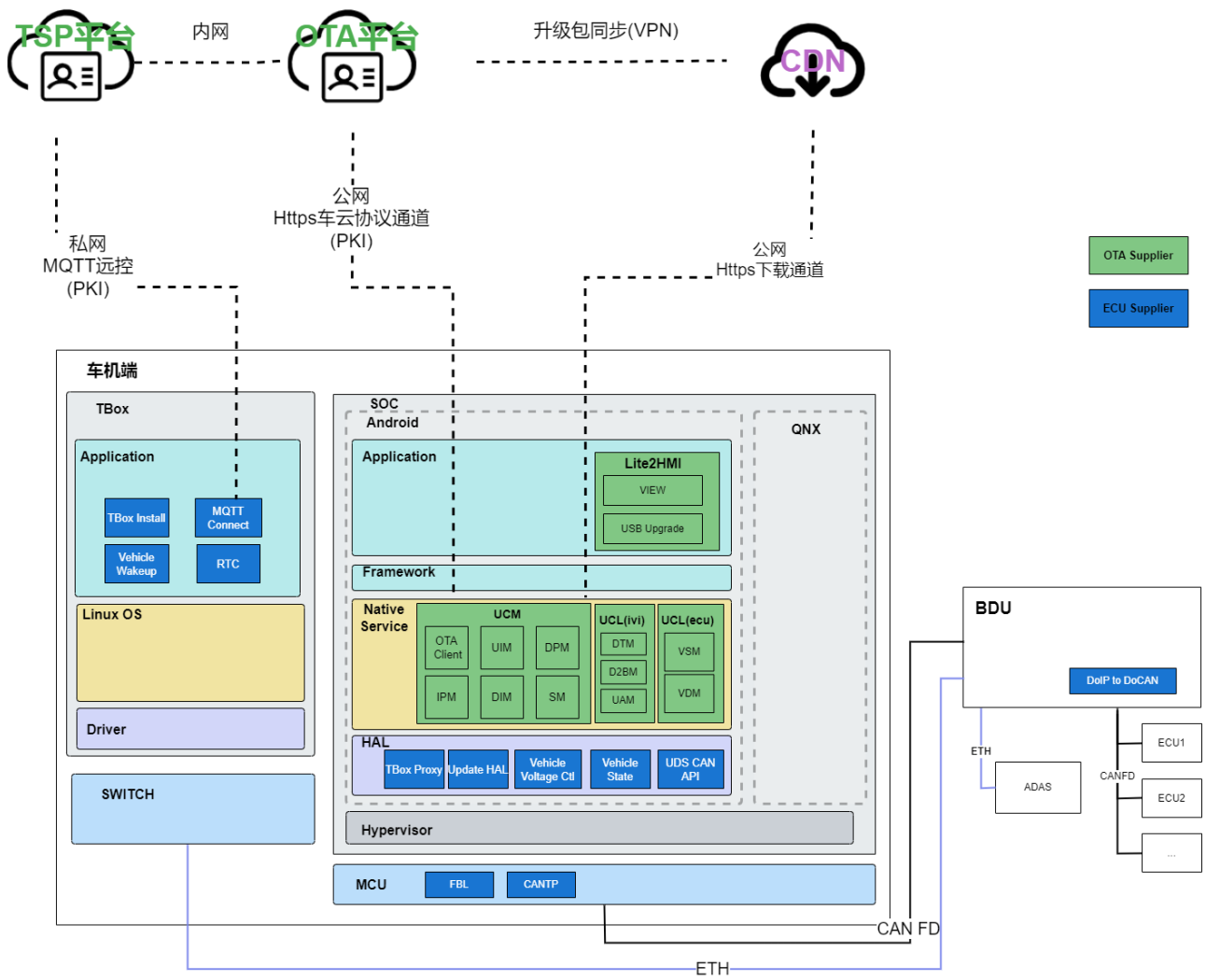
描述业务端到端模块分布及连接关系（系统上下文级别 L1）：

本项目是基于J0x车型架构展开OTA车端设计，车端OTA组件主要部署于IVI中，整车网络拓扑见下图。



整车网络拓扑图

IVI系统中的部署情况见下图：



OTA在IVI系统部署图

2.3.2 上下文组件列表 Context Component List

描述上下文模块的功能和与本软件模块的关系：

模块名称	功能描述以及和本模块的关系	模块设计引用
[redacted]		
UIM(User Interaction Manager)	用户交互管理模块，用于管理人机交互	
DPM(Download Policy Manager)	下载策略管理	
IPM(Install Policy Manager)	安装策略管理	
DIM(Device Interface Manager)	设备交互管理模块，用于管理需要被升级的零件	

SM（Security Manager）	提供必要的安全措施，包括数据加密、身份验证和完整性校验，以保护Lite设备免受未经授权的访问和篡改。	
D2B(Device to Business Manager)	需要宿主设备提供的一些接口来实现功能	
UAM（Update Agent Manager）	在宿主设备是智能件时，负责升级刷写操作和部分零件信息的采集工作	
VSM（Vehicle Status Manager）	在OTA过程中读取、设置和监控车辆状态，如车速、档位、电池电量等，以及设置整车的高压状态。	
VDM（Vehicle Diagnostic Manager）	适用于非智能件的刷写及车辆诊断，以及执行部分零件信息的采集工作	
USB Upgrade	USB升级模块	

2.3.3 外部接口依赖 External Interface Dependency

描述依赖的外部接口，这些接口由上下文模块提供。

接口名称	功能描述	使用方	提供方
OTA 车云协议	UCM向OTA管理平台发起车辆注册、车辆配置信息获取、版本检测、升级包下载、信息上报等	UCM	OTA Server
UCM与UCL交互接口	UCM向Lite请求特定零部件的信息采集、升级执行等	UCM	Lite
UCM与Lite2Hmi交互接口	Lite2hmi向UCM转发人机界面的指令，如触发下载、安装等	Lite2HMI	UCM
UCM与CDN交互协议	UCM从CDN服务器下载升级包	UCM	CDN

2.3.4 对外暴露接口 Exposed Interface

无

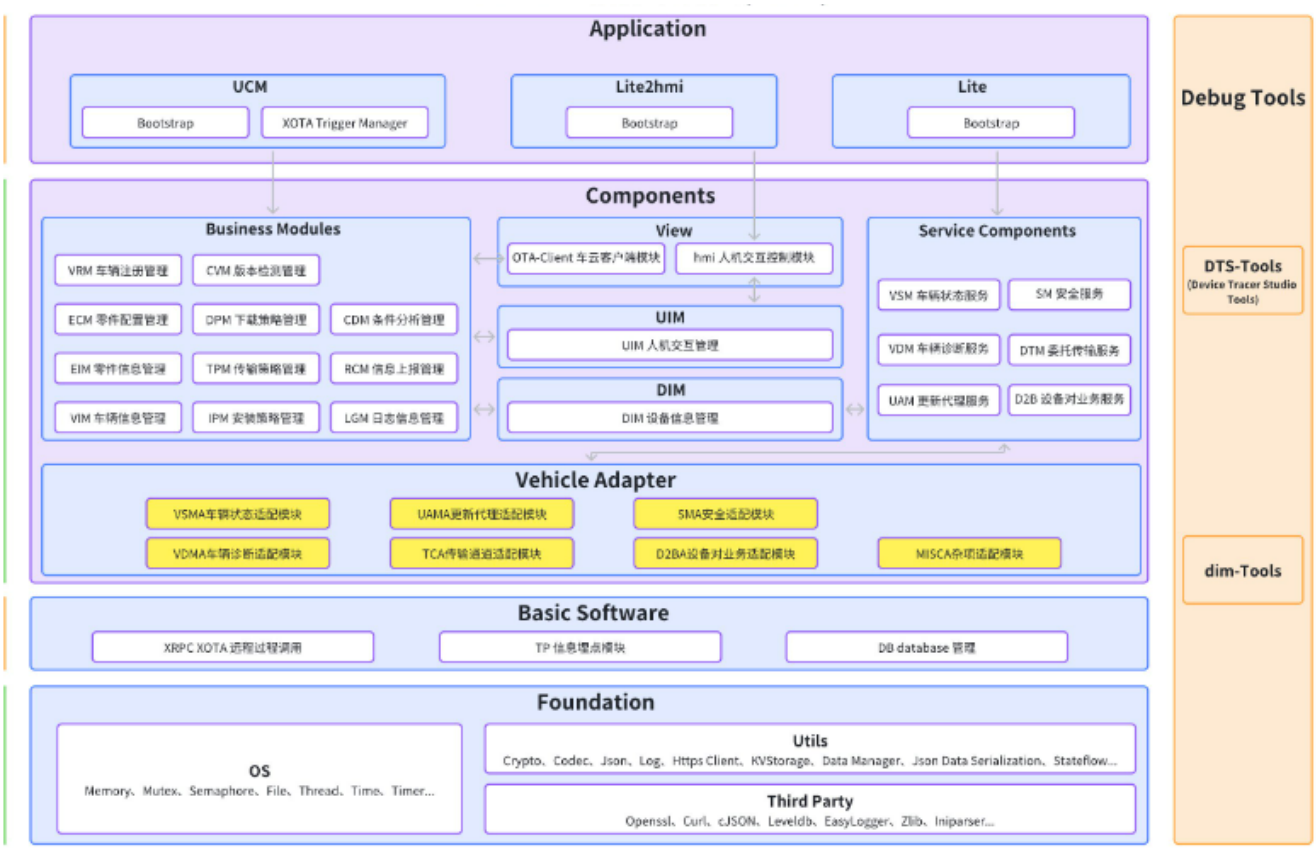
2.4 软件模块/组件设计 Software Module/Component Design（颗粒度L2/L3级）

2.4.1 模块/组件框图 Module / Component Diagram

展示内部模块（子模块）及模块间层次关系、依赖关系（L2/L3）

L3级 组件/模块 数据流/控制流关系 前端示例：

我们的设计目标是构建一个能够适应多种OTA业务集成场景的架构。我们期望该架构能够随着业务的增长沉淀出标准化的基础业务能力组件，这些组件通过统一的抽象和封装，使得上层应用能够像搭积木一样，通过组合相关组件快速集成对应的OTA场景应用。为了实现这一目标，我们的架构需要支持模块化、高扩展性，同时在项目构建时保持简单、灵活、易维护和可裁剪。



采用分层设计，将系统分为四层：

Application（应用层）： 此层旨在构建UCM应用程序，负责具体的OTA业务场景应用逻辑处理。它能够根据特定业务需求，调度下层组件或其他模块，以实现具体的OTA场景应用。

Components（组件层）： 此层提供OTA业务场景所需的功能组件实现，包括检测、下载、安装、人机交互、设备交互、车云交互等关键组件。

Basic Software（基础软件层）： 此层为XOTA提供坚实的基础软件平台，实现事件处理、进程间通信、信息埋点等基础软件的系统级功能。

Foundation（平台适配层）： 此层专注于提供系统的可移植性和跨平台能力，确保基于OTA系统的应用能够跨不同硬件平台运行，并与多种操作系统无缝集成。它通过抽象实现隐藏了底层细节，使开发人员能够专注于OTA应用程序的逻辑开发，无需过多关注底层平台的差异。此外，还提供了OTA常用工具的实现。

该方案中，我们增加了平台适配层，用于提供系统跨平台能力。同时模块间使用消息传递的方式实现模块之间的通信。模块可以发送消息给其他模块，并接收消息后做出相应的处理。这种方式可以有效降低模块之间的耦合度，提高系统的可扩展性。同时，对于后续模块的裁剪与扩展也比较友好。此外，消息通信机制允许系统在不重启的情况下动态地添加或移除模块，增强了系统的灵活性和适应性。

2.4.2 模块/组件列表 Module / Component List

描述每个内部模块的功能，并建立与具体的需求条目进行双向追溯。

ASPICE编号 (ASPICE_HLD_ID)	模块名称	功能描述
FVW_01、FVW_02、FVW_06、FVW_07、FVW_08、FVW_36、FVW_37、FVW_38、FVW_39	OTA CLient	OTA升级控制主应用，加载组件层中Business Modules，组织和驱动OTA车端行为，多以一个后台服务形式存在。
FVW_20、FVW_22、FVW_24、FVW_25	UIM(User Interaction Manager)	用户交互管理模块，用于管理人机交互
FVW_10、FVW_12、FVW_13、FVW_19	DPM(Download Policy Manager)	下载策略管理
FVW_22、FVW_27、FVW_28、FVW_31、FVW_30、FVW_29、FVW_32、FVW_33、FVW_34、FVW_35	IPM(Install Policy Manager)	安装策略管理

FVW_22、FVW_27、FVW_28、FVW_31、FVW_30、FVW_29、FVW_32、FVW_33、FVW_34、FVW_35	DIM(Device Interface Manager)	设备交互管理模块，用于管理需要被升级的零件
FVW_14、FVW_15、FVW_17	SM（Security Manager）	提供必要的安全措施，包括数据加密、身份验证和完整性校验，以保护Lite设备免受未经授权的访问和篡改。
FVW_22、FVW_27、FVW_28、FVW_31、FVW_30、FVW_29、FVW_32、FVW_33、FVW_34、FVW_35	D2B(Device to Business Manager)	需要宿主设备提供的一些接口来实现功能
FVW_22、FVW_27、FVW_28、FVW_31、FVW_30、FVW_29、FVW_32、FVW_33、FVW_34、FVW_35	UAM（Update Agent Manager）	在宿主设备是智能件时，负责升级刷写操作和部分零件信息的采集工作
FVW_22、FVW_27、FVW_28、FVW_31、FVW_30、FVW_29、FVW_32、FVW_33、FVW_34、FVW_35	VSM（Vehicle Status Manager）	在OTA过程中读取、设置和监控车辆状态，如车速、档位、电池电量等，以及设置整车的高压状态。
FVW_22、FVW_27、FVW_28、FVW_31、FVW_30、FVW_29、FVW_32、FVW_33、FVW_34、FVW_35	VDM（Vehicle Diagnostic Manager）	适用于非智能件的刷写及车辆诊断，以及执行部分零件信息的采集工作
FVW_03、FVW_04、FVW_05、FVW_22、FVW_27、FVW_28、FVW_31、FVW_30、FVW_29、FVW_32	VIEW	为可交互模块提供功能服务
	USB Upgrade	USB升级模块

2.4.3 备选方案

在这个方案中，将FOTA系统的开发分为三个主要层次，以提高系统的组织性、模块化和可维护性。下面是对该方案的详细描述：

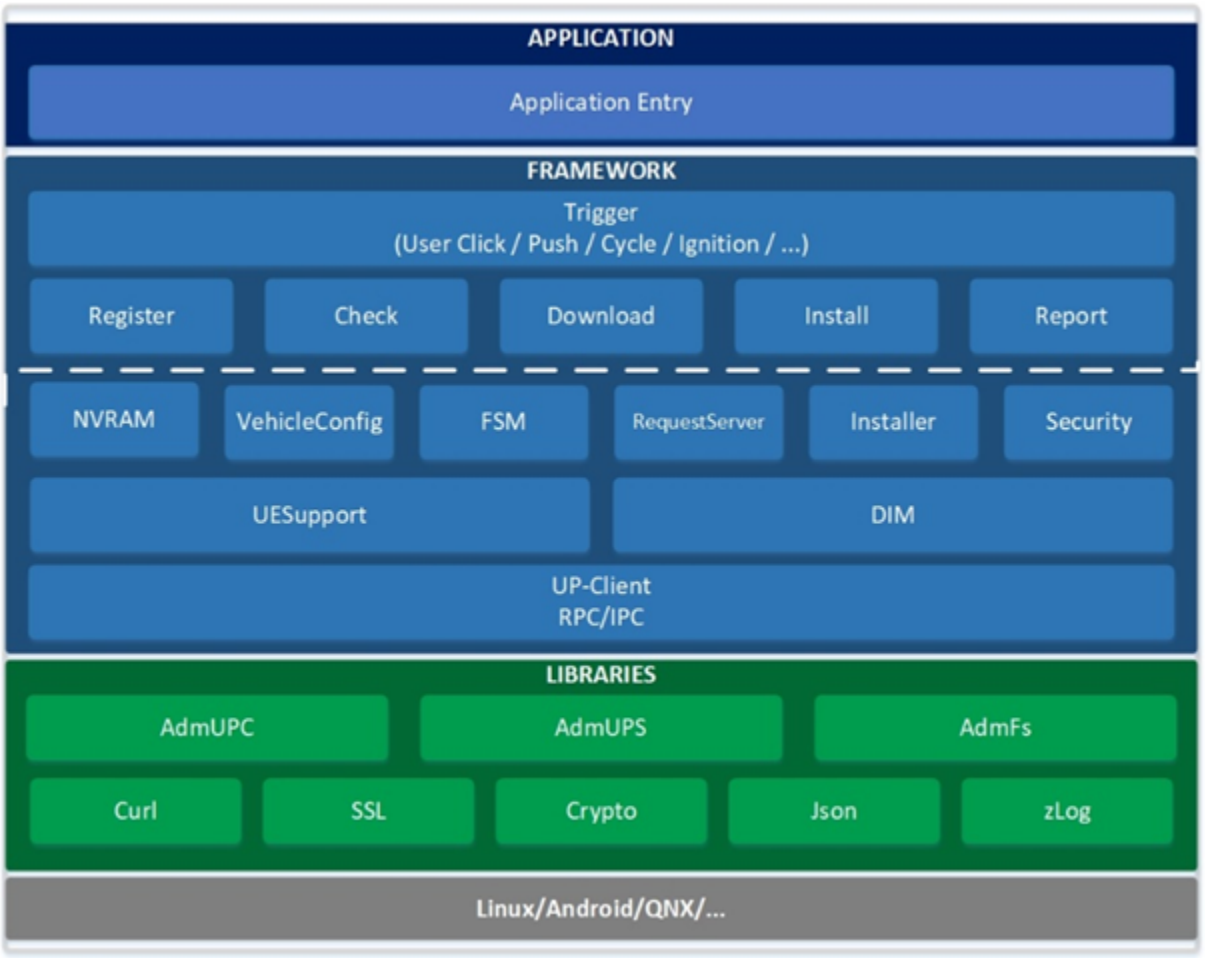
顶层 - Application（应用层）： 作为UCM的程序入口点。主要职责是初始化OTA系统所需的资源，为系统的运行提供基础环境。

中间层 - Framework（框架层）： 构成UCM的核心，负责实现FOTA业务的具体需求。该层按照功能需求进一步划分为多个模块，每个模块都具有特定的职责。

底层 - Libraries（库层）： 提供应用程序依赖的常用组件库，如Curl（用于网络请求）、OPENSSL（用于加密和安全传输）、Json（用于JSON数据的解析和生成）、Mqtt（用于轻量级的消息传输）等。

在这种分层架构中，Framework层的各模块通过定义清晰的接口来实现数据传递和交互。接口包括输入参数、输出参数和可能的异常情况。模块间的通信遵循这些接口规范，确保了代码的清晰性和模块间的松耦合，从而提高了整个系统的灵活性和可维护性。此外，UCM与各从控设备之间的通信采用MQTT协议，这是一种适合物联网环境的轻量级消息协议，支持低带宽、高延迟或不可靠的网络环境，适合进程间通信。

这种分层架构的优势在于其高度模块化的设计，它将系统划分为独立且可重用的组件，并通过定义清晰的接口，极大地简化了系统的维护和更新过程。



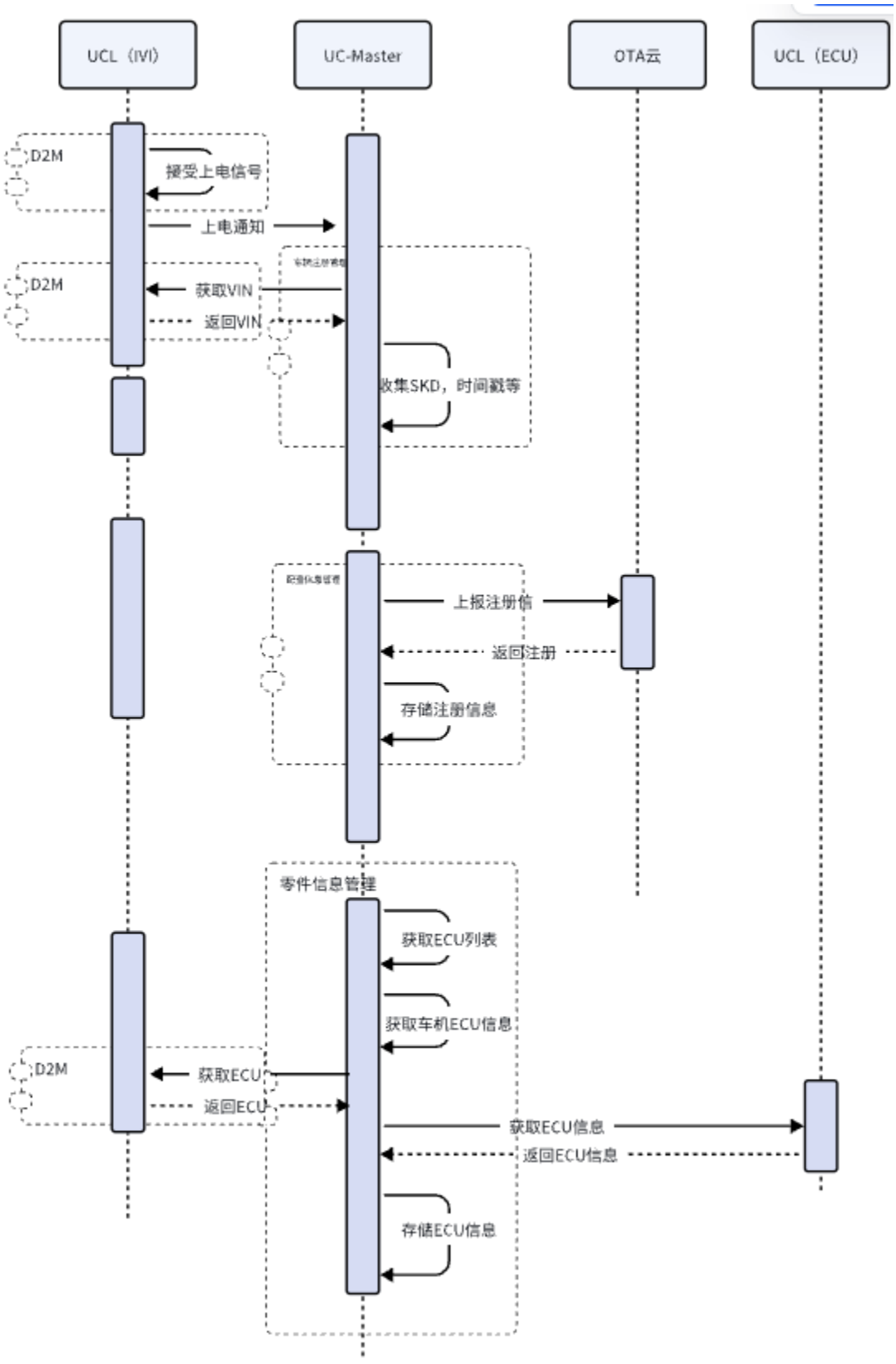
3 关键用例时序 Key Use-Case Sequence（需求/产品规格说明书中，有助于设计的关键场景用例分析，必选）

3.1. 获取配置和资产上报流程

3.1.1. 描述

通过获取车辆配置信息，可以实现两个主要目标：首先，用户可以利用OTA管理平台对车辆的OTA客户端软件相关输出进行控制，例如调整车端OTA软件日志文件的上报周期和输出等级，以优化车辆的性能和用户体验。其次，通过提前获取待升级的ECU列表，车辆可以预先采集对应的ECU DID值，从而在进行版本检测时大幅缩短用户体验时间，提升整体的升级效率。

3.1.2. 示意图



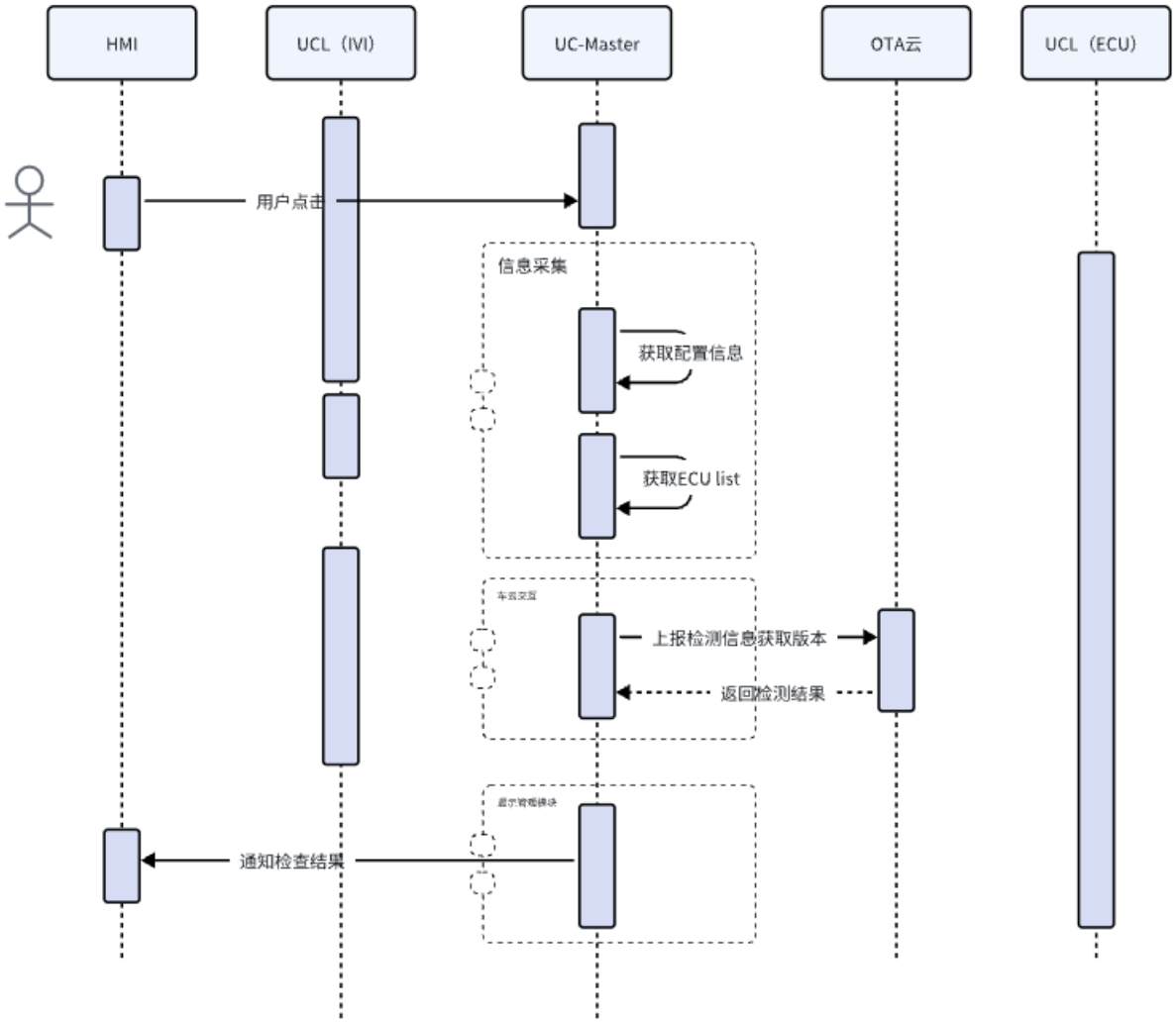
3.2. 手动检测

3.2.1. 描述

版本检测流程是指汽车端上报ECU详细信息，OTA管理平台据此判断ECU版本有是否需要更新的过程。版本检测流程由UCM发起，OTA管理平台响应。

版本检测管理模块主要负责车端OTA版本检测的整体流程，包括获取零件配置、读取待升级零件信息、向OTA管理平台发送版本检测请求、处理新版本策略如下载HMI策略文件等。

3.2.2. 示意图



3.3. 新版本推送

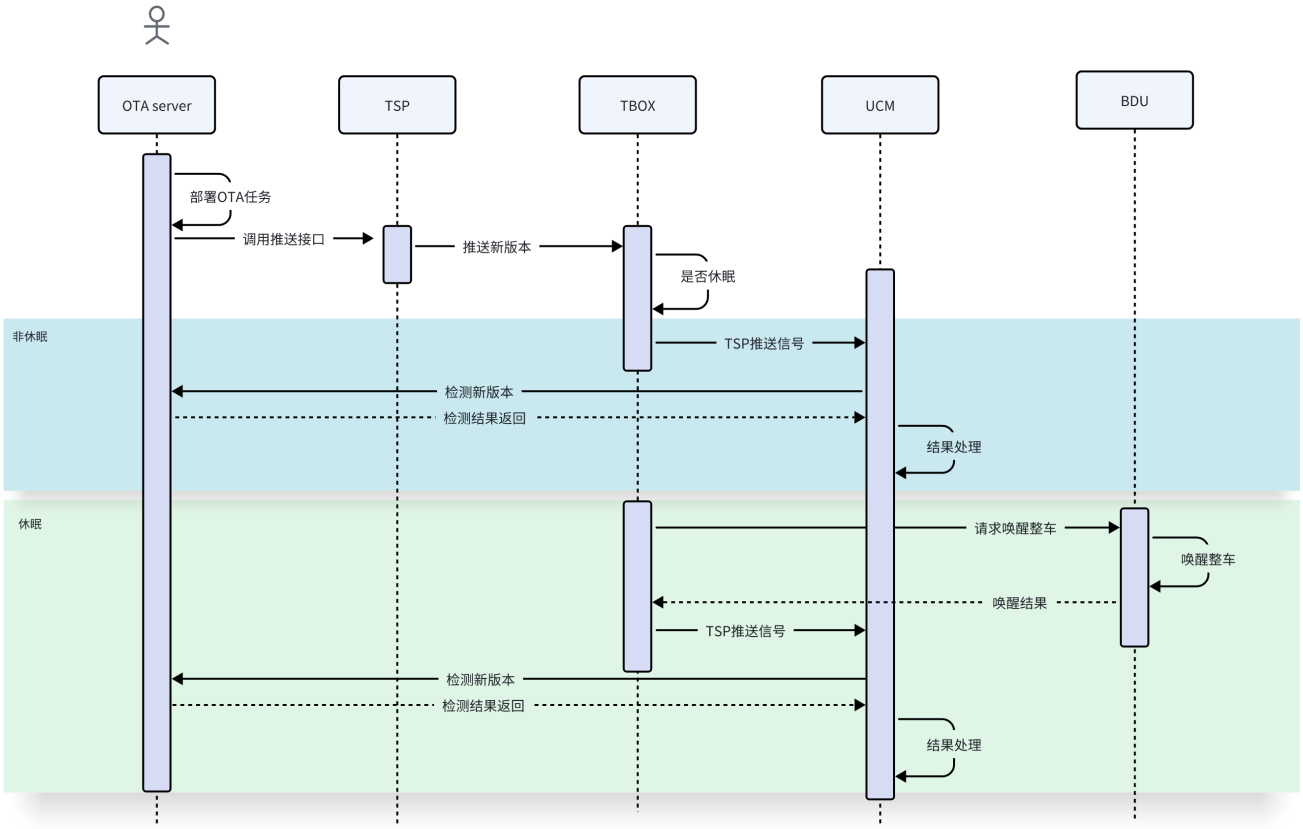
3.3.1. 描述

OTA云端部署任务，部署完成后请求TSP下发推送指令到tbox，tbox判断当前处理状态是否为休眠状态

非休眠：此时整车设备处于运行状态，OTA系统在运行中，可接收TSP推送信号触发检测流程获取任务信息，该状态下用户正在用车，只提示不弹框，上电信号或者下电信号时弹框提示用户。

休眠：此时tbox和BDU处于休眠态，其他零件处于未运行状态。首先要唤醒整车，唤醒成功后OTA系统处于运行中。可接收TSP推送信号触发检测流程获取任务信息，根据任务类型进行下一步操作。

3.3.2. 示意图



3.4. 新版本推送

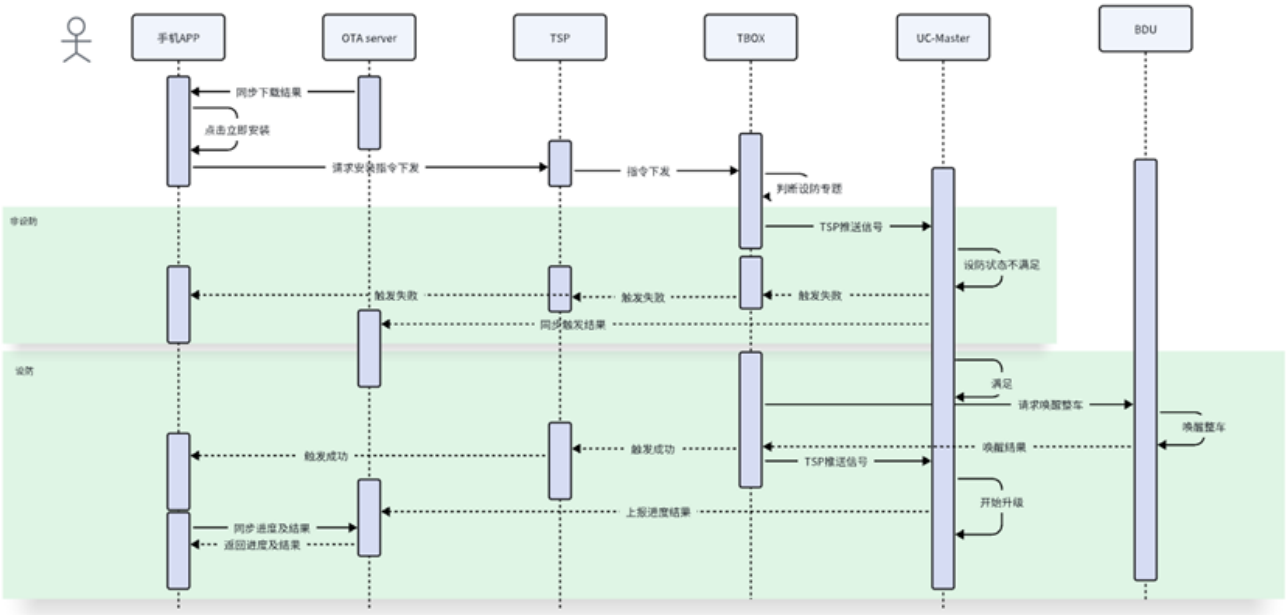
3.4.1. 描述

手机APP运行时向OTA云获取当前车辆状态如果是下载完成可以触发立即安装，手机APP调用tsp推送立即安装指令到tbox判断设防状态

非设防状态：当前条件不满足远程控制升级返回触发失败

设防状态：满足远程升级必要条件，tbox请求BDU唤醒整车，tbox接收到唤醒结果，同步结果到tsp，如果结果为成功，tbox发送信号给UM-Master进行tsp推送升级，手机APP向OTA云同步升级进度及结果；

3.4.2. 示意图



3.3. 远程修改预约升级时间

3.3.1. 描述

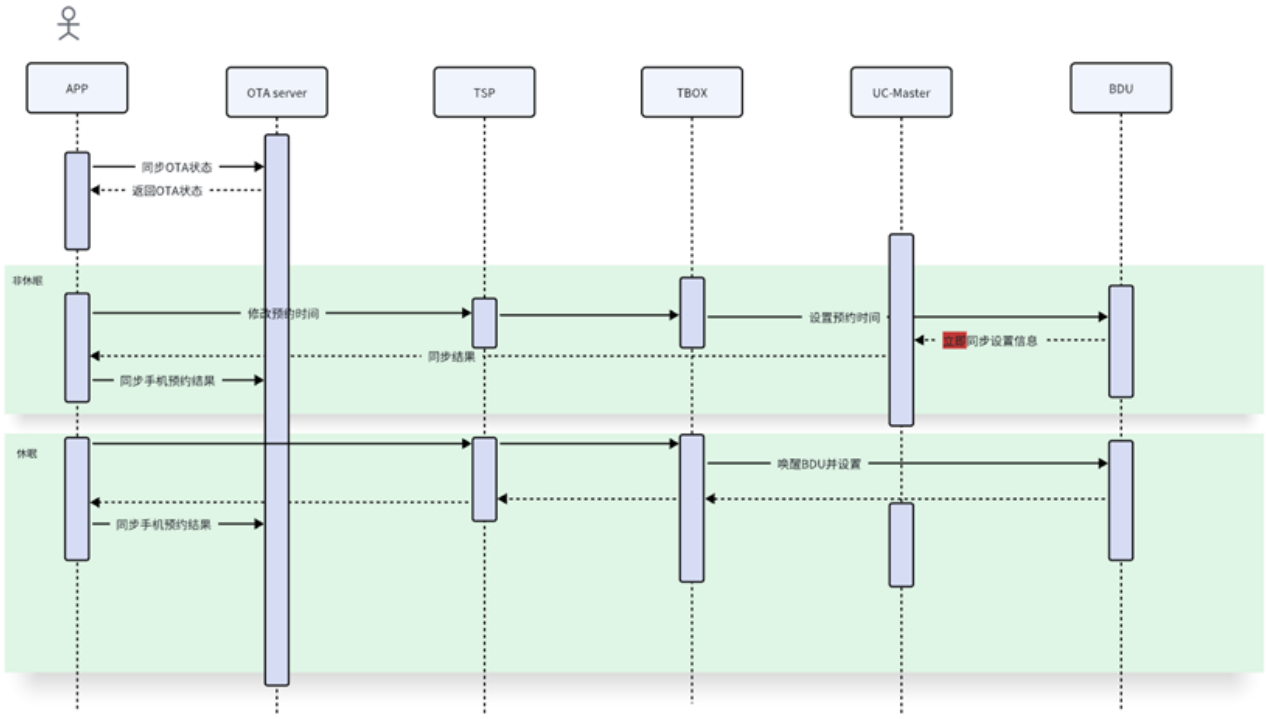
手机APP运行时向OTA云获取当前车辆状态如果是下载完成或者已经设置预约，手机app和执行修改预约时间推送

手机APP调用tsp推送修改预约时间指令

休眠：tbox唤醒BDU，设置预约时间返回设置结果，通过TSP同步到app用于显示，同时将结果通过TSP同步到OTA云

非休眠：tbox设置预约时间BDU返回设置结果，通过TSP同步到app用于显示，同时将结果通过TSP同步到OTA云

3.3.2. 示意图



4 数据库/数据结构设计 Database/Data Structure Design

设计数据库表结构或者数据结构，数据库请用ER图表示

数据库/数据结构兼容性设计

5 非功能需求设计 Non-functional Requirement Design

5.1 性能 Performance

- 5.1.1 系统资源评估

Function	CPU (DMIPS)			DRAM (MB)			GPU (GFLOPS)		FLASH (MB)	
	Standby	Background	Foreground	Standby	Background	Foreground	Background	Foreground	APK/BIN Size	Data Size
Media										

- 5.1.2 系统性能评估

需要来源	需求	性能需求	技术关键点
MOSI 技术任务书-J01项目座舱软件 V5.0.doc	USB插入后3000ms内提供USB音频播放	USB设备自动播放	1. USB插入检测； 2. USB音频扫描策略；

5.2 稳定性 Stability

- 5.2.1 应用/服务启动 & 重启监控设计
- 5.2.2 关键资源访问设计

TZ：访问场景，频率，etc

ShareMem：访问场景，频率，etc

I/O, HW, etc.

- 5.2.3 通信设计

通信协议：TCP/IP, HTTPs, SOMEIP?

通信需求评估：当次数据量，通信频率

- 5.2.4 稳定性测试设计

针对以上关键稳定性分析，设计本模块稳定性专项自测试用例

5.4 安全 Security

以下是本软件需要实现的安全需求及对应设计：

需求编号	技术要求	概要设计	ASPICE_HLD_ID
SEC_DOC_01	安全概念文档，分解到各个安全技术要求的具体实现方案中	分解到各技术要求的落地方案	
SEC_DOC_02	APP全生命周期的保密性、完整性、可用性、认证性方案设计；	1. 本地服务和灰彩服务均通过Gerrit平台实现V2签名。 2.要求第三方上架到APPSTORE的应用具备至少V2版本及以上的签名方案。	

SEC_DOC_03	B样提供第三方源码扫描报告，需要包括：组件名称、供应商名称、许可证类型、使用版本、最新版本、官方连接；并符合所有第三方软件的所有许可条款；同时，对其所涵盖的源代码的任何修改都应进行声明，并以源代码形式供公众审查。	"1. 开发提供OSS列表，包含这些信息： 组件名称、供应商名称、许可证类型、使用版本、最新版本、官方连接。"	
SEC_DOC_04	C样提供静态代码扫描报告；禁止集成不可访问代码或死代码。	集成到流水线；每两周扫描1次，开发人员对有问题的代码进行修复，确保C样之前扫描报告符合要求。持续扫描到SOP发布。【7月下旬工具到位】	
SEC_DOC_05	APP渗透测试报告【提供内部标准】	由德赛负责APP渗透，并输出报告。	
SEC_DOC_06	OSS漏洞修复报告；确保不存在已公布6个月以上未解决的漏洞	"1. 漏洞管理流程由平台安全负责设计并输出对应的文档，前端安全配合实施 2. 开发提供OSS列表，由德赛负责扫描并出报告，并由开发负责修复漏洞"	
SEC_CODE_01	C/C++安全编码规范：CERT C/C++ ；JAVA安全编码规范：SEI CERT Oracle Coding Standard for java，SEI CERT Android Secure Coding Standard, OWASP	"1. 开发人员参考编码规范编写程序，减少代码不规范问题。 规范文档连接：https://wiki.sei.cmu.edu/confluence/ 2. 修复扫描出来的代码问题"	
SEC_CODE_02	禁止将口令、私钥、密钥、证书硬编码方式写入代码以及配置文件，资源文件等； 禁止在代码中预留后门端口或接口，规避一切数据泄密风险； 禁止私自集成连接第三方非授权后端的SDK； 禁止将监听端口以及后端地址硬编码写入代码，应写入配置文件； 禁止自研APP热更新SDK； 针对上传到AppStore的第三方应用，应通过合同约束禁止热更新SDK（责任人待定）；	"1. 开人员遵循安全开发规则 2. 修复扫描出来的问题"	
SEC_CODE_03	使用VW 80180_1标准定义的加密算法；不同用途的密钥必须不同	1. 远程通信加密，各模块开发人员采用安全开发提供的TLS参数配置进行开发； 2. 本地加密，宜采用车机证书管理系统提供的加解密接口进行加解密；	
SEC_APP_01	注：本条款适合于作为独立进程运行的模块，不适合供调用的LIB。 提供运行时资源清单: 运行需要的Android定义的标准Permission清单 运行需要的自定义Permission清单 运行时需要监听的服务端口清单 运行时需要访问的后端服务地址（或名称）清单 运行时需要跨进程访问的文件资源清单 运行时可以共享给其他进程访问的文件资源清单	按要求及模板输出《权限及资源清单》，文档模板链接:权限及资源清单模板	
SEC_APP_03	重要参数安全：本进程使用的参数（如车控车设配置）文件只能供本进程访问；若需共享，则需要提供可以共享给其他进程访问的文件资源清单	各个模块的配置文件应参考SEC_APP_01文件清单关于文件权限控制的定义实现； 除此之外关于Permission, 监听端口，服务端口相关配置也需要遵照SEC_APP_01执行。	
SEC_APP_04	敏感信息处理规则： 1. 进程间通信内容：避免明文传递隐私数据 2. 进程间通信接口：避免提供敏感功能接口 3. 隐私数据缓存安全:程序退出时删除缓存的敏感数据文件 4. 隐私数据显示安全：脱敏显示隐私数据；禁止隐私界面被录屏或截屏；口令输入时需要提醒用户“注意避免口令泄露”	1. 进程间通信内容：避免明文传递隐私数据 实施方案：进程间若存在隐私输出传递，需要对隐私数据做加密处理 2. 进程间通信接口：对通信接口进行加密后调用 实施方案：如需要，在《权限及资源清单》的permission中说明 3. 隐私数据缓存安全:程序退出时删除缓存的敏感数据文件 实施方案：避免缓存隐私数据，如果业务需要，必须缓存，则可以考虑脱敏或加密后缓存 4. 隐私数据显示安全：脱敏显示隐私数据；禁止隐私界面被录屏或截屏；口令输入时需要提醒用户“注意避免口令泄露” 实施方案：HMI显示隐私数据时，需要参考加密脱敏策略文档，如针对手机号码进行局部*号代替，针对密码用全*号代替等 涉及以上敏感信息处理方式之一的模块，需要遵循安全开发规则，具体实施方案可参考上面描述。	
SEC_APP_06	所有APK都需要V2签名	本地服务和灰彩服务均通过Gerrit平台实现V2签名。	

SEC_APP_07	发布版本安全: 1. IDE编译选项禁用调试功能; 2. IDE编译选项禁用AllowBackup和Debuggable; 3. IDE编译选项打开代码混淆功能; 4. 代码中禁用调试信息;	1. 针对Android APK以及组件库，对应的AndroidManifest.xml，需要设置android:debuggable=""false"", AllowBackup也需要禁用。 2.代码混淆的开关在，build.gradle文件做设置。 minifyEnabled true 针对代码中禁用调试信息，需要在代码中进行自检，删除或通过开关控制调试信息。	
SEC_APP_08	APP加固	采用第三方加固工具对选定的APK进行加固; APK加固后进行全功能测试，避免加固引起入异常。	
SEC_CERT_01	1. 数字身份应以证书的形式存储在载体中，证书的使用应符合国家密码管理要求。 2. 应采用具有存储密钥、数字身份并具有授权访问机制和数据加密功能的载体（安全芯片）。 3. 车辆与外部系统间通信始终是保密的，禁止使用Null的密码套件。 4. 应使用OEM信任的证书。 5. 如果证书验证失败，则链接的建立必须被拒绝，并且返回失败原因。 6. 加解密、签名验签需要在安全域进行 7. 不允许硬编码在应用程序的源代码中来实现身份认证。 8. 客户端证书写入嵌入式身份载体时应核验身份编号与汽车身份标识的汽车身份编号一致性 9. 客户端证书有效期应规定在一定范围内，宜具备客户端证书在线更新功能（具备证书状态检查、提前更新、已过期后更新能力） 10. 客户端证书应由系统层级统一管理使用，应提供标准接口或其他方式供在线应用完成身份认证，证书管理与使用方式应与客户达成统一意见后方可采用 11.车机端根CA证书文件应只包含一级根（顶级签发机构）内容，不应包含中间证书及叶子证书，车机中的CA证书文件应由客户评估后方可集成。 12. CA证书应由系统层级统一管理使用，应提供标准接口或其他方式供在线应用完成认证。根证书的使用不应指定单一证书文件，应采用证书集或信任列表的形式完成认证，CA证书管理与使用方式应与客户达成统一意见方可采用 13. 根证书文件应具备在线更新能力，在相关业务根证书到期或服务端证书签发机构变更时提前完成更新; 14. 在线更新与云端通信应采用专用的安全通道，该通道应确保长期可用（汽车生命周期内可用），并与业务通道进行区分。 15. 应确保系统中自带的TrustStore及其他不受控的CA证书内容为空，以防其他在线应用私自调用 16. 若涉及多系统，应确提供具体的多系统证书管理与应用的解决方案 17. 应使用OEM信任的客户端证书 18. 在线应用应使用系统层级提供的标准客户端证书调用方式完成身份认证。 19. 在线应用应按照客户要求节点提供其CA证书信任文件及认证方式说明（包括影响业务范围）。 20. 在线应用应使用系统层级提供标准的CA证书使用方式完成身份认证。 21. 如在线应用服务端的证书签发机构变更或即将到期，应按客户要求时间节点进行预警，并配合完成车机端CA证书的更新; 22. 在线应用不应使用系统中自带的TrustStore及其他不受控的CA证书文件。 23. 在线应用如无法使用系统提供的统一证书调用方式，应确保其自身可以完成证书的管理及应用（证书更新与安全性），保障业务连续，并提供说明供客户评估后方可采用。 24. 非对称密钥长度RSA应不低于3072bit，ECC/SM2应不低于256bit。 25. 任何密码或随机数算法都必须满足VW80180的要求。	1. 凡是与云端连接的应用，如灰彩服务，都需要采用统一的证书管理系统提供的证书 2. 需要本地加密存储的应用，宜采用证书管理系统提供的KMS功能接口，进行数据加解密，因为加解密过程发生在安全域，其密码保存在安全域，因此避免了自己保管密钥产生的安全隐患及违规事件。	
SEC_TLS_01	TLS软件版本，必须支持TLS V1.2及以上版本。 检查是否使用第三方SSL库，若采用第三方库需进行评审： 以下为部分三方库版本要求 1. Openssl 3.0 2. wolfSSL min. 3.12.2 3. gnuTLS min. 3.6.1 4. CycurTLS min. 2.0 5. botan min. 2.15	绝大多数非必要的情况下，灰彩服务均处于Android应用层，不会涉及到底层SSL的选择与控制，意味着SSL底层库属于系统内置，其版本由系统版本决定，并由德赛负责。如果某些应用必须使用非系统SSL库，则需要遵循技术要求；如果被审计后发现SSL使用不合规，需要修复。	
SEC_COM_01	向车外传输敏感个人信息和重要数据，应采用加密措施，加密算法应选择公开的、已发布的、有效的密码算法，选择适当长度和有效秘钥。	这里指远程通信，采用TLS双向认证	
SEC_LOG_04	感数据保护：禁止敏感数据写入日志，可以加密或脱敏方式写入	全体员工遵循本条规则 若采用加密存储，宜采用车机证书管理系统的KMS模块统一处理	

SEC_SensitiveFun_05	定位安全：具有交互界面的车载信息交互系统，在调用定位功能时，在用户明示同意后，才能执行定位功能；且需向用户提供后台定位控制功能以配置应用软件是否可调用定位功能；以上两种功能应让用户分别操作；具有交互界面的车载信息交互系统，在调用定位功能时，应在交互界面上给用户相应的状态提示	按照PRD文档设计进行开发 产品PRD文档连接：XXX	
SEC_ACT_01	禁止在开源代码社区（如github、gitee等）上传业务相关源代码。	全体员工遵循本条规则	
SEC_ACT_02	禁止将自身搭建的代码托管平台或代码相关的wiki平台（如Gitlab、Confluence）发布在互联网上	全体员工遵循本条规则	