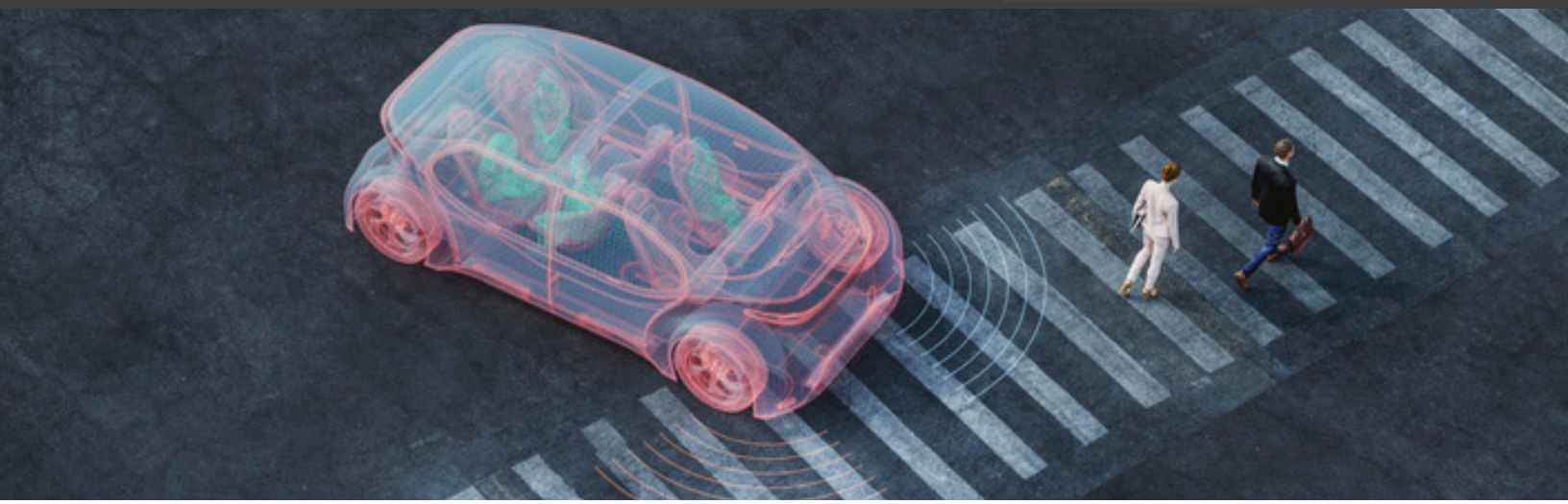


软件容器在安全关键型环境中的优势

从硬件定义汽车到软件定义汽车，这种转型正在彻底改变汽车行业及其他各行业。这个转型为互联、自动驾驶和电动汽车领域释放了前所未有的创新潜力。软件定义汽车，促使 OEM（整车制造商）在售前和售后快速推出新功能，以满足消费者不断变化的品味和需求。

不过，为了使之成为可能，软件定义汽车需要一个优化程度更高、更适合设计演进的硬件架构，以及一个更开放、支持软件开发和管理新方法的软件架构。OEM 需要的是修改单个功能的灵活性，而不是对整个庞大的代码库进行更新，并且他们需要一种技术来确保任何更改都不会对汽车中的其他安全关键型软件产生负面影响。

软件容器化提供了这种灵活性。在安全关键的汽车环境中实现容器将需要汽车软件架构的转变，以及整个行业的标准化。

全新架构

随着软件驱动的功能在汽车领域的增加，计算硬件也在不断发展。执行特定功能的专用硬件和软件的电子控制单元 (ECU) 已经过时，取而代之的是新出现的高性能计算平台，例如中央车辆控制器 (CVC) 和开放服务器平台 (OSP)。CVC 可以向上集成车身控制功能和网络，而 OSP 则整合高级驾驶辅助系统 (ADAS) 和用户体验特定域的功能。

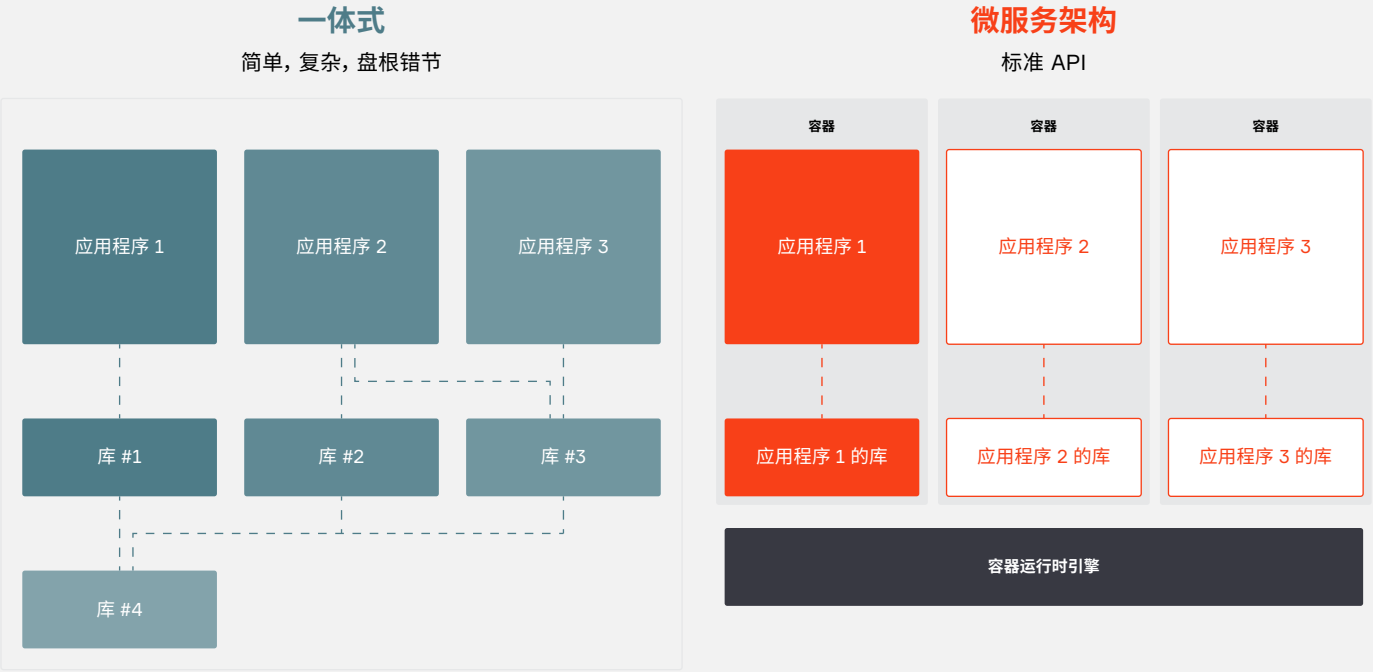
这种新的硬件方法节省了空间和成本，但也需要一种新的软件方法。如果软件还像 ECU 一样采用一体式架构，改进软件就需要对运行软件的整个设备进行全面的重新测试和验证，并且每次无线 (OTA) 更新都将比实际需要的更新大得多，会占用大量带宽和时间。

摆脱整体式方法的关键是对功能进行模块化和抽象化，使其成为可管理的软件块，这是 IT 领域常见的做法。例如，汽车内的设备由专门的软件控制，该软件负责处理设备收发的所有信号，同时向上层软件提供标准、简化的服务接口。这样，上层软件就不必处理设备的具体管理细节，而开发人员（有时可能对汽车的具体细节知之甚少）则可以专注于上层逻辑（设计）。

事实上，所有车辆功能都可以进行抽象化并作为服务提供给其他软件，从而打造大家所熟知的基于服务的架构。软件容器非常适合这种方法。

放弃一体式架构

容器化消除了对通用软件库的相互依赖，允许每个功能独立更新。



容器化是为云计算而开发的一种方法，该方法将应用程序置于标准结构中，以确保应用程序之间的依赖关系（以服务接口的形式）是已知且受控的。这有助于提高代码库的稳定性和一致性，简化服务隔离，并有效降低与其他软件相互干扰的可能性。此外，容器化还可以防止针对一个应用程序的攻击传播到其他应用程序，从而提高安全性。

通过使用容器，OEM 和供应商可以采用现代、敏捷的软件方法：小团队（使用容器化方法）在车辆的整个生命周期中逐个改进各个功能，通过 OTA 更新将这些更改快速且自动地集成到汽车中。通过与云原生计算基金会等开放规范保持一致，OEM 及其供应商可以轻松更新车辆平台，甚至可以将功能从一个平台移植到另一个平台，从而节省时间和金钱，同时降低错误和系统故障的风险。

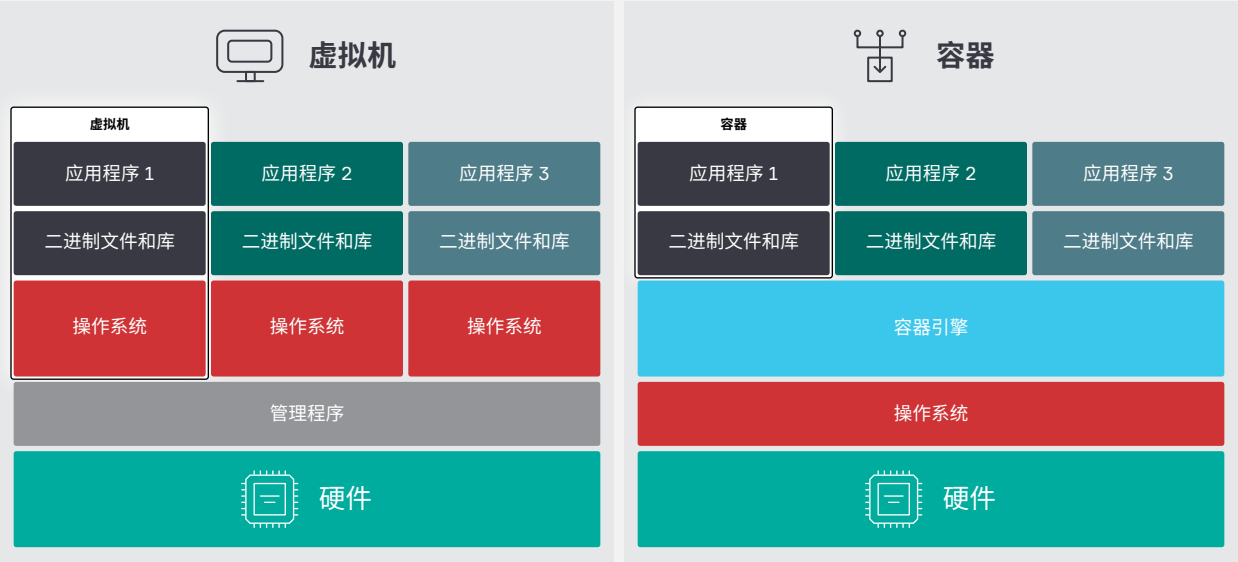
作为向软件定义汽车转型的一部分，容器化在非安全相关功能方面备受整个行业的关注。由于容器尚未在汽车中推广使用，因此其尚未经过认证，不能用于 ADAS 等安全关键型汽车应用程序，但容器在很多方面都非常适合安全功能。在其他任务关键型行业（包括航空航天、国防、电信和医疗）中，利用容器的解决方案已经证明了其性能，这有助于验证其可行性。主要行业参与者正在根据开放容器计划 (OCI) 制定的规范来合作制造符合行业标准安全认证的容器。

容器在汽车计算中的作用

容器是一种轻量级的软件打包格式。它通常包括用于一项功能的应用程序以及运行该应用程序所需的特定组件，例如库或编程语言运行时。驻留在容器中的多个应用程序可以在一个片上系统或 CPU 上运行，并在操作系统级别共享处理器周期、内存、存储和网络资源。

虚拟机与容器

管理程序支持虚拟机，每个虚拟机都有自己的操作系统和应用程序，
而容器位于操作系统和容器引擎之上，因此更加轻量。



最初开发容器的目的，是为了能够轻松地分发和扩展基于云的应用程序。与虚拟机 (VM) 一样，容器允许在共享计算环境中同时运行不同的工作负载。然而，虚拟机在硬件级别进行虚拟化，并为客户应用程序整合了完整的计算环境，包括操作系统。容器在操作系统级别进行虚拟化，并且通常更小。汽车软件架构可以同时包含虚拟机和容器：虚拟机允许在共享硬件上运行多个操作系统，而容器则可以在每个操作系统下分离功能。

车辆平台完成容器化所需的三个主要要素：支持容器的运行时环境、容器编排系统（例如 Kubernetes）以及用于安全可靠地与云动态交换容器的 OTA 网络连接。运行时环境提供了容器之间的标准化环境和用于监视每个容器运行状况的可见性。容器编排系统额外增加一层，用于监督给定车辆中的所有容器。

容器化将每一项功能都变为标准模块，使其可以像乐高® 积木一样任意添加或移除，让软件的安装、更新和管理更加简便。在可扩展性方面，操作系统可以在特定时间（例如车辆启动时）临时添加更多功能实例。在可移植性方面，全面执行一项功能的容器可能能够在不同的领域、模型和平台中重复使用。容器还简化了旧版软件与下一代车辆的集成工作。

相比之下，向传统汽车代码库添加一项功能可能需要进行长达 100,000 英里的驾驶测试，才能对整个代码库重新测试，因为无法预测该功能与其他功能交互的方式。随着车辆平台变得越来越复杂，这种传统方法的时间和金钱成本也越来越高。

在汽车环境中使用容器有几项潜在的好处，包括 OTA 更新更轻松、可移植性更高、能够采用现代软件方法以及可以跨开发环境进行协调。

OTA 更新

对于 OEM，容器化最重要的好处就是可以更快、更容易且更频繁地进行 OTA 更新。越来越多的消费者期望在产品的整个生命周期中获得新功能，并且很多时候愿意为新功能付费，因此 OTA 更新变得至关重要。OTA 更新提供了一种可以提高车主满意度和品牌忠诚度，并有可能产生额外收入的新方法。

在传统的整体式软件环境中，售后更新往往是大型更新，其频率较低且难以通过无线方式分发。借助容器化，售后更新可以变得更小、更有针对性，对于要处理的功能数量和要分发的车辆数量都是如此。代码验证是一项标准化常规流程，而较低的带宽要求意味着大多数更新可以通过蜂窝网络发送。

更新的速度和频率取决于许多因素，其中最重要的是容器的结构及其设计的轻量程度。在开发人员确定如何才能以最佳方式将应用程序划分为更小的部分并将每个部分置于各自容器中时，需要重点关注这一方面。

通过确保容器结构得到优化，OEM 可以更轻松地执行更新，以推送新功能、订阅、第三方应用程序和各车辆的个性化内容。当消费者对新功能表现出兴趣时，专门的团队可以快速交付这些功能，从而形成高效的反馈循环。

可移植性

借助容器，汽车功能还可以在不同车辆平台、模型甚至供应商上进行移植。只需进行少量集成工作，即可将针对一个平台开发的功能部署在整个 OEM 的产品系列中。当放置在标准容器中时，旧版应用程序可以轻松集成到新的容器化车辆平台中。可移植性还可以防止供应商绑定：OEM 可以更轻松地转向新的供应商，将以前供应商的软件替换为提供相同服务的另一供应商的容器化应用程序。

容器的可移植性意味着软件开发不再受制于硬件平台的要求。OEM 的车辆平台从专用 ECU 发展到域控制器，再发展到在中央计算引擎上实现全面服务器化，在这个过程中，OEM 可以重复使用容器化应用程序。由于软件更新成本大幅降低，制造商甚至可以更改当前车型或已上路车辆的硬件。

现代方法

相比其他行业，汽车行业很晚才开始采用能够提高敏捷性和发展速度的软件方法，包括 DevOps 和持续集成/持续部署。使用传统瀑布式软件开发流程的 OEM 需要三到五年的开发周期，这已经无法跟上技术潮流，也无法满足消费者的需求。

容器提供了快速适应需求和设计变化所需的模块化架构。致力于特定功能开发的团队可以在固定的发布周期中开发和更新软件，在每个产品甚至多个平台的整个生命周期中管理该功能代码。此外，采用这些方法的汽车公司将有更多的软件开发人员可供选择和招聘。

协调

OEM 及其供应商对相同容器规范的标准化程度越高，就能越好地简化软件部署。大多数汽车应用程序在仿真、开发、测试、验证和生产过程中都会经历多个环境，每个环境都由单独的团队进行管理。如果没有容器，团队需要仔细协调任何可能环境带来的更改，这些环境可能影响程序的运行方式。

重用通用容器格式意味着每个函数在所有环境中都具有相同的上下文，并且开发人员可以使用标准测试来验证容器中的代码是否正确运行。这样就消除了云或任何其他环境中的开发瓶颈。此外，车辆上路后，容器化应用程序能够使用云来获得更强的处理能力。例如，一些自动化或信息娱乐功能可以部分在云中相同容器内运行。

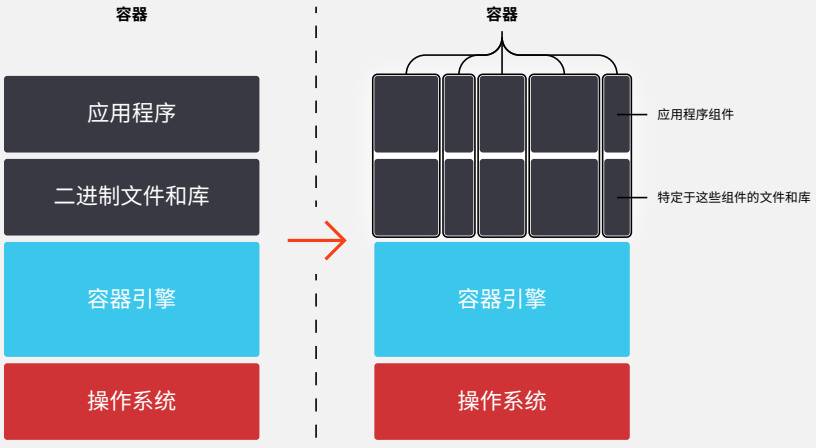
安全挑战

传统上，安全关键型汽车软件是静态的，在专门的 ECU 中隔离，并且大小有限，以最大限度地减少认证代码的开销。未来，OEM 希望能够在共享硬件上运行这些功能并更频繁地更新它们。

将安全关键型功能与其他应用程序放置在一起时，需要在具有不同安全要求的应用程序之间建立逻辑边界。容器化提供了一种通过轻量级、易于管理的组件来实现模块化架构的方法，这些组件可以封装安全关键型功能并建立起这些边界，这样，在修改一个容器映像时，就不会对同一环境中运行的其他容器产生不可预测的影响。这可以保护安全关键型应用程序免受非关键功能意外故障的影响，否则这些故障可能会蔓延并违反安全要求。

优化的容器

容器化和微服务带来了新的可能，应用程序本身不必是一体式的。如有必要，应用程序可以分为更小的模块，每个模块都在自己的容器中，使更新更有针对性、更加轻量。



虚拟机的作用

在当今的一些车辆平台上，这种逻辑边界是由具有汽车安全认证的管理程序所监管的虚拟机创建的。

虽然虚拟机将继续在安全关键环境中发挥重要作用，但容器具有一定的优势。因为与虚拟机相比，更多的容器可以共享给定数量的计算资源，所以容器可以更轻，并且验证安全性所需的开销更少。此外，容器更容易跨环境进行标准化并通过无线方式更新。

虚拟机和容器可以在安全关键型场景中一起使用。例如，虚拟机可以运行实时操作系统（如 Wind River VxWorks®），同时软件容器可以在该 RTOS 上运行。

可靠性的价值

容器具有明确定义的边界，非常适合那些需要准确掌握未来情况的应用程序。

例如，如果没有容器，开发人员在尝试加载某个特定更新之前，可能无法确定该更新是否会过度使用硬件资源，即使更新可行，他们会犹豫是否要进行任何更改，因为担心会导致它停止工作。与之相反，容器具有定义好的边界，这意味着开发人员始终准确地知道它将使用的资源量。

同样，容器也不是永久性的，也就是说每次车辆启动时都会从主镜像加载容器。它们不会随着时间的推移而改变，并且它们使用的数据存储在不同的位置。从安全角度来看，这一点很有吸引力，因为这意味着容器始终能以预期的相同方式运行，并且容器所提供的服务可以被认为是不可变的。这还意味着可以关闭未使用的应用程序，从而释放系统资源。

标准至关重要

经过安全认证的容器运行时环境可以确保所有容器都拥有保证安全所需的资源。容器运行时还没有获得汽车安全认证，但我们已经着手解决这个问题。要创建所有不同开发阶段所需的流程和文档，还需要完成很多工作。

标准化可以让更多汽车供应商享受到容器化的好处，为下一代软件发掘更多商机，并且更加轻松地集成应用程序。此外，广泛采用一种容器格式将为安全认证创造动力。

业界对容器标准化理念的支持日益增长，而 OCI 容器规范提供了一种经过验证的方法。OCI 为汽车软件开发提供的好处与 OCI 为云计算提供的好处类似：开发人员不必让应用程序去适应每个 OEM 的环境并执行耗时的集成工作，而是可以在任何车辆平台上找到符合 OCI 的运行时环境，并且只需对编写的代码进行少量修改即可添加到平台。

这将大大有助于实现这样一种愿景：软件定义汽车作为平台承载来自于广泛生态系统的供应商提供的应用程序。OCI 的作用类似于 Android 操作系统在智能手机中的作用。OEM 可以搭配更广泛的功能，客户可以在购买车辆时进行选择或稍后下载（或订阅）。

标准化面临的阻碍

容器化是一种被广泛接受的计算方法，但主要用于资源不受限制的环境中。由于车辆有着严格的安全要求，并且在空间、重量和功耗方面存在严格限制，因此带来了各种问题。

尽管如此，在汽车中使用容器的挑战与其说是技术性的，不如说是文化性的。一些供应商已经在开发支持容器的汽车运行时环境，并正在努力获得安全认证。为了利用容器化带来的好处，OEM 和供应商还需要改变长期的开发实践，并将一些资源从集成和测试转移到创新和持续更新。虽然所有参与者最终都会受益，但需要有一些参与者打破僵局，率先开始转型。

第一个在自身供应商生态系统中引入容器化的 OEM 可能需要三年或更长时间才能实现目标。但如果行业围绕一个标准（如 OCI）联合起来，则其他标准也会很快跟进，以便同样实现提高敏捷性、创新、节省成本、可升级性和缩短上市时间等各种好处。对于 OEM，下一步是在车辆平台上实施容器管理层，此过程可能还需要几年时间。

推动行业发展

安波福正在与其他供应商合作，加速整个汽车行业的容器采用和标准化。在 CES 2023 上，我们演示了如何使用容器编排系统将符合 OCI 的容器通过 OTA 部署到车辆上。风河（2022 年安波福收购的边缘软件提供商）提供支持 OCI 容器的嵌入式运行时操作系统，我们目前正在与风河开展合作，计划于 2024 年更新其汽车安全认证。

过去，使用传统整体式软件的成本和局限性并未成为 OEM 商业模式或竞争力的主要障碍。但随着车辆平台变得越来越复杂，消费者期望各种安全、舒适和便利功能更快地发展，每次发布新车型都会带来更多的痛苦。通过有力的行业合作，支持安全关键型功能的标准化容器很快就会出色地实现更敏捷的开发和持续改进。

作者简介

Florian Baumann
安波福高级汽车架构软件资深总监

Florian Baumann 领导的团队专注于开发尖端技术，以解决汽车行业一些最棘手的问题。Florian 在机器学习、软件开发、DevOps 和云架构方面拥有丰富的经验，他带着自己毕生对技术的热情投身安波福下一代车载软件平台的创建工作中，（下一代平台）降低了平台的复杂性，同时（和现有平台相比）提供无缝的开发者体验。

Michel Chabroux
风河产品管理副总裁

Michel Chabroux 负责推动风河智能边缘产品组合的技术和业务战略。他拥有 20 多年的从业经验，从事过技术销售、支持、培训和产品管理方面的职位。

Timm Zimmermann
安波福软件定义汽车架构副总裁

Timm Zimmermann 领导着一支汽车软件架构团队，专注于如何充分利用安波福的智能汽车架构SVA™实现软件定义汽车。Timm 在主机开发和安全技术领域深耕多年，他运用自己的丰富经验为下一代电子/电气架构设计更先进的设计原则。

详情请见 APTIV.COM/MWD →