

从域融合&Hypervisor出发思考 汽车操作系统的技术路线和发展策略

中瓴智行

2023-03



目录 CONTENTS

01

域融合现状与趋势

02

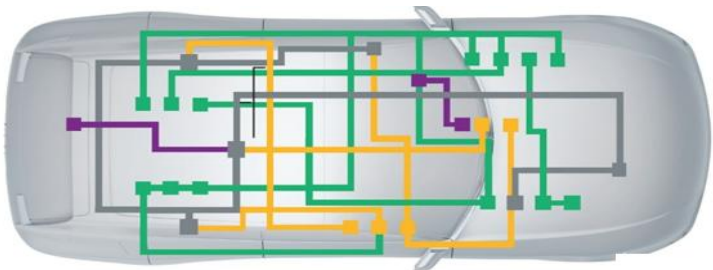
汽车操作系统的技术路线

03

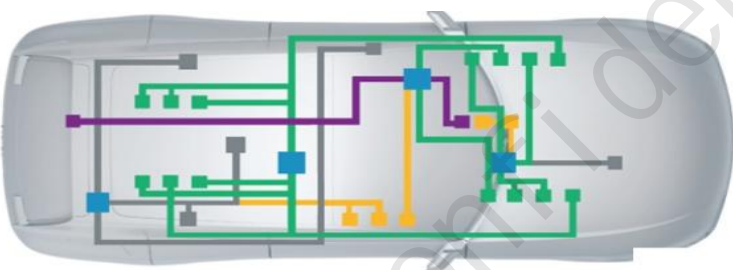
汽车操作系统的发展策略

智能网联汽车电子电气架构正加速向域融合方向演进

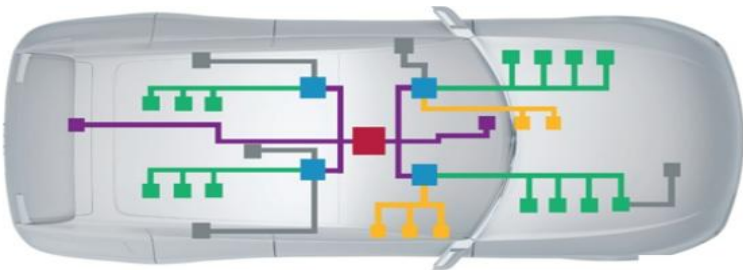
分布式架构



域控制器架构

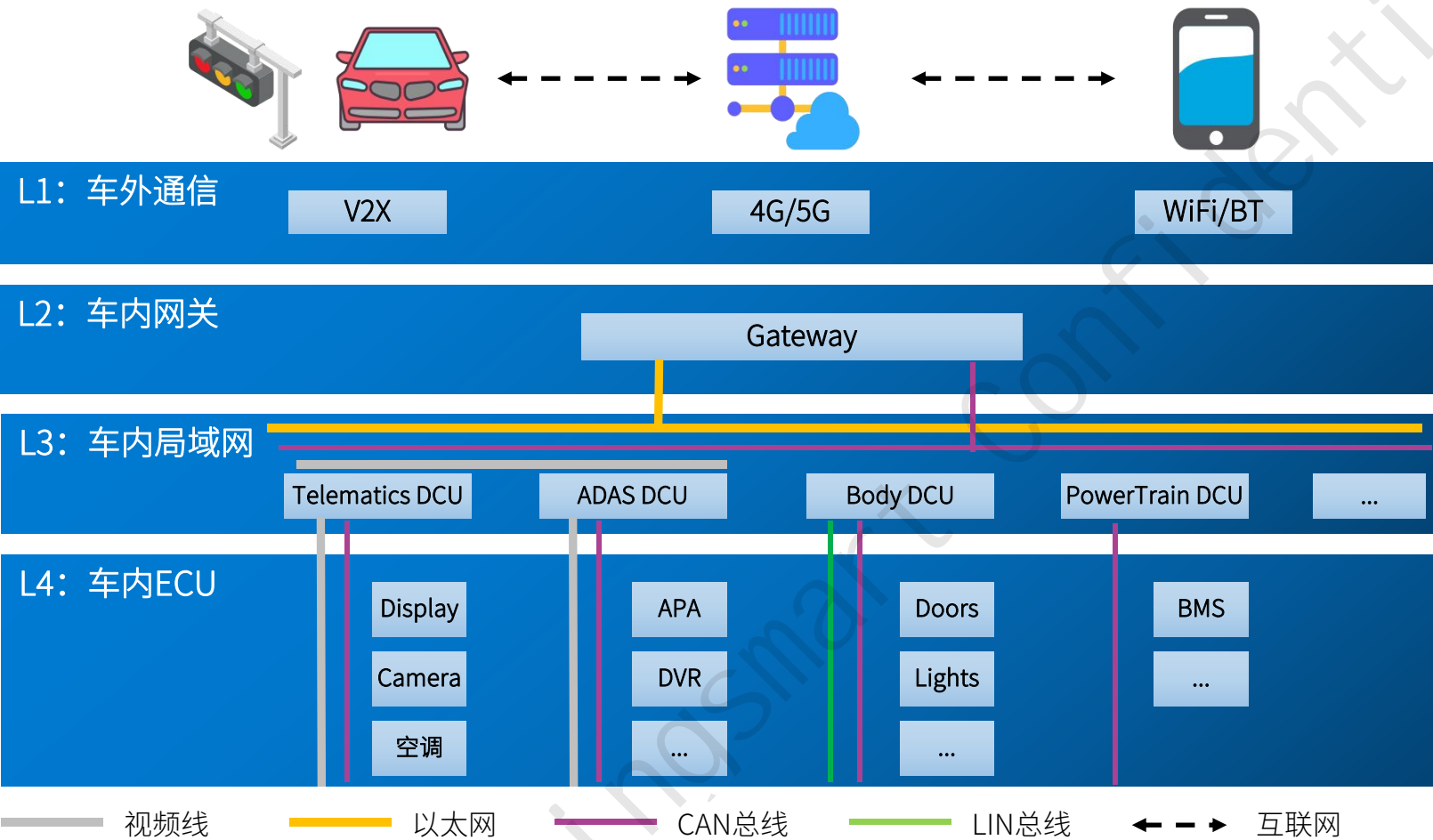


中央计算架构



智能汽车	分布式	域控制器（DCU）	中央计算（CCU）
ECU	Cluster、IVI、中控、网关、T-Box、行车记录仪、车身控制、辅助驾驶等 10+个ECU	座舱域控、车身域控、辅助驾驶、中央网关等典型域控制器	1个HPC和多个Zone控制器
传感器	每个ECU单独传感器，功能越多，需要的冗余传感器越多	传感器连接域控制器，可复用大部分传感器	传感器连接到HPC或Zone控制器，所有业务功能通过服务访问
整车线束	>6km	3~5km	<1km
骨干网络	CAN	CAN/Ethernet	Ethernet
通信方式	信号	信号或服务	服务

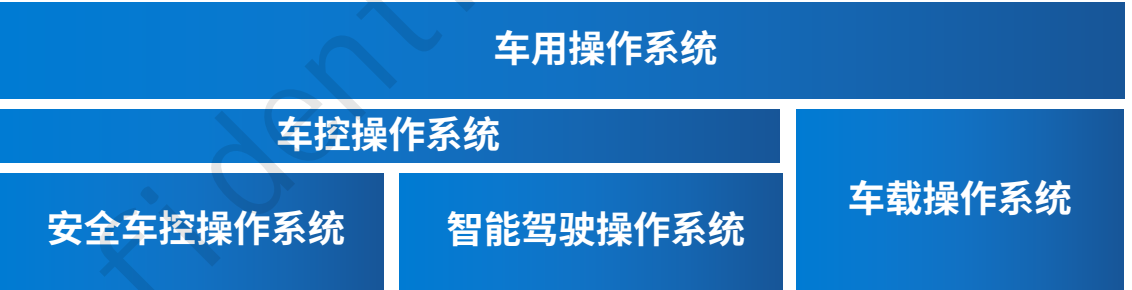
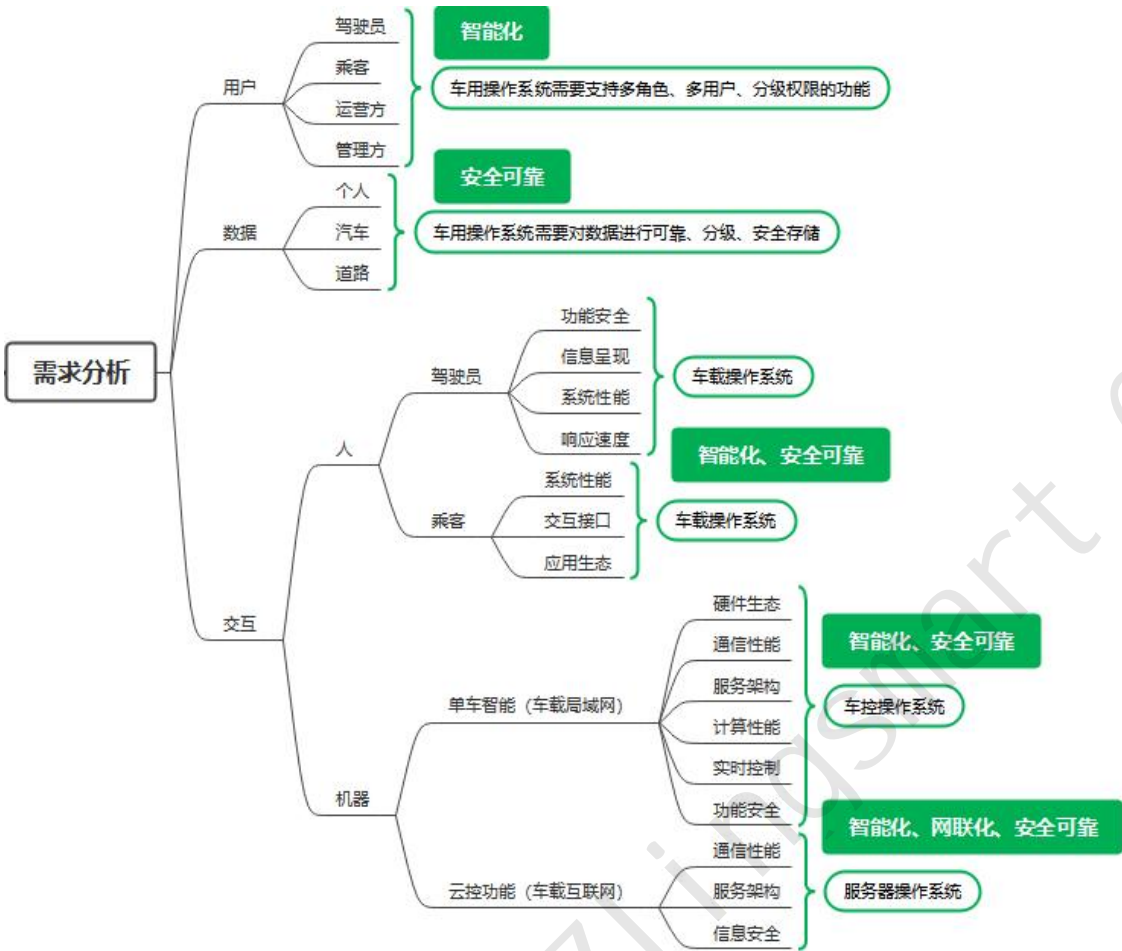
智能网联汽车是复杂网络系统和多网络节点的电子电气架构



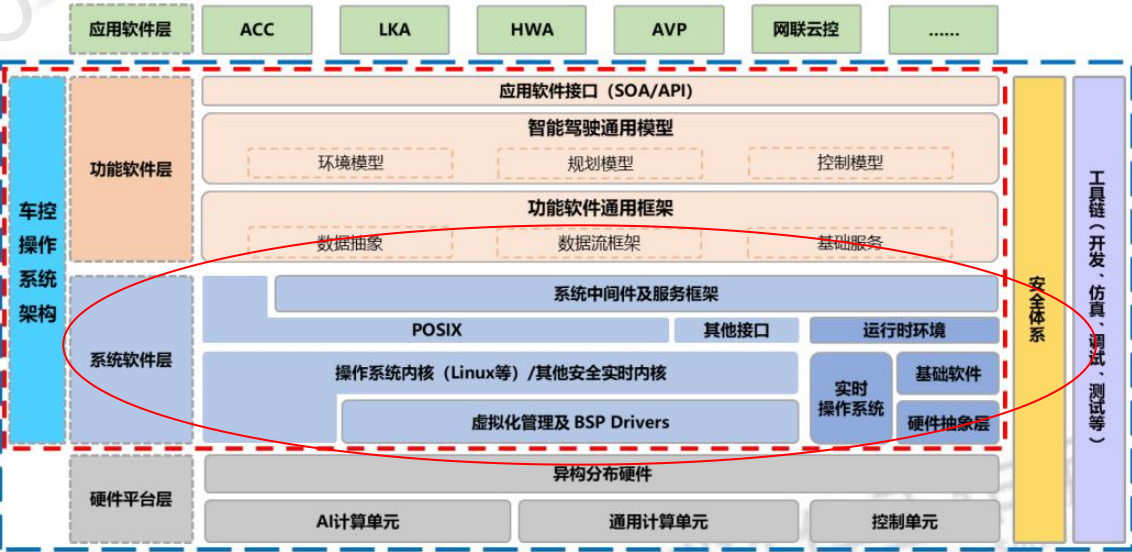
工业互联网参考模型

不同网络节点业务需求具有差异性，需要多个不同类型的操作系统来达成业务目标

智能网联汽车需多类型操作系统满足差异化的业务需求



来源：《车用操作系统标准体系》



来源：《车载智能计算基础平台SOA软件架构白皮书》

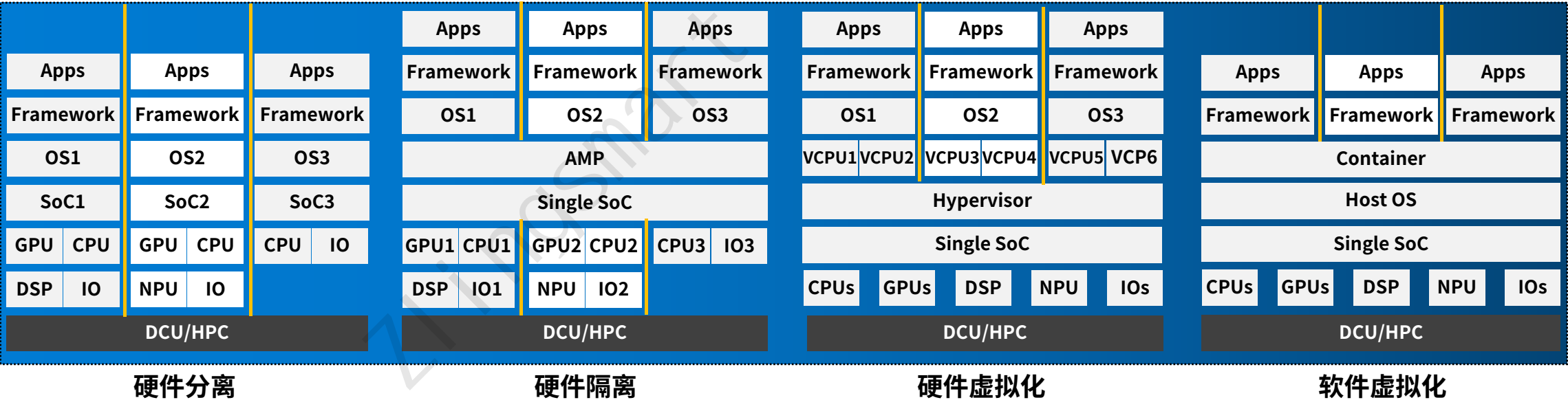
智能网联汽车域融合趋势下的多操作系统架构

■ 汽车EEA四种典型多系统架构

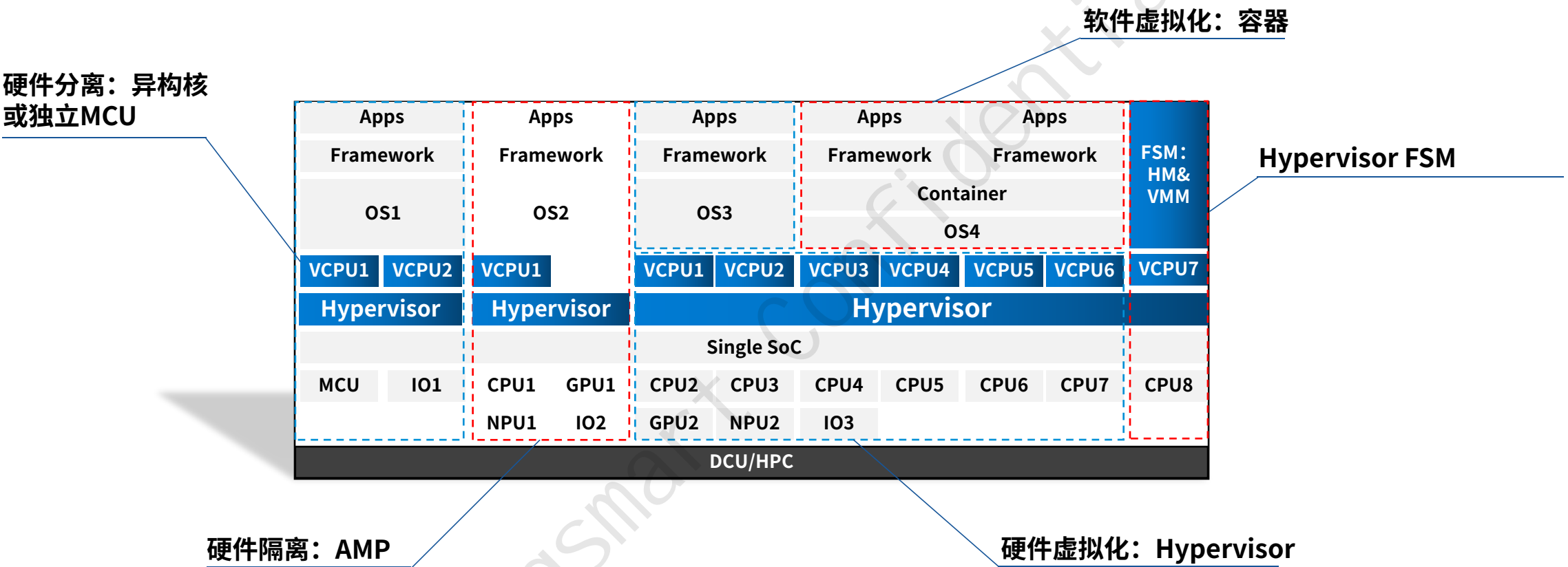
- ✓ 硬件分离（不同SoC）
- ✓ 硬件隔离（相同SoC、不同硬件资源）
- ✓ 硬件虚拟化（Hypervisor）
- ✓ 软件虚拟化（容器）

技术路线	功能安全	系统性能	实时性	软件平台化	配置灵活性	系统成本
硬件分离	高	高	高	中	低	高
硬件隔离	高	中	高	中	中	中
Hypervisor	高	高	高	高	高	中
容器	低	高	低	高	高	低

■ Hypervisor既有硬件分离和硬件隔离方案的安全性、系统性能和实时性优点，也具有容器方案的平台化、灵活配置和低成本特征



四种多系统架构短期并存，随MPU/MCU芯片的发展 Hypervisor会成为汽车域融合架构中最基础的底座



关键需求

1. 资源占用轻量化（ROM/RAM）
2. ISO 26262 ASIL D
3. 跨SOC/CPU/VM的快速IPC机制
4. MCU的支持
5. 多类型和多数量的虚拟机支持
6. 硬件设备的软件抽象

国内外主要Hypervisor方案和应用情况

《中国汽车基础软件白皮书V3.0》

2.4.4 典型应用案例

在汽车智能化发展历程中，虚拟化主要应用于智能座舱、智能驾驶、智能网关等融合场景。智能驾驶受技术成熟度、政策法规所限，基本处于预研、方案原型阶段。智能网关业务功能相对同构，并且有可能进一步融合到其他场景方案中。因此，目前主要的应用案例集中在智能座舱中。

智能座舱域融合也是在近几年启动，正在不断迭代演进中。受芯片算力、虚拟化技术成熟度、生态链对于虚拟化解决方案的掌控能力等因素影响，有些厂商同时采用了硬隔离方案来实现域融合，一方面最大程度地沿用既有技术能力，有确定性保障，但是缺少了软件定义的灵活性，智能化程度有限，是域融合的一种可选方案。在嵌入式虚拟化技术方面，国外的 QNX、OpenSynergy、PikeOS 等有先发优势，尤其在汽车领域已耕耘多年，因此在这两年涌现了较多的应用案例。在智能本土化发展的趋势带动下，国内这几年也出现了不少芯片厂商、独立软件厂商研发嵌入式虚拟化技术、产品、解决方案，如中瓴智行的 RAITE Hypervisor(RHOS)、中兴 GoldenOS、斑马智行的 AliOS Hypervisor、中汽创智 CAIC Hypervisor 等。

1. 智能座舱域控制器产品

某厂家智能座舱域控制器产品，如图 2.4-9 和图 2.4-10 所示，基于高通 8155、瑞萨 R-Car H3 处理器，采用 QNX Hypervisor，搭载 QNX Host、Android P/R/S Guest OS，可配置输出最多 6 块高清大屏独立显示，集成了娱乐系统、液晶仪表、车身控制、DMS、APA 等功能，支持独立四音区、多屏互动和音视频分享，集成度高，在长城、长安、宇通客车等多款车型上适配量产。

另外，国产化方案芯驰 X9HP+ 平台，采用硬分区、Hypervisor 两种方案灵活配置实现中低端智能座舱域控制器产品。



国外案例：QNX Hypervisor
(高通/瑞萨/芯驰/...)

2. RHOS 智能座舱域控制器平台

(1) NXP I.MX8QM 座舱域控制器

某厂家基于自研的 Type-1 型虚拟化软件 RHOS(Raite Hypervisor OS)，适配支持了 NXP I.MX8QM，提供一个轻量、灵活的汽车智能座舱虚拟化解决方案，已在东风车型量产上市。其系统架构如图 2.4-11 所示：



图 2.4-11 NXP I.MX8智能座舱系统架构

在 SoC 上运行 Hypervisor 后可支持同时运行多个操作系统，比如 Linux 系统可以运行实时性和安全性较高的业务，如全液晶仪表等，可以扩展运行 DMS、HUD 等业务。另外一个虚拟机运行 Android 操作系统，上面部署信息娱乐等安全性和实时性要求较低的业务。为保证系统具备良好的市场竞争力，域控制器兼容 TBOX 功能需求，系统能够支持休眠唤醒和快速启动。

Linux 和 Android 虚拟机可按需进行资源的配置，包括内存、CPU、存储空间、外设等。该架构支持系统升级，包括对虚拟机和 Hypervisor 的升级，支持异常日志记录，包括虚拟机内核和 Hypervisor 日志。

多屏交互是智能座舱重要的应用场景，Android 的 APP 应用程序可以通过 Hypervisor 推送到 Linux 仪表进行显示。

国内案例：RAITE Hypervisor
(MTK/NXP/瑞萨/瑞芯微/芯驰/...)

目录 CONTENTS

01

域融合现状与趋势

02

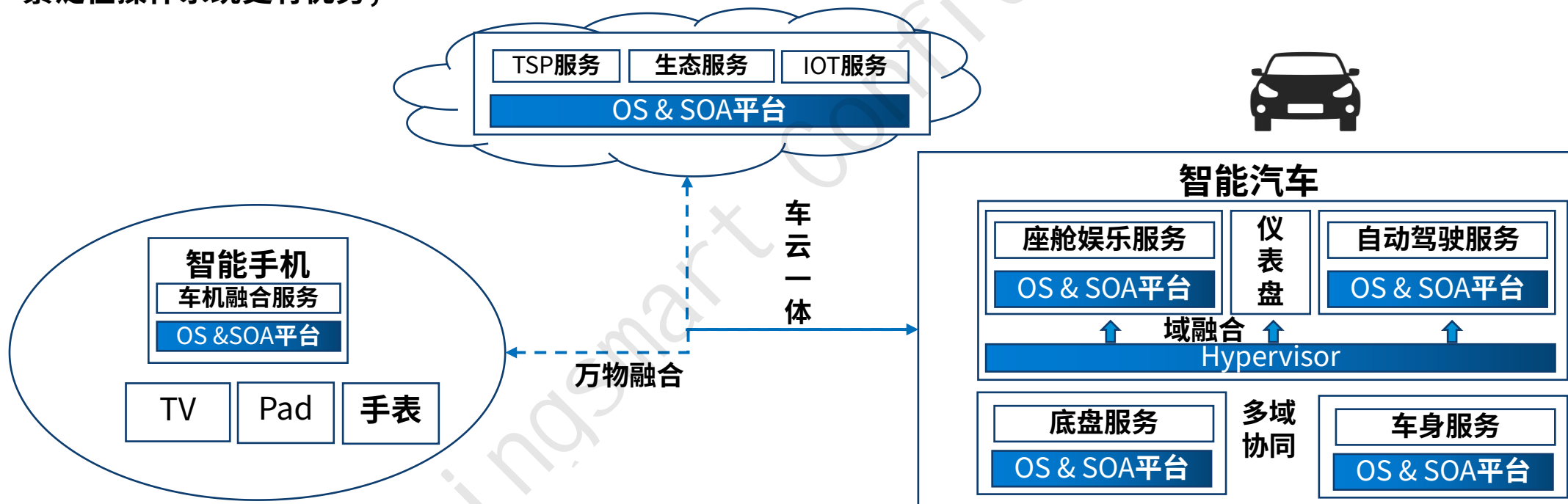
汽车操作系统的技术路线

03

汽车操作系统的发展策略

汽车操作系统特性趋势 – 智能化、网联化

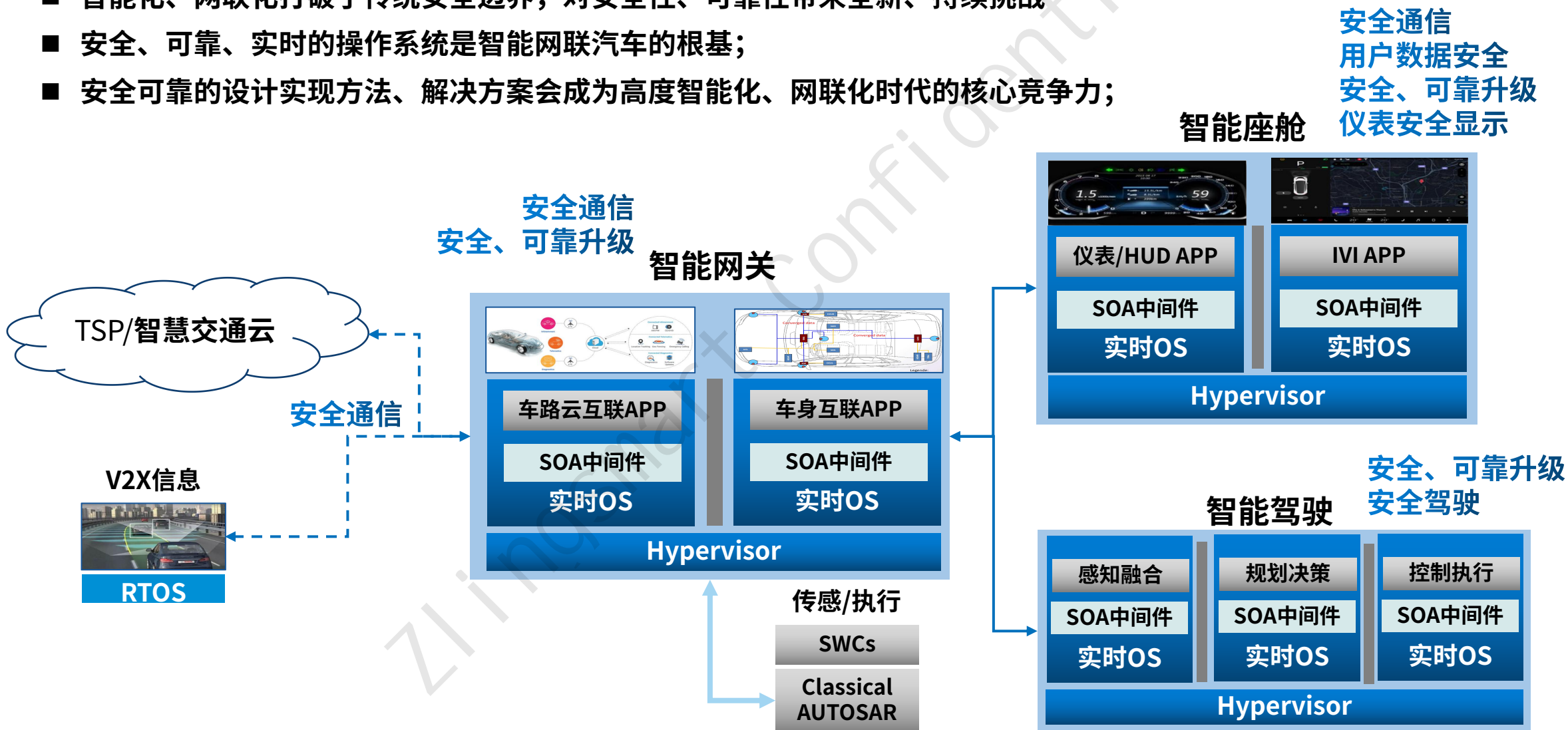
- ICT技术发展使能一分多(域融合、复合型设备)、多合一(网联化)，需要功能更强大的操作系统支撑业务场景；
- 虚拟化是实现一分多的首选支撑技术；
- 短期看，多合一可在现有多种异构OS上演进SOA平台实现融合；长期看，原生服务化、可扩展、统一架构规范的全场景泛在操作系统更有优势；



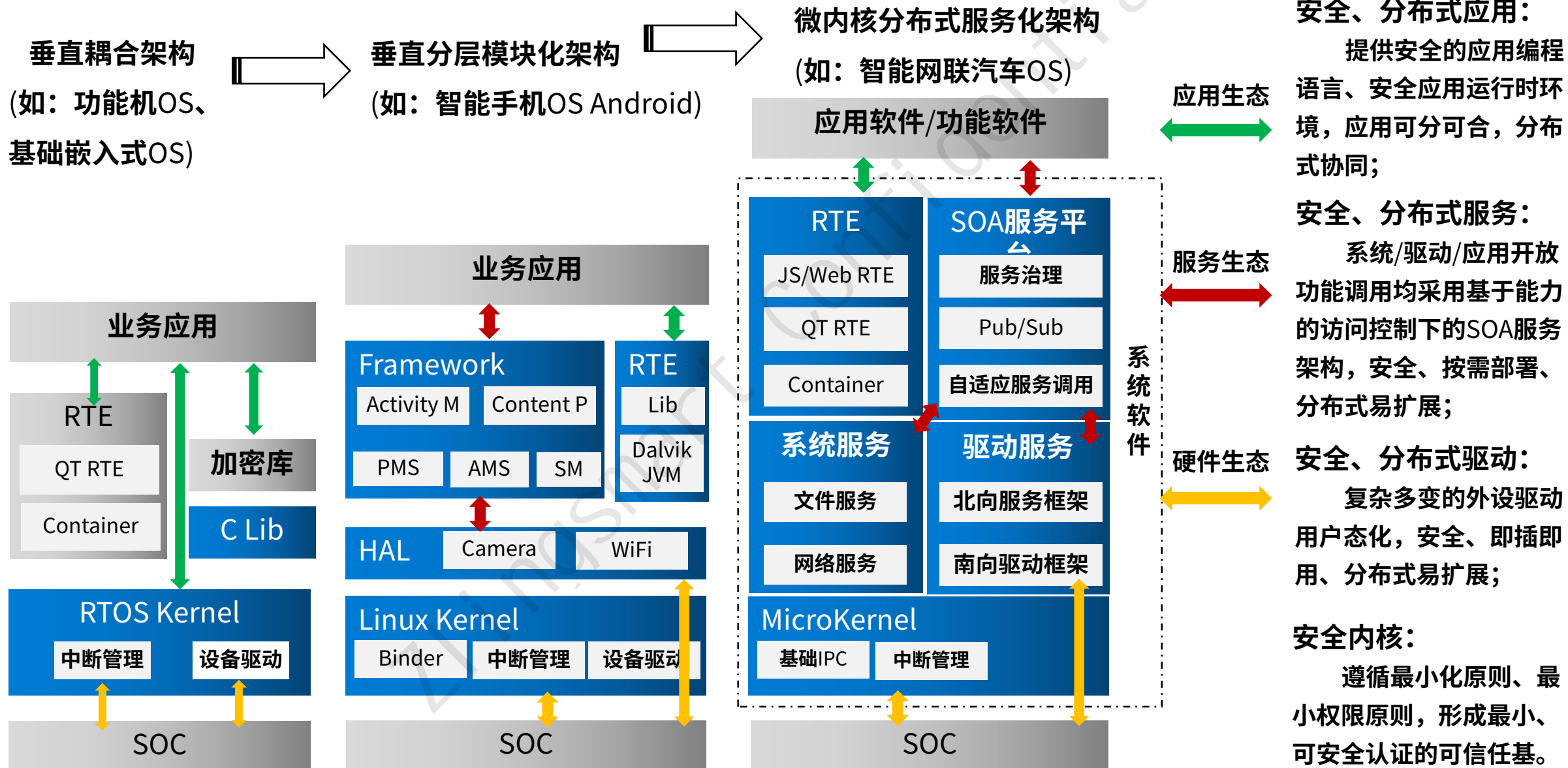
- 智能化：单设备既可完成自有特色功能，也可以软件定义成复合型设备，完成原先多个设备的功能；
- 网联化：多设备通过SOA实现算力、外设、数据、服务、应用融合，相互赋能、各展所长，实现超能系统、最佳体验；

汽车操作系统特性趋势 – 安全、可靠

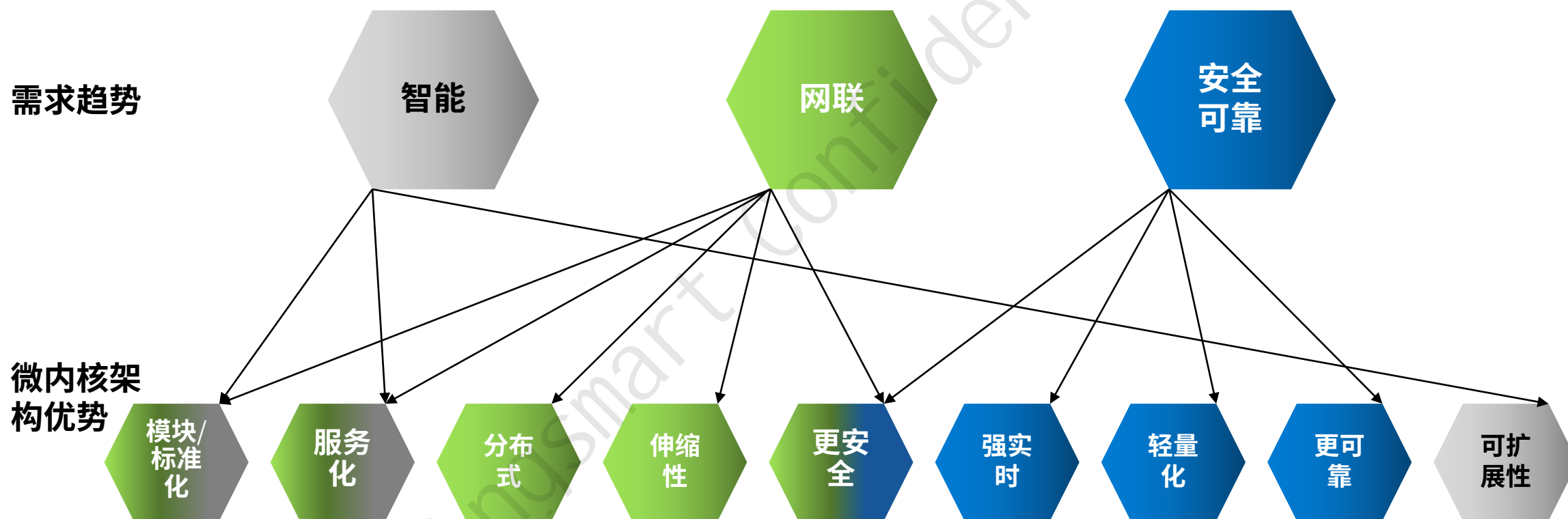
- 智能化、网联化打破了传统安全边界，对安全性、可靠性带来全新、持续挑战
- 安全、可靠、实时的操作系统是智能网联汽车的根基；
- 安全可靠的设计实现方法、解决方案会成为高度智能化、网联化时代的核心竞争力；



汽车操作系统由满足功能性需求向智能网联、安全可靠演进



微内核架构理念符合操作系统发展趋势，原生满足智能、网联、安全可靠发展需求



典型OS架构对比分析

OS架构	特权模式	内核	驱动	应用	服务化扩展	访问控制	安全开发/认证	典型OS
传统RTOS	通常不区分，安全性差	通常为平板式内核	与内核同地址空间，直接调用，相互影响	地址空间不隔离，相互影响。可利用MPU实现空间隔离。	无	无	代码量小，方便认证。但应用/驱动集成后使认证无效。	FreeRTOS
宏内核OS	宏内核以特权模式运行其整个代码，增加了发生灾难的概率。	系统功能集中于内核，庞杂、攻击面大	主体处于内核中，驱动的不稳定影响内核安全。Android有重构改进。	应用与内核分离、应用与应用分离，系统调用较多，有安全威胁	标准Linux无。Android扩展了部分机制。	DAC、MAC、RBAC、ABAC	分散开源开发，难以遵循安全流程。难以做到高等级安全。未达到ASIL-B。	Linux, VxWorks
微内核OS	充分利用硬件特权级别(EL0~EL3)，安全管控完整。	最小化原则，只保留最基本功能、早小机制。信任基小，可证明安全。	主体处于内核外，相互独立，稳定性、安全性好。	应用与内核分离、应用与应用分离，基于安全管控的IPC调用	原生架构支持服务化及安全调用，FC、本地IPC、远程IPC自适应。	基于能力的访问控制，类零信任安全机制。	符合ASIL-D最高等级开发流程认证、产品认证，可形式化证明。	QNX

- 微内核的最小化原则、最小权限原则决定其原生适用于高信息安全、高功能安全系统；
- 微内核的服务化架构决定其原生支持智能化、网联化；

目录 CONTENTS

01

域融合现状与趋势

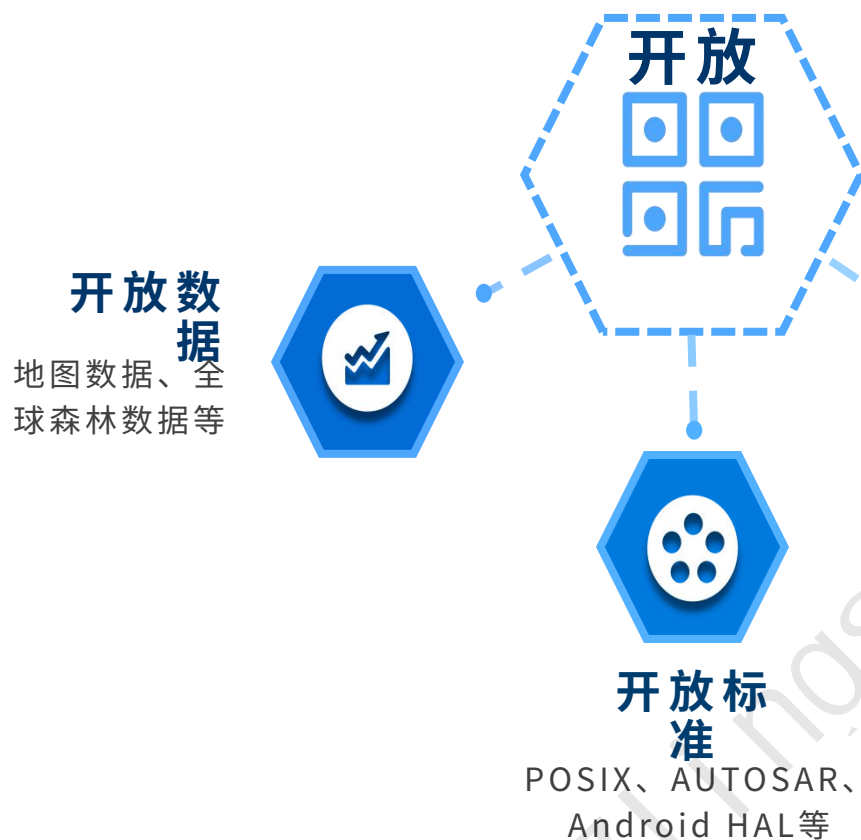
02

汽车操作系统的技术路线

03

汽车操作系统的发展策略

开放 ≠ 开源，开源 ≠ 自由、免费



用户并非可以自由随意的修改和使用任何开源软件，需遵循开源软件的许可协议要求。

开放源码

源码可公开查看，需要遵循开源合规要求

开源许可

宽松型

BSD

Apache

MIT

著作权型

GPL

LGPL

MPL

商业模式

软件市场模式

AOSP+GMS

专业服务模式

Redhat

售卖开源软件

BSD Unix

延迟开源模式

QT

带动产品售卖

芯片SDK

认证和培训

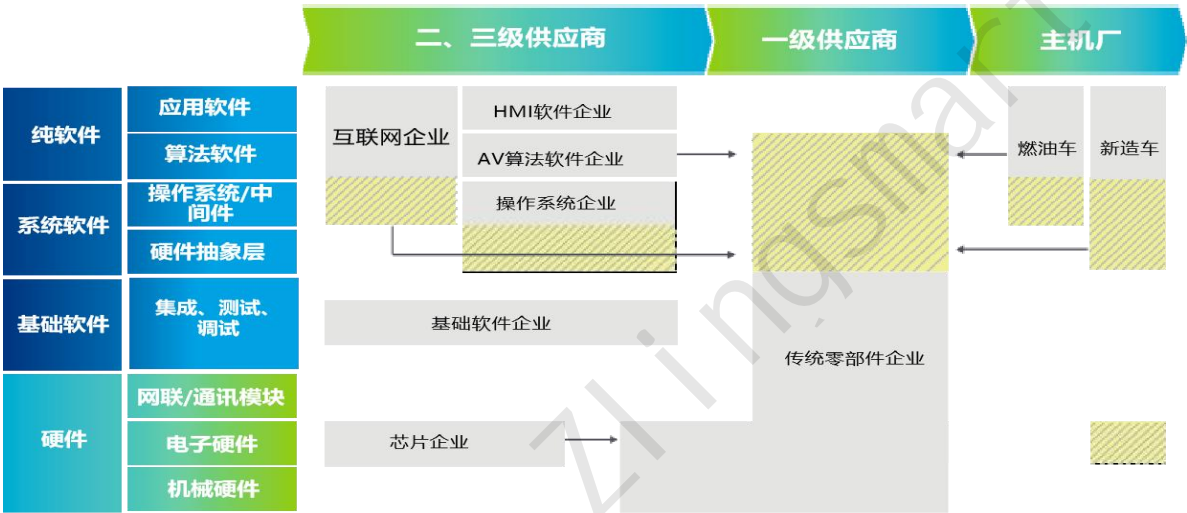
悬赏、众筹、捐赠等

品牌效应

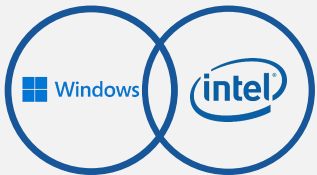


开源软件不仅靠兴趣爱好，还要持续运营的经济基础。

汽车操作系统应具备开放性，兼容应用、中间件、硬件等生态，支持上下游供应商的重塑

编号	开放模式	模式说明	产品示例
1	架构、接口、组件全封闭	需求方与供应商约定产品功能、性能指标等	特定领域的OS
2	架构、接口开放标准，组件封闭	架构和接口依据行业、客户、技术标准，如POSIX、ARINC653、AUTOSAR等	QNX、VxWorks、Windows、ARINC653、AUTOSAR OS等
3	架构、接口开放标准，部分组件开放源码	<div><div>➤ 标准化、开放的产品架构和接口</div><div>➤ 多供应商产品组件可组合、可对接、可替换，保证生态兼容，供应链灵活性、持续性</div></div>	Android、Harmony等
4	架构、接口、组件全部开放源码	开源开放指源代码可见，授权协议约束使用方式	Linux、Redhat、FreeRTOS等



⑤智能驾驶芯片与操作系统适配过程中存在哪些痛点和瓶颈？

编号	生态模式	模式说明	OS类型
1	PC时代 - Wintel联盟 	<ul style="list-style-type: none"> ➤ 架构和应用编程接口标准化，向上开放给应用软件开发 ➤ Windows重点支持Intel/AMD的X86架构，其它芯片生态难以兼容 ➤ 应用生态因标准化而繁荣，芯片生态封闭 	标准化操作系统
2	智能数码时代 - AA联盟 	<ul style="list-style-type: none"> ➤ 架构和应用编程接口标准化，向上开放给应用软件开发 ➤ 内部系统组件（AOSP）开源，支持系统深度定制，发布ROM型OS ➤ HAL等组件接口标准化，芯片和硬件厂商遵循标准可闭源提供 ➤ 系统软件因开源而产生特色化的ROM型OS，应用和芯片生态因标准化而繁荣 	定制型操作系统
4	智能汽车时代 - 生态联盟 	<ul style="list-style-type: none"> ■ 芯片特征 <ul style="list-style-type: none"> ➤ 架构异构：CPU、MCU、GPU、NPU、DSP等 ➤ 供应商重塑：传统芯片供应商（NXP、Renesas等）、新势力（地平线、黑芝麻等）、跨界供应商（NVIDIA、高通、MTK等） ■ OS特征 <ul style="list-style-type: none"> ➤ 需求差异性：安全车控OS（实时、安全、可靠）、智能驾驶OS（安全、可靠、高计算性能）、车载OS（人机交互、高性能） ➤ 供应商重塑：传统OS供应商（QNX、VxWorks、EB、Vector）、新势力（斑马OS、RAITE OS） ■ 服务组件 <ul style="list-style-type: none"> ➤ SOA框架：跨芯片架构、跨网络的SOA服务架构 ➤ 中间件：算法、功能组件等与SOA的集成 	泛在操作系统

谢谢!

RAITE

中瓴智行
Zlingsmart

<http://www.zlingsmart.com/>