AUTOSAR 技术概述

AUTOSAR 的计划目标主要有 3 项，第一是建立独立于硬件的分层的软件架构；第二是为实施应用提供方法论，包括制定无缝的软件架构堆叠流程并将应用软件整合至 ECU 中；第三是制定各种车辆应用接口规范，作为应用软件整合标准，以便软件构件在不同的汽车平台上的复用。

1、AUTOSAR 软件架构

为了实现 AUTOSAR 的目标，即实现应用程序和基础模块之间的分离，汽车电子软件架构被抽象成几个层，如图 1 所示。

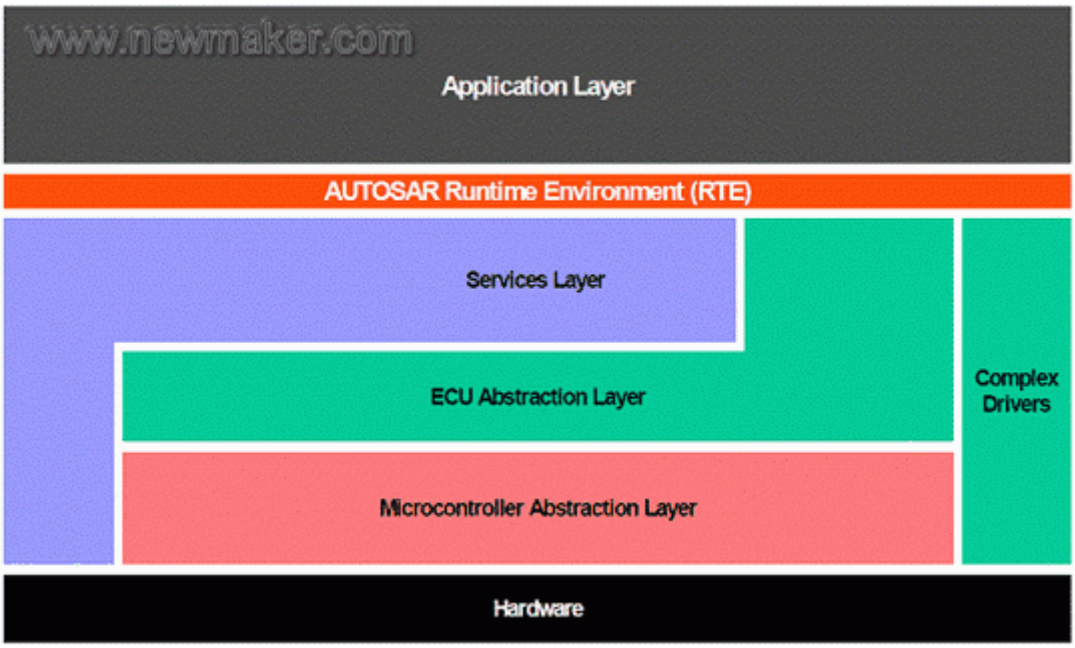

图 1：AUTOSAR 软件架构层次图

为了区别软件依赖和硬件依赖，基础软件分为四个层次：服务层（Services Layer）、ECU 抽象层（ECU Abstraction Layer）、微控制器抽象层（Microcontroller Abstraction Layer）和 RTE（Runtime Environment）。除此四层外，在 AUTOSAR 软件架构中还有复杂驱动（Complex Driver），由于对复杂传感器和执行器进行操作的模块涉及到严格的时序问题，在 AUTOSAR 中这部分没有被标准化。

* 服务层提供包括诊断协议、存储管理、ECU 模式管理和操作系统等在内的系统服务。除了操作系统外，服务层的软件模块都是与平台无关的。

* ECU 抽象层将 ECU 结构（如外设与 ECU 的联接方式等）进行了抽象处理。该层与 ECU 平台相关，但与微控制器无关。

* 微控制器抽象层包括微控制器相关的驱动（如 I/O 驱动、ADC 驱动等）。

* RTE 层负责 AUTOSAR 软件构件（即应用层）相互间的通信以及软件构件与基础软件之间的通信。RTE 层之下的基础软件对于应用层来说是不可见的，必须通过 RTE 进入，它将软件构件从对底层软件和硬件平台的依赖中独立出来，实现了应用程序和基础软件之间的分隔。

2、 AUTOSAR 方法论

AUTOSAR 为符合该标准的汽车电子软件系统开发过程定义了一套通用的技术方法，这种方法即被称为 AUTOSAR 方法论（AUTOSAR Methodology）。汽车 OEM 作为整车系统功能的规划和设计者，需要了解并掌握 AUTOSAR 提供的这套开发流程，才能主导和推进符合 AUTOSAR 标准的系统的开发过程。

兼容 AUTOSAR 标准的汽车电子软件系统设计与开发流程如图 2 所示。

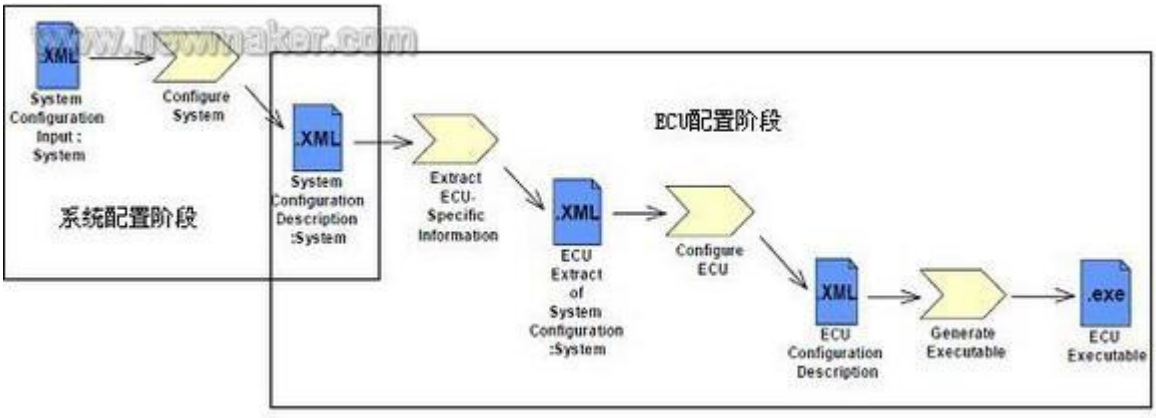

图 2：AUTOSAR 系统设计与开发流程

主要步骤可划分两个阶段：

第一个阶段是系统配置阶段，这属于系统级设计决策工作。首先是编写系统配置输入文件，

为 XML 类型的文件。应用软件的描述术语在 AOTUSAR 中为软件构件（Software Components），该文件将确定需要使用的软件构件（即系统具有哪些功能）和硬件资源（ECU），以及整个系统的约束条件。AUTOSAR 提供了一系列的模板（软件构件模板，ECU 资源模板和系统模板）和标准的信息交换格式，工具供应商可据此提供相应的工具支持，从而简化系统设计的工作，最终系统设计者只需要使用工具填充或编辑相应的模板即可导出系统配置输入文件。

系统配置输入包含三部分内容，第一个输入是软件构件描述，定义每个需要的软件构件的接口内容，包括数据类型，端口，接口等；第二个输入是 ECU 资源描述，定义了每个 ECU 的资源需求，如处理器、外部设备、存储器、传感器和执行器等；第三个输入是系统约束描述，定义总线信号，拓扑结构和软件构件的映射关系。

系统配置阶段接下来的工作是将初步获得的系统配置输入文件借助系统配置生成器生成系统配置描述文件，同样为 XML 文件，这是系统配置阶段的最终工作成果。该文件将包含所有的系统信息，包括将软件构件映射到相关的 ECU 上（这种映射需要考虑到构件的需要、构件的连接、资源需求以及约束条件，有时也需要考虑成本等方面的因素），以及通信矩阵（整车的网络结构、时序以及网络数据帧的内容）。

第二个阶段是 ECU 的配置，这阶段的工作需要对系统中每个 ECU 分别进行。首先是使用第一个阶段的工作成果——系统配置描述文件，从中提取出与各个 ECU 相关的系统配置描述信息，提取的信息包括 ECU 通信矩阵、拓扑结构、顶级功能组合（据此产生需映射到该 ECU 上的所有软件构件），将放在另一个 XML 文件中。提取信息的工作可借助工具完成。然后进入 ECU 配置的实际工作中，这一步负责往输入对象中添加具体应用所必需的信息，如任务调度、必要的 BSW 模块、BSW 配置信息、给任务分配的可运行实体等。这一步的结果被放在 ECU 配置描述文件中，它包含了具体 ECU 所需的所有信息。最后一步是生成具体 ECU 的可执行程序，此步将根据 ECU 配置描述文件中的配置信息构建完成 ECU 的基础软件的设置和与基于 AUTOSAR 构件的应用软件的集成，最终生成 ECU 的可执行代码。

此外，要说明的是，AUTOSAR 系统的设计过程使用了虚拟功能总线（Virtual Functional Bus）的概念。虚拟功能总线（Virtual Functional Bus）将 AUTOSAR 软件构件相互间的通信以及软件构件与基础软件之间的通信进行了抽象，同时使用预先定义的标准接口。而对于虚拟功能总线来说，ECU 内部通信和外部总线通信并没有什么区别，这种区别要等到系统布局以及 ECU 的具体功能最终确定才会体现出来。软件构件本身对于这种区别并不关注，因此我们可以在独立的情况下开发软件构件。在系统实现过程中，虚拟功能总线所代表的功能最

终以 RTE 的生成来体现。

3、标准化的应用接口

通过 RTE 实现 AUTOSAR 软件构件（即应用程序）相互间的通信以及软件构件与基础软件之间的通信的前提是，软件构件必须具有标准的 AUTOSAR 接口。目前，AUTOSAR 3.1 版已定义了一些典型的汽车电子应用领域（动力，车身/舒适和底盘）的标准接口。AUTOSAR 按照功能逻辑分别将这些领域的系统划分成若干个模块，这些模块可被视为一个软件构件或多个软件构件的组合，这些功能性的软件构件的接口被明确定义，所定义的接口的内容包括名称，含义，范围，数据类型，通信类型，单位等。应用软件开发者在软件构件的设计与开发时需要应用这些接口定义。

这里以车身/舒适系统的雨刷管理的软件构件的接口定义为示例，如图 3：

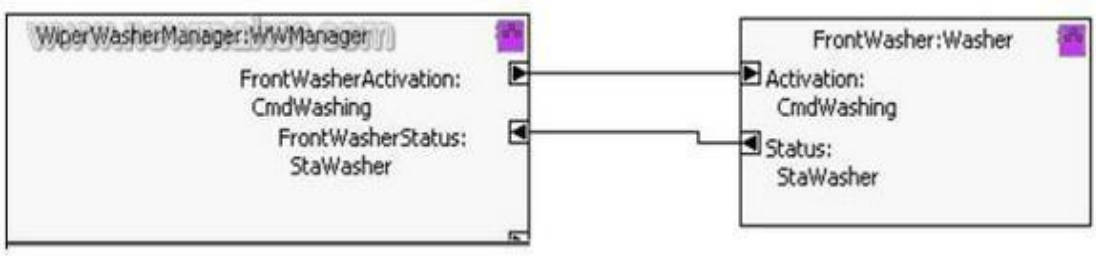

图 3：软件构件的接口定义

说明：

雨刷管理构件（WiperWasherManager）有两个接口，CmdWashing 和 StaWasher,图中 WWManager 表示为雨刷管理软件构件的实例。针对 CmdWashing 接口定义了以下信息：

1) CmdWashing 接口由 WiperWasherManager 构件提供，其数据内容为 FrontWasher 构件的 Activation 接口所使用。

2）CmdWashing 包含一个“Command”的数据元素。

3）“Command”的数据类型为“t_onoff”。

4）“t_onoff”属于“RecordType”，该类型描述一般的开/关信息。

应用软件开发者应该意识到，面向 AUTOSAR 运行时环境（RTE）接口的应用软件设计的重要性，及早地将 AUTOSAR 应用层接口引入到实际的项目中来，为实现应用软件的可复

用性做好准备，从而优化整个软件开发流程。