智能化与控制

船用柴油机电控系统软件架构的安全性设计与验证

姜春宇[1,2]，金江善[1,2]，胡建村[1,2]

(1. 七一一研究所，上海 201108；2. 船舶与海洋工程动力系统国家工程实验室，上海 201108)

摘　要：船用柴油机自动化程度不断提高，软件已经成为控制系统乃至柴油机安全运行的重要影响因素。从软件安全性设计角度提出了一套船用柴油机控制软件架构阶段的开发流程、设计及验证方法，包括：软件架构开发流程、架构安全性设计、软件集成后的时间和空间验证等，以符合船级社指南对安全性评估的要求。

关键词：船用柴油机；电控系统；软件架构；安全性

中图分类号：TK424.3　TP311.5　　**文献标识码**：A　　**文章编号**：1001 -4357(2018)06 -0001 -05

Safety Design and Verification of the Software Architecture for the Electronic Control Systems of Marine Diesel Engines

Jiang Chunyu[1,2], **Jin Jiangshan**[1,2], **Hu Jiancun**[1,2]

(1. Shanghai Marine Diesel Engine Research Institute, Shanghai 201108;
2. National Engineering Laboratory for Marine and Ocean Engineering Power System, Shanghai 201108)

Abstract: With the improvement of automation in marine diesel engine, software has been one of the key safety factors of the control system or even the whole diesel engine. From the aspect of software safety, a whole set of development process, design and verification method in the stage of software architecture for the control systems of marine diesel engines are put forward, including the development flow of software architecture, safety design and the verification of time and space after software integration, which is expected to meet with the CCS guidelines for software safety evaluation.

Key words: marine diesel engine; electronic control system; software architecture; safety

0　引　言

安全性（Safety）的一般性定义是“避免那些可能引起人员死亡、伤害、疾病，或者设备、财产的破坏或损失，或者环境危害的条件”[1]。软件代码本身不会造成危险，但是，当软件是安全关键系统的一部分，由于软件的问题（如意外操作、非法访问、程序失效、执行次序颠倒、数值错误等）通过硬、软件接口使硬件发生误动或失效，就会造成严重的安全事故。随着船用柴油机控制系统功能的不断增加，软件的复杂度和规模也随之增加，在考虑系统的安全性时，也必须重视软件的安全性。2015 年，中国船级社发布了船舶领域的《船用软件安全级可靠性评估指南》，将柴油机燃油喷射控制功能定义为最高等级，Ⅲ级；将调速功能定义为安全等级，Ⅱ级，明确了船用柴油机电控系统软件功能模块的安全关键特性[2]。

在船用柴油机电控软件中，不同安全等级的软件模块通过实时操作系统的调度，运行在同一个控制芯片内，低安全等级模块的失效，极有可能引起高安全等级软件的失效。因此，必须确保不同安全等级的软件模块在空间和时间上隔离，以防止低安全等级软件故障的传播。

对于混合安全等级的软件，在软件安全规范、

收稿日期：2018-02-09；**修回日期**：2018-04-11

作者简介：姜春宇(1981 -)，男，高级工程师，主要研究方向为柴油机智能化控制技术，jiangchunyu@csic711.com

设计指南中都提出了明确的设计要求。在 IEC61508 和 ISO26262 中明确要求：不同安全等级的软件模块须在时间和空间上隔离，以防止低安全等级软件的失效影响高安全等级软件的运行[3-4]。在 GJB/Z102A《军用软件安全性设计指南》和 NASA－GB－8719.13《软件安全性指南》都提出了容错和容失效的设计要求："为了防止故障在软件中传播，安全关键的部件应完全独立于非安全关键的部件，还应能够既检测出自身内部的错误，又不允许将错误传递下去[5-6]。"

为满足安全规范的要求，国内外学者都进行了相关研究并提出了安全设计措施。徐丙凤从构件化系统静态结构及动态失效行为两方面深入开展了适用于构件化嵌入式系统的安全性分析关键技术研究，包含：构件之间安全等级依赖的一致性研究；安全关键软件失效行为的最小割序集以及关键事件研究等[7]。杨仕平针对分布式任务关键事实系统从应用级、操作系统级和网络级研究设计相应的安全性机制，包括：应用级多级关键度划分和防危机制，操作系统级探究基于时间隔离和空间隔离保护机制，针对网络级研究基于事件触发与时间触发实时网络协议的设计[8]。Christoph Ficek 等针对混合安全关键等级软件集成中，为防止不同安全等级任务故障的传递，改进了传统 RMS 调度方式，提出了一种新的实时操作系统调度算法[9]。

本文结合船用柴油机软件的特点，开展软件架构安全性设计分析及验证。

1 嵌入式控制软件架构安全设计过程

船用柴油机控制软件为典型的基于实时操作系统运行（RTOS）的嵌入式软件，须在软件架构设计阶段，实现软件的设计分析与决策，从软件部件的静态结构和执行序列的动态行为确定软件的整体架构，包括：部件设计、执行方案和接口设计三个方面。部件设计是在分析各个软件需求（用例）后，以面向对象或基于结构的设计方法获取软件部件，确定部件的属性、功能和软件部件间的关系等，并基于硬件平台和操作系统软件平台的特性，对软件部件划分层次结构。基于实时操作系统运行的软件，软件部件的功能都会分配到操作系统各个任务内，软件的动态行为由操作系统统一调度完成。因此，执行方案的设计必须在实时操作系统的基础上开展，确定任务的执行内容，并从软件上电复位后的整体运行序列和各个关键用例的运行序列两个方面，优化任务的分配、任务协作关系和执行序列。接口设计完成软件外部接口、部件间的内部接口和任务的接口设计。

除了完成通常的架构设计内容外，从软件安全性设计角度出发，还须进行架构的安全性设计，并在详细设计和编码完成后，在软件集成阶段开展安全性验证。在部件设计完成后，根据软件需求的安全关键等级，确定各个部件的安全关键等级，并分析各个部件安全等级依赖的一致性。在确定完成部件的安全关键等级后，对高安全等级部件开展软件 FMEA 分析，分析失效模式、失效原因和后果，并提出改进措施，降低失效风险。在执行方案设计完成后，要做好不同安全关键等级软件部件在时间和空间方面的隔离，防止低等级软件部件故障影响高等级软件部件的执行。在软件集成阶段，按照软件代码的实际执行情况，在时间维度上计算软件任务的 WCET（最恶劣执行时间）和 WCRT（最恶劣响应时间），验证任务的可调度性；在空间维度上计算任务的最恶劣执行路径，验证堆栈空间的安全性。图 1 为船用柴油机软件架构设计流程。

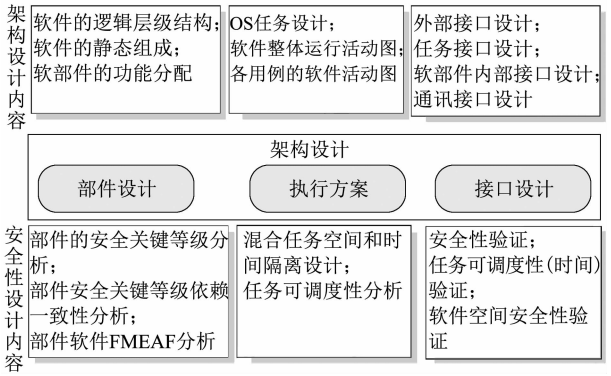

图 1 船用柴油机软件架构设计流程图

2 软件模块安全等级设计

船用柴油机控制软件中，不是所有的软件需求都是相同的安全等级。如果同一软件部件或软件任务处理不同安全等级的软件需求，则会由于时间或空间耦合，导致故障传播。因此，必须在软件设计时，确定好软件部件和任务的安全等级，做好不同安全等级软件部件和软件任务的隔离。

2.1 软件部件安全性设计

架构阶段的部件设计，通过面向对象或面向结构的方法，确定出实现功能需求（用例）的软件部件。采用统一建模语言（UML）的类图/对象图，描述每一软件部件的属性、操作以及部件之间的关系。

在进行不同安全等级软件的架构设计时，必须将需求阶段定义的安全等级分配到部件设计阶段的产物中，明确软件部件的安全等级；同时，要依据部件间的关联关系，分析软件部件安全等级依赖的一致性。一个软件部件的安全等级至少与依赖它的软件部件当中的最高软件等级相等。换言之，如果某软件部件被其它软件部件依赖，则该软件部件的安全等级至少与依赖它的软件部件当中最高的软件等级相等，保证安全等级最高的软件部件能够按照要求实现。在 UML 类图中增加标签 < 安全等级 >，并为每个部件确定具体的安全等级数值，通过部件之间的关系，分析是否存在安全等级依赖的不一致问题。

以船用电控单体泵柴油机调速控制软件为例，软件部件设计如图 2 所示，顶层确定了“转速闭环控制”、“燃油喷射控制”、“发动机转速计算”和“发动机相位计算”4 个部件。“转速闭环控制”部件从“发动机转速计算”部件获取当前转速，结合目标设定转速，进行转速的 PID 闭环计算，产生目标油量需求。“燃油喷射控制”根据目标油量，确定各缸的喷油正时和喷油脉宽，从“发动机相位计算”部件获取当前发动机相位后，依据正时和脉宽信息，驱动喷油器进行燃油喷射。

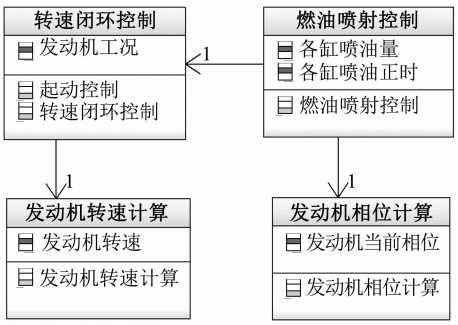

图 2　部件设计的柴油机控制软件静态结构

《船用软件安全级可靠性评估指南》中定义了柴油机燃油喷射控制功能安全等级为Ⅲ级，调速功能安全等级为Ⅱ级[2]。为上述 4 个软件部件增加 < 安全等级 > 标签及其值，见图 3。“燃油喷射控制”部件为Ⅲ级，则其依赖的部件“转闭环控制”和“发动机相位计算”部件都必须为Ⅲ级。然而“转速闭环控制部件”服务于调速功能，在指南中只要求Ⅱ级。基于以上分析，可以根据系统实际情况进行安全性架构的决策。一种方式是将“转速闭环控制”及其依赖的部件的安全等级升级，这会直接带来设计分析和测试等的任务量和开发成本的提高；另一种方式是隔离“燃油喷射控制”与“转速闭环控制”的直接依赖关系，采用 Katwijk 和 Zalewski 等人提出了防危壳技术[10]，在两者之间增加“油量防护壳”。该部件对写入的喷油量进行安全防护，如对油量变化率和极值采取限制等防护策略，以防止油量变化超过率以及最大和最小值超过防护要求。

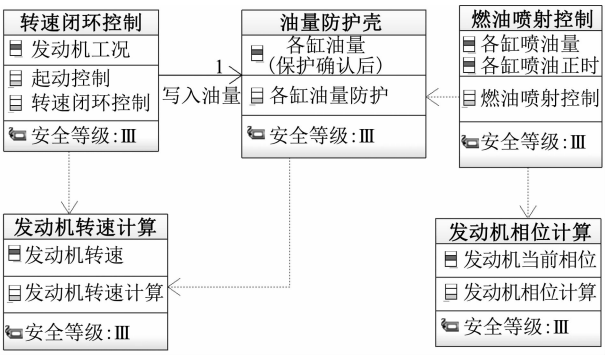

图 3　安保设计后的柴油机控制软件静态结构

2.2　操作系统任务安全性设计

基于实时操作系统的软件，多个任务会同时运行，任务之间基于优先级的不同竞争 CPU 的处理时间，这就会带来任务响应时间的不确定。对于安全关键的任务，如果未在规定时间内做出响应，则意味着失效。因此，当存在混合安全等级的软件任务时，必须合理设计实时调度策略，确保安全关键任务运行时间可预测性。Liu 和 Layland 提出的经典速率单调调度（RMS）被证明是最优的静态实时调度，在软件任务满足以下条件（1）时，保证所有硬实时任务在规定的截止时间前完成任务的执行[11]。

$$\sum_{i=1}^{n}\frac{C_i}{T_i}\leq n(2^{\frac{1}{n}}-1) \tag{1}$$

其中：C_i 为任务执行时间，T_i 为任务的执行周期（同时也是硬实时任务的响应截止时间）。

RMS 的任务优先级由执行周期决定，周期短的任务优先级高。但在混合安全等级任务共存情况下，低安全等级的任务周期可能小于高安全等级的任务，如果采用 RMS，就会出现低安全等级任务优先级高于高安全性等级任务，出现对高安全等级任务的 CPU 运行时间的抢占，也就出现了时间的耦合。因此，必须将任务的安全关键等级纳入到任务优先级的确定中。在混合安全等级任务中，首先要以安全等级确定任务优先级，安全等级越高，任务优先级越高。在相同安全等级的任务中，按照任务周期确定任务优先级。但是该种调度只能保证最高安全等级任务执行时间的确定性，无法采用 RMS 公式对所有任务的可调度进行验证。因此，

必须在软件集成后，对所有任务是否在响应截止时间前完成执行进行验证。

3 架构安全性验证

为保证各个任务的安全性，必须为每个操作系统任务分配合理的 CPU 处理时间和 RAM 运行空间。

3.1 任务时间验证

在综合考虑安全等级和执行周期确定任务优先级后，必须对任务的可调度性进行验证，即确保任务在响应截止时间前执行完毕。Joseph 和 Pandya 提出了最恶劣响应时间（worst case response time，WCRT）的计算公式及保证可调度的关系式，如式(2)。提出：当任务和比它优先级高的任务同时激活时，该任务响应时间为最长[12]。图 4 为对于 RMS 调度的 $T3$ 的 WCRT 的示意图。

$$R_i = C_i + \sum_{j \in \mathrm{hp}(i)} C_j \left\lceil \frac{R_i}{T_i} \right\rceil \leq D_i = T_i \qquad (2)$$

其中：R_i为任务的响应时间（WCRT）；C_i 为任务在没有被其他任务抢占时的最恶劣执行时间（worst case execution time，WCET）；hp（i）为优先级高于 Task i 的所有任务的组合；T_i为任务 Task i 的周期；D_i为任务 Task i 的响应截止时间，与任务周期的关系为：$D_i \leqslant T_i$。

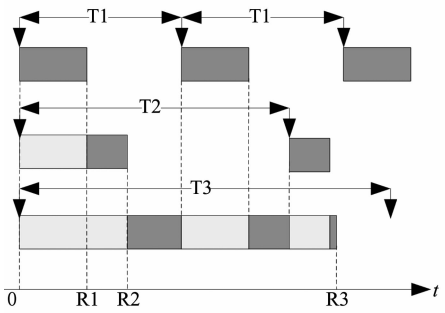

图 4 RMS 任务调度图

在将软件安全等级纳入到软件优先级配置后，任务的 WCRT 就完全不同于采用 RMS 的软件调度。如将任务 T3 定义为安全等级Ⅲ，任务 T1 和 T2 为安全等级Ⅰ，则 T3 优先级最高，T1 和 T2 任务周期短的优先级高。得到 3 个任务的优先级为 T3 > T1 > T2。在 T1、T2 和 T3 同时激活时，优先级最高的 T3 执行完成后，再执行 T1 任务，见图 5。

T3 任务的响应时间：$R_3 = C_3 < D_3 = T_3$；T1 任务的响应时间：

$$R_1 = C_1 + \sum_{j \in \mathrm{hp}(i)} C_j \left\lceil \frac{R_i}{T_i} \right\rceil = C_1 + C_3 > D_1 = T_1$$

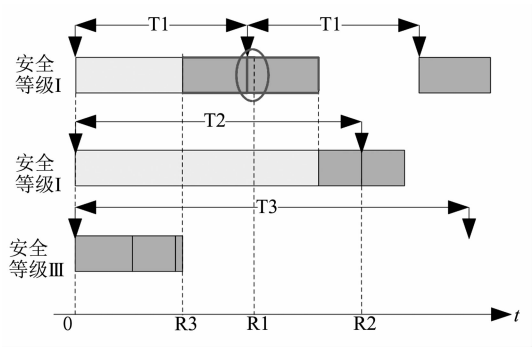

图 5 考虑安全等级调度策略的任务调度图

任务 T1 的响应时间超过了截止时间，出现不可调度。为使得 T1 满足可调度条件（$C_1 + C_3 < D_1$），必须缩短 C_1、C_2。最直接的方式是通过升级硬件平台提升 CPU 频率，同时缩短 C_1和 C_2。但是在实际项目开展过程中，软件集成已经在中后期，在嵌入式领域更改硬件平台涉及硬件和软件的重新设计等，对项目进度和质量带来极大风险。

Vestal、Steve 等提出的“转换任务周期”方法[13]，在不改变软硬件平台和任务优先级的前提下，使得任务满足可调度的要求。该方法将任务切割成多份，任务每激活一次执行其中一部分的代码，任务的执行时间 C_i得到降低；同时通过提高任务执行频率，确保任务所有内容在截止时间前执行完毕。将图 5 中 T3 任务切割成 3 份，$C_3 = C_{3_1} + C_{3_2} + C_{3_3}$；同时将执行周期缩短，$T_3 = 3 \times T'_3$。任务 T3 仍然为最高优先级，以确保所有的执行内容在截止时间 D_3前完成；同时，T1 和 T2 可以在任务 T'_3的周期间隔内抢占 CPU，转换周期后的调度见图 6。原来无法调度的 T1 任务的响应时间成为：

$$R_1 = C_1 + \sum_{j \in \mathrm{hp}(i)} C_j \left\lceil \frac{R_i}{T_i} \right\rceil = C_1 + C_{3_1} < D_1 = T_1$$

T2 任务的响应时间成为：

$$R_2 = C_2 + \sum_{j \in \mathrm{hp}(i)} C_j \left\lceil \frac{R_i}{T_i} \right\rceil = C_2 + C_{3_1} + C_{3_2} + C_1 < D_2 = T_2$$

满足可调度性要求。

3.2 任务空间验证

在空间方面，除了须要对 ROM、FLASH 和 RAM 空间的合理分配验证外，还须特别关注操作系统任务堆栈空间分配合理性的验证。随着程序运行堆栈的实际使用大小的不断变化，如果预先给堆栈分配的空间过小，会导致堆栈溢出，而堆栈溢出是嵌入式软件最不能接受的异常错误，可能直接导致系统崩溃。因此，在程序实际运行前，须对程序在最差情况下使用到堆栈的最大值进行测试验证，

确保当前分配的堆栈空间合理。

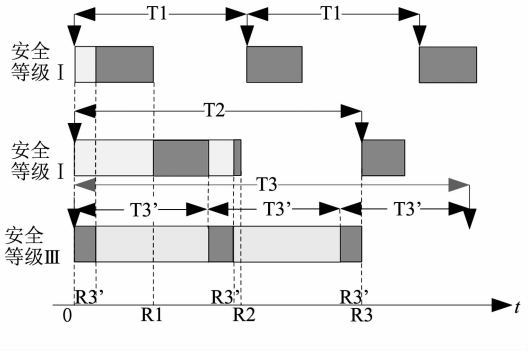

图 6　转换 T3 任务周期后的任务调度图

检测软件的堆栈最大使用值，通常有两种方式：动态测试和静态计算。动态测试通过创建测试用例实时运行软件，查看堆栈使用值和任务响应时间。但是，由于软件逻辑功能复杂，无法充分证明测试用例已经覆盖到最恶劣的软件运行情况和极限的函数调度深度。静态计算则通过配置实际运行环境（pipeline、CPU 主频等参数），分析软件的可执行代码，计算任务最大执行时间和最大堆栈使用值。目前已有成熟的商业软件支持该功能的计算。

4　总　结

本文结合船用柴油机控制软件的特性，提出了一套软件架构阶段的设计和验证流程及设计方法。在静态设计和动态设计过程中，提出了安全性设计和验证的内容及方法，以期符合船级社指南对安全性评估的要求。随着船舶柴油机智能化程度的提高，控制系统已经从单一控制单元向分布式控制系统发展，分布式控制系统中系统架构的安全性设计与验证将有待进一步研究。

参考文献

[1] Leanna Rierson. 安全关键软件开发与审定 [M]. 崔晓峰，译. 北京：电子工业出版社，2015.

[2] 中国船级社. 船用软件安全性及可靠性评估指南 [R]. 2015.

[3] 全国工业过程测量和控制标准化技术委员会. 电子/电气/可编程电子安全相关系统的功能安全：IEC61508 [S]. 北京：中国标准出版社，2007.

[4] ISO/TC22/SC32. Road vehicles-functional safety：ISO26262 [S]. 2011.

[5] NASA. Software safety guidebook：NASA-GB-8719.13 [S]. 2004.

[6] 军用软件安全性设计指南：GJB/Z 102A-2012 [S]. 2012.

[7] 徐丙凤. 构件化嵌入式软件安全性分析方法研究 [D]. 南京：南京航空航天大学，2014.

[8] 杨仕平. 分布式任务关键实时系统的防危（Safety）技术研究 [D]. 西安：西安电子科技大学，2004.

[9] FICEK C，KAI R，FEIERTAG N. Schedule design to guarantee freedom of interference in mixed critical system [C]. SAE，2012，5 (1)：46-54.

[10] ANDERSON E D. A study of safety issues in critical real time systems [D]. Department of Science of Computer Engineering in the College of Engineering at the University of Central Florida，Orlando，Florida，1990.

[11] LIU C L，LAYLAND J W. Scheduling algorithm for multiprogramming in a hard real time environment [J]. Journal of the ACM，1973，20 (1)：46-61.

[12] JOSEPH M，PANDYA P. Finding response times in a real-time system [J]. The Computer Journal，1986，29 (5)：390-395.

[13] VESTAL S. Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance [J]. In Proceedings of the 28th IEEE International Real-Time Systems Symposium，2007：239-243.