FORD LDM Localization - 功能 #3383

Task # 3240 (进行中): FORD文档输出

功能 # 3349 (进行中): SWQA文档

功能 # 3353 (已解决): TDR_RQT_003701_705377 Software Design Best Practices

TDR_RQT_003701_705377 004 动态内存分配（Dynamic Memory Allocation）

2025-03-17 09:47 - 力常 张

状态:	已关闭	**开始日期:**	2025-03-17
优先级:	普通	**计划完成日期:**	2025-03-25
指派给:	凡慧 孔	**% 完成:**	100%
类别:		**预期时间:**	0.00 小时
目标版本:	FORD_TDR_Documents	**耗时:**	0.00 小时

描述

参考软件需求“ DR-003701-707964 Dynamic Memory Allocation ”

需求描述
本设计规则支持
RQT-003701-705377《软件设计最佳实践》，适用于包含以下类型软件的所有电子控制模块：非AUTOSAR操作系统（Non-AUTOSAR OS），AUTOSAR Adaptive 平台，第三方软件，并作为该RQT的合规性实施方法。
.
禁止使用任何内存分配函数（如C语言中的malloc()）创建动态数据结构（即使简单原子数据类型亦不允许），仅允许使用静态数据结构。
. 针对ASIL等级为QM/A/B的软件，可按照"分析/讨论"章节规定的例外流程使用动态内存分配。

技术规范
动态内存分配（dynamic memory
allocation）对软件系统构成风险，因其引入非确定性（non-determinism），无论分配行为发生在操作系统层、中间件层、应用层或相关依赖库中。因此，限制动态内存分配可通过以下机制提升系统健壮性：
预防内存碎片化（memory fragmentation）
规避内存泄漏（memory leaks）
消除内存耗尽风险（out-of-memory condition）
阻断堆栈冲突（stack/heap collision）
简化堆管理复杂度（heap management）
理想状态：禁止使用动态内存分配技术。
例外许可：在无高等级安全约束的常规功能（commodity）及基础设施（infrastructure）特定用例中可合理应用此类技术。

注：非静态函数局部变量（non-static function-local variables）不被视为动态内存分配对象。

分析/讨论
本设计规则支持RQT-003701-705377“ 软件设计最佳实践 ”，适用于所有带软件的电子模块，是符合RQT的一种方法。
设计规则的闭合以完成软件技术设计评审（Technical Design Review,
TDR）为标志，具体通过"SWQA通用TDR检查单"中的下列问题项落实：
是否使用动态内存分配？是否使用内存分配函数（如C语言malloc()）？

历史记录

#1 - 2025-03-17 09:48 - 力常 张

- 状态 从 新建 变更为 已解决

#2 - 2025-03-24 15:10 - 稚媛 黄

- 主题 从 DR-003701-707964 动态内存分配（Dynamic Memory Allocation） 变更为 TDR_RQT_003701_705377 004 动态内存分配（Dynamic Memory Allocation）

#3 - 2025-03-24 15:25 - 稚媛 黄

- 描述 已更新。

#4 - 2025-03-25 10:54 - 涛 陆

- 计划完成日期 被设置为 2025-03-25

- 指派给 从 力常 张 变更为 凡慧 孔

#5 - 2025-03-25 14:38 - 凡慧 孔

- 状态 从 已解决 变更为 已关闭