软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2014,25(2):200−218 [doi: 10.13328/j.cnki.jos.004530]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

嵌入式机载软件安全性分析标准、方法及工具研究综述*

黄志球, 徐丙凤, 阚双龙, 胡 军, 陈 哲

(南京航空航天大学 计算机科学与技术学院,江苏 南京 210016)

通讯作者: 黄志球, E-mail: zqhuang@nuaa.edu.cn, http://www.nuaa.edu.cn

摘 要: 嵌入式软件在安全关键系统中的应用,使得保障软件安全性成为软件工程领域的研究热点之一.以典型嵌入式软件系统机载软件为基础,对机载软件安全性保障的标准、方法及工具进行综述.首先,对机载软件领域所采用的软件安全性相关的标准进行简介,并给出机载软件安全性分析框架;其次,从机载软件安全性分析框架出发,将机载软件安全性保障方法划分为 3 个方面,即,机载软件安全需求的提取与规约、面向标准的机载软件开发、机载软件安全需求验证.对这 3 个方面的现有研究工作以及工业应用进行了综述;然后,针对当前适航标准的要求对机载软件安全性保证过程中软件安全证据的收集方面的研究工作进行了总结;最后,提出机载软件安全性领域存在的挑战和未来的研究方向.

关键词: 嵌入式软件;机载软件安全性;适航认证;安全性分析;软件工具

中图法分类号: TP311 **文献标识码**: A

中文引用格式: 黄志球,徐丙凤,阚双龙,胡军,陈哲.嵌入式机载软件安全性分析标准、方法及工具研究综述.软件学报,2014,25(2):200−218. http://www.jos.org.cn/1000-9825/4530.htm

英文引用格式: Huang ZQ, Xu BF, Kan SL, Hu J, Chen Z. Survey on embedded software safety analysis standards, methods and tools for airborne system. Ruan Jian Xue Bao/Journal of Software, 2014,25(2):200−218 (in Chinese). http://www.jos.org.cn/1000-9825/4530.htm

Survey on Embedded Software Safety Analysis Standards, Methods and Tools for Airborne System

HUANG Zhi-Qiu, XU Bing-Feng, KAN Shuang-Long, HU Jun, CHEN Zhe

(College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China)

Corresponding author: HUANG Zhi-Qiu, E-mail: zqhuang@nuaa.edu.cn, http://www.nuaa.edu.cn

Abstract: With the widespread use of embedded software in safety critical system, software safety assurance becomes one of research hotspots of software engineering. In this paper, a survey is presented on Software safety standards, methods and tools in aircraft systems. First of all, airborne software safety definitions and standards are introduced, and the safety analysis framework is also presented. Secondly, airborne software safety analysis methods are classified into three types, namely, software requirement elicitation and specification, safety standard oriented software development, and software safety property verification. Existing research and application of software safety analysis methods are reviewed according to these three types. After that, existing researches on safety evidence collection during airborne software safety assurance process according to airworthiness certification are summarized. Finally, potential research directions of airborne software safety assurance are discussed.

Key words: embedded software; airborne software safety; airworthiness certification; safety analysis; software tool

嵌入式软件在汽车、核能、航空等安全关键领域的普遍应用,使得软件的失效可能造成财产的损失、环境

* 基金项目: 国家自然科学基金(61272083, 61100034); 江苏省普通高校研究生科研创新计划(CXZZ11_0218); 中央高校基本科研业务费专项资金(CXZZ11_0218)

收稿时间: 2013-05-07; 定稿时间: 2013-09-29

的破坏甚至人员的伤亡,保障嵌入式软件的安全性已成为近年来软件工程领域的研究热点.机载软件是一类典型的嵌入式软件,随着计算机系统在航空航天领域的广泛应用,现代飞机几乎所有重要的系统功能都与计算机软件相关.从第二代飞机起,通过软件实现的功能随着每一代飞机而翻番.对于第三代和第四代飞机来说,软件已经成为飞行控制、通信导航、火力控制以及维修保障的核心(如军机 F-22 飞机上嵌入式系统代码高达 300 万行,F-35 机载和地面的嵌入式系统代码高达 1 500 万行[1]).软件的大量应用,使机载装备性能有了很大的飞跃,有效提高了机载装备的精确性、灵活性和快速反应能力.但是,由于软件的质量和可靠性水平远低于硬件,对机载系统的使用安全和保障产生了很大的负面影响,甚至可能造成灾难性的后果.近年来,由于软件故障引起的事故屡见不鲜,例如,2009 年,法国航空公司 AF447 航班的 A330-200 飞机由于测速仪结冰,给出了飞行控制软件错误的攀升指示,而软件中未设置高度值上限,最终导致飞机在大西洋上坠毁[2];同年,一架土耳其航空公司波音 737- 800 型飞机在机场跑道进近期间,由于无线电高度值错误使得飞机失速而机载软件缺乏“处理失速情况”的程序,导致飞机坠毁[3].可见,软件对整个机载系统安全的影响已经占据了重要地位,软件控制系统的故障和安全性问题直接影响到系统的安全性.

此外,机载软件安全关键(safety-critical)的特性,使得必须对所有机载设备软件系统以及与飞机交联的软件系统进行安全认证才能投入使用[4].这里的安全认证是指根据航空领域的安全标准对机载软件进行评估,判断其是否满足相关标准中提出的目标.目前,航空领域广泛采用的是由美国航空无线电委员会(the Radio Technical Commission for Aeronautics,简称 RTCA)提出的航空适航认证标准体系(如 DO-178B[5]以及最新发布的 DO-178C[6]).该标准体系规定了软件开发过程中的各阶段软件制品所要达到的安全目标,对机载软件系统的安全性提出了严格的要求.为了使机载软件符合适航认证标准体系的要求,需要对特定安全目标的安全证据收集技术和方法进行深入研究,以便通过第三方的适航认证.

目前,机载软件安全性的研究主要分为以下 5 个方面:危害分析[7]、软件安全需求的提取与分析[8]、软件安全性设计[9]、软件安全验证和确认[10]、软件安全认证及标准[11].已有的研究工作分别针对机载软件安全性研究的各个方面进行了相关研究工作,但是仍缺乏研究工作从软件工程的角度对机载软件安全性分析整个过程中的理论研究工作以及相关标准和成功工业应用进行综述.在这个背景下,本文采用了美国通用适航总局所给出的机载软件安全性分析框架,从软件工程的角度对软件安全性分析的工作进行了划分,并且对相关的理论研究、成功的工业应用以及安全标准进行了综述.

本文第 1 节介绍机载软件安全性的理论、相关标准以及本文综述的总体思路.第 2 节对机载软件需求提取以及规约的研究工作进行总结.第 3 节对面向标准的软件开发及验证过程进行概述.第 4 节对机载软件安全需求验证方法方面的工作进行讨论.第 5 节对面向安全认证的需求可追踪性及其证据收集方法进行介绍.最后对全文进行总结,并提出未来的研究方向.

1 机载软件安全性标准及分析框架

1.1 机载软件安全性及相关标准

软件安全性,是指软件运行不引起系统危害的能力[12].软件作为系统的重要组成要素,不会直接危及生命、财产和环境等安全,但借助软件实现的人机交互却可能因软件失效造成人员误操作,从而形成危害.对于无人机交互的嵌入式安全关键系统而言,也可能因为软件错误地控制系统而造成灾难性的后果.以机载软件为例,飞机低于特定高度时,着陆轮控制系统必须控制机载系统放下着陆轮,否则容易造成飞机坠毁.当软件可以引起危害或者控制危害的发生时,该软件是危险的[13].软件安全性(software safety)一词最早出现在 1979 年美国发布的 MIL-STD-1574A[14]中,1986 年,MIT 大学的 Leveson 教授提出,软件的安全性在系统的全生命周期中应该作为一个单独的课题加以重点研究[15].目前,各个学术机构都提出了对软件安全性的定义,如,NASA 8719.13A 中指出,软件安全性是指在软件声明周期内,应用安全性工程技术,确保软件采取积极的措施提高系统安全性,确保降低系统安全性的错误已经减少到或控制在一个风险可接受的水平内[16];Leveson 指出,软件安全性是指确保软件在系统上下文中执行不会导致系统发生不可接受的风险[15];此外,GJB 900-90[17]以及 GJB/Z 142-2004[18]中也对

软件安全性的过程和方法进行了阐述.

对于机载软件而言,为保障其安全性,还具有适航方面的要求.适航性(air-worthiness)作为航空器能够在预期环境中正常飞行的固有品质,反映了公众对空中活动进行管理的需求,以保证公共利益[19].这种品质要求航空器必须符合型号设计并处于安全运行状态,对航空器的安全性(safety)、可靠性、可维护性、可测试性以及综合保障性都提出了较高要求.各国家地区的航空管理部门先后出台了带有法规性质的适航审定标准,如美国航空管理局(FAA)的 FAR、欧洲航空安全委员会(EASA)的 JAA、中国民航总局(CCAC)的 CCAR 等,这些适航审定标准为航空器提供了最低安全标准.在适航审定过程中验证飞机设计、制造符合标准,并以相应的适航符合性验证流程评估飞机安全性,是适航审定能否有效开展的关键所在.为了保障机载软件安全性,在产业界和学术界已经形成了一系列相关的标准和研究成果.图 1 是我们总结出的前美国航空管理局所提出和采纳的机载软件安全性相关的标准及其之间的关系.在适航标准中,适航标准 FAR 2x.1309 就机载软件的可用性和无错性提出了原则性要求.咨询通告 AC 2x.1309 针对上述要求,从系统的分析与设计层面提出了进一步细化的目标与方法建议.在咨询通告 AC 20-115B[20]和 AC 20-171[21]中,就机载软件的符合性验证提出了明确标准.咨询通告 AC 20-148[22]则专门就机载软件的可重用性进行了规定.在业界规范中,SAE ARP 4754/4761 标准在系统层面提出了针对 FAR 2x.1309 的可实施的流程与方法学指导.RTCA/DO-178B 从软件层面提出了与 SAE ARP 4754[19]相对应的目标与方法,并被 AC 20-115B 所采纳.而 RTCA/DO-178C 在 DO-178B 的基础上增加了 DO-330(软件工具验证)、DO-331(基于模型驱动的开发和验证)、DO-332(面向对象编程)、DO-333(形式化方法)等相关目标的规定,将模型驱动与形式化方法明确为对机载软件进行分析与验证的推荐手段.在我国的适航审定中,也分别在 CCAR 2x.1309 和对应的咨询通告“机载系统和设备合格审定中对软件的考虑”中对软件的适航符合性验证提供了初步的标准.

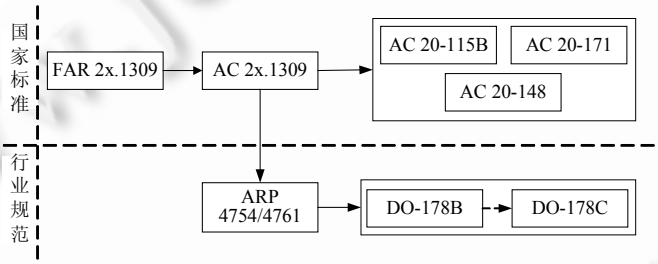


Fig.1 Related FAR standard and norm of software safety

图 1 FAR 软件安全性相关标准与规范

1.2 面向安全性的软件分析框架

针对机载软件的高安全性要求,必须采用全面的软件安全性保障方法,从系统开发的各个阶段为机载软件的安全性保证提供支持.当前,国内外学者都对软件安全性的相关理论和方法进行了总结,如,Lutz 针对软件工程中的安全性问题中危害分析、安全需求的规约和分析、安全设计等 6 个软件安全相关的研究领域的研究现状进行了概述,并且提出了未来需要深入研究的方向[23];Hauge 给出了软件安全性在系统安全性中的相关描述以及软件安全性的分析方法综述[24];McDermid 等人给出了有关软件安全性当前所存在的一些争议,并且针对当前软件安全性研究以及工业实践相关标准中的问题进行了总结[25];Mekikonda 等人基于 McCall’s 软件质量模型提出了一种新软件安全性框架,这种软件安全性分析框架可用于系统危害分析、需求完整性、基于安全约束的设计、运行时问题管理以及软件安全关键测试[26];McDermid 对当前开发和应用证据框架的研究工作进行总结,并且提出了证据框架普遍应用的难点所在[27].国内学者分别针对软件安全性研究方法[28]、安全关键软件的测评[29]、软件质量保障[30]等几个方面对软件安全性的相关理论以及技术进行了综述.这些综述都是从软件安全性的理论层面进行综述,而从软件工程的角度针对在实际工业应用中所采用的软件安全性分析框架以及对应的标准和方法,目前缺乏相关的文章对其进行总结.图 2 是我们根据美国通用适航技术总局提出的以机载软

件为代表的安全关键软件系统安全性分析流程基础上总结出的面向安全性的机载软件安全性分析框架.从图 2 可以看出,为了保障机载软件的安全性,需要从机载系统和机载软件自身两个部分出发:一方面,从识别系统的危害和故障模式出发,进而识别软件造成的风险,并且从中获取软件的安全需求,支持软件设计以及提供安全验证的需求;另一方面是从软件执行功能分析出发,根据适航认证标准确定软件的安全等级,并根据适航认证标准分配相应的软件保证目标,在软件开发的过程中通过达到这些安全目标来保障软件的安全性;此外,在整个软件开发的过程中,必须对所生成的软件制品进行验证,保证所有的软件制品都满足安全性需求.从图 2 可以看出,机载软件的安全性分析框架可以划分为 3 个部分,分别是机载软件安全需求的获取及规约、面向标准的机载软件开发过程、机载软件的安全需求验证.

本文从图 2 给出的机载软件安全性分析框架的 3 个部分出发,即:(1) 机载软件安全需求的获取及规约;(2) 面向安全标准的软件开发过程;(3) 机载软件安全需求验证,从这 3 个方面对机载软件的安全性分析方法、工具进行综述,并且在此之后,针对机载软件软件开发的过程管理以及安全证据收集方面的研究成果进行总结.

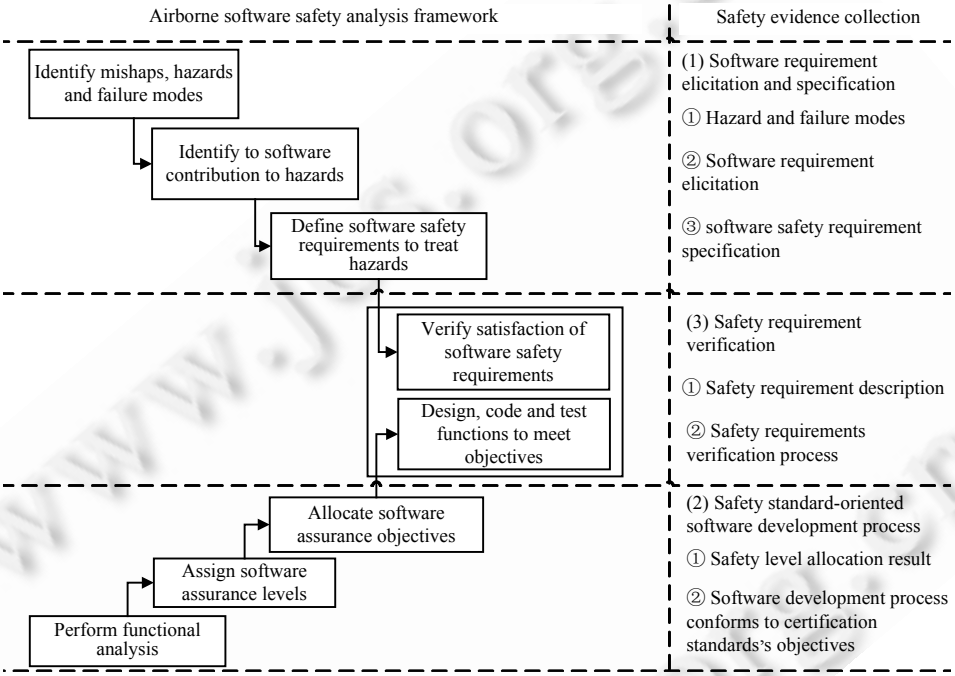


Fig.2 Safety analysis framework of safety critical software

图 2 安全关键软件系统安全性分析框架

2 软件安全需求的提取与描述

本节对机载软件开发及分析框架中的第 1 部分,即安全需求的获取与规约方法进行综述.首先,对系统安全性分析方法进行阐述,并且对当前广泛采用的安全性分析工具进行介绍;在此基础上,对软件安全需求的规约方法方面的研究工作进行总结.

2.1 软件安全需求提取

软件安全需求是指,通过约束软件的行为,使其不会出现不可接受的违反系统安全的行为[31].软件安全需求的获取是根据已知的系统信息,如软件危险条件等以及其他一些类似的系统数据和通用惯例,完成通用软件安全性需求的裁剪和特定软件安全性需求的获取工作[32].软件安全性人员与系统安全性人员密切合作,负责从相关标准、系统安全性需求、环境和设施需求、接口需求、系统危险分析、相关经验等方面获取软件安全性需

求,分析之后作为软件设计的依据.其中的软件安全性需求应该既包括与系统安全性相关的功能需求,又包括造成系统不安全行为的软件需求,而且还包括预防系统进入不安全状态的软件需求或设计要求.正确且充分地获取软件安全性需求,是软件安全性工作的核心环节,是后续软件开发和开展安全性相关工作的依据.获取软件安全需求可以与系统危险分析一起进行,或者在系统危险分析之后立即进行.该工作是不断迭代的,即,随着软件分析、设计工作的深入开展,软件安全性需求将得到不断的补充.这项工作可一直持续到软件测试阶段.

软件安全性需求的确定[33,34]是整个软件安全性工作的关键,它是开展后续软件安全性设计、实现与测试工作的依据.软件安全性问题多数情况是由于软件安全性需求获取不全导致的,如对环境失效和各种异常情况的处理考虑不完善等.软件的安全性需求分析工作的最终目的是获得相对完整、正确的软件安全性需求集合,通常需要采用安全性分析方法.通过该方法,安全分析人员对于特定危害发生的所有可能途径和因素进行分析和建模,得到危害场景模型.当前的安全性分析方法主要分为两类:一类是归纳方法,即从原因归纳结果;另一种是演绎方法,即从结果触发追溯原因.

在系统安全性分析当中,归纳方法主要应用表格式的方法用于发现危害或者特定操作条件或失效的危害结果.预先危害分析(preliminary hazard analysis,简称 PHA)[35]也称为初始危险分析,是一种表格形式的安全性分析方法,在一个系统或子系统(包括设计、施工、生产)运转活动之前,对系统存在的危险类别、出现条件及可能造成的结果做宏观的、概略的分析.失效模式和影响分析(failure mode and effect analysis,简称 FMEA)[36]也是一种表格形式的安全性分析方法,可用于确定潜在失效模式及其原因,能在产品设计或生产工艺真正实现之前发现产品的弱点.FMEA 可以应用在系统或者构件层,当前已经在航空和汽车行业广泛使用.另一种表格形式的分析技术是危害与可操作性研究(hazard and operability studies,简称 HAZOP)[37].该技术最早出现的化学领域,该方法又分为基于引导词的 HAZOP、经验式的 HAZOP、检查表式 HAZOP 以及基于模型的 HAZOP,能够系统地分析流程潜在危害的分析方法.这些表格形式的技术具有结构化的特征,而且能够使用自然语言灵活地解释各种情形,因此在工业应用中,这些表格形式的技术比形式建模技术更容易被采用.不过,采用这类方法无法对危害的概率等进行准确的计算.另外,由于都是用自然语言的描述,这些方法的自动化实现起来也比较困难.此外,事件树分析(event tree analysis,简称 ETA)[38,39]也是一种常用的归纳式安全性分析方法.事件树从作为危险源的初始事件出发,根据后续事件或者安全措施是否成功作分支,最后到灾害事件的发生为止.

演绎形式的安全性分析方法比较有代表性的包括可靠性框图(reliability block diagram,简称 RBD)[40]和故障树分析(fault tree analysis,简称 FTA)[41,42]:

- RBD 是系统单元及其可靠性意义下连接关系的图形表达,表示单元的正常或失效状态对系统状态的影响.RBD 利用互相连接的方框来显示系统的失效逻辑,分析系统中每一个成分的失效率对系统的影响,以帮助评估系统的整体可靠性.
- FTA 也是当前工业界应用最广泛的安全性分析技术.它描述系统中构件失效和整个系统失效之间的关系,顶事件表示危害情形,通过逻辑门与基本事件表达危害发生的因果链,可对顶事件发生的概率及其他定量指标进行分析.

由于 FT 无法表达动态系统中的时间依赖以及序列依赖的行为,已有一系列工作对其建模能力进行扩展,以支持动态系统以及实时系统的建模及分析.主要包括:动态故障树(dynamic fault tree,简称 DFT)在标准故障树的基础上扩展了能够描述复杂系统行为与交互的动态门[43];时序故障树(temporal fault tree,简称 TFT)基于连续逻辑[44]、区间时序逻辑[45]等描述系统中的时序关系,并针对 TFT 进行定性分析,考虑可达性、可满足性等;构件故障树(component fault tree,简称 CFT)[46]和状态事件故障树(state/event fault tree,简称 SEFT)[47]在动态故障树的基础上扩展了构件的概念,用于进行构件化系统的安全性建模与分析.此外,还存在一些研究工作对故障树进行逻辑门扩展[48,49]以及对模型本身进行扩展[50,51].

对于演绎式的安全性分析方法,如故障树,目前已经存在一系列的基于故障树模型的软件工具.目前,在工业界和学术界最常用故障树工具包括 Galileo dynamic fault tree analysis tool[52],Relex fault tree analysis software[53]和 Fault Tree+[54]:

- Galileo 是一个由 Virginia 大学开发的动态故障树建模和分析工具.Galileo 组合了模块化的动态故障树分析方法,Gaileo 的主要建模和分析特征包括:① 自动模块化故障树并且对模块进行独立,静态故障树使用基于 BDD 的方法,而动态子树使用 Markov Chains 的方法;② 多种失效时间分布(确定概率、指数分布、Weibull);③ 对静态和动态子树的不完善故障覆盖建模;④ 阶段任务建模和分析;⑤ 进行构件重要性分析.
- Relex fault tree analysis software 同时支持定量和定性分析,基于用户的需求提供计算灵活性.Relex fault tree analysis tool 可以计算系统不可靠性、失效频率、失效数等,此外还提供了计算最小割集的方法.它是仅有的能够支持动态故障树精确分析的商用软件包.
- Fault Tree+软件是由英国 ISOGraph 公司开发的故障树分析工具,允许在综合集成环境下进行故障树、事件树、马尔可夫分析.

2.2 软件安全需求的形式化规约

获取软件的安全需求之后,需要采用合适的表达方式对软件的需求进行描述.通常,在需求工程中的需求描述方式分为 3 种:自然语言描述符方式、半形式化描述方式以及形式化描述方式.由于自然语言的二义性以及模糊性,采用自然语言描述软件安全需求通常会导致软件开发人员理解方面的偏差,因此,在安全关键软件的需求描述通常采用半形式化模型或者形式化模型.

半形式模型是工程人员所熟悉的表达方式,比较方便工程人员的使用.目前,常用的对软件安全需求进行描述的半形式化模型包括 FTA、SysML 和 UML 当中的模型图以及 FHA 等方法.但是,由于半形式化模型缺乏准确的语义,在软件开发的过程中为了针对软件安全需求进行验证,必须采用形式化的描述方法描述软件的半形式化以及自然语言描述的安全需求,如采用形式化语言自动机及其扩展模型、Petri 网及其扩展模型、进程代数、本体、LTL、CTL 等形式化模型.目前,已经存在很多这部分的工作,如,文献[55]提出了一种基于 HAZOP 和 FHA 场景的安全关键系统需求获取方法;文献[56]介绍了一种基于 FTA 和 FMEA 的软件安全需求的获取方法,并且采用状态机对最终获得的软件安全需求进行表示;文献[44]提出了一种从 FTA 中获取采用时序逻辑形式描述的软件安全需求的方法;文献[57,58]基于 FTA 抽取软件安全需求作为模型验证的安全属性;文献[59]采用 SysML 需求图进行系统安全需求到软件安全需求的分解及表示.

形式化需求规约[60]有助于准确而无二义地理解和描述系统,并且可以支持规约的形式化检测.在软件的安全需求获取过程中,为了保证安全关键软件的高安全性要求,通常最终采用形式化的软件安全需求描述方法,方便软件开发及验证过程中对软件安全需求的检查.直接采用形式化语言对软件的安全需求描述方面也存在一系列的研究工作,如,文献[61,62]提出了一种基于形式化语言的软件安全需求的表示及分析方法;文献[63]针对核能系统保护系统提出了一种新的形式化软件需求规约方法 NuSCR;文献[64]针对航电系统构建其软件需求知识本体,并且对其完整性、一致性等进行了分析;文献[65]提出了基于本体的软件安全性需求建模和验证方法;文献[66]提出了基于知识的需求获取方法.

但是总体而言,软件的安全需求的获取与规约方面的工作仍然存在不足:一方面,由于软件的安全性分析方法目前都是继承自传统的系统安全性分析方法,因此在描述软件的特性方面尚存在不足,导致在软件安全性需求分析的过程中会造成部分信息的丢失;另一方面,如何从系统的安全需求里面快速而有效地提取软件的安全需求,目前还缺乏有效的手段.此外,当前在安全关键应用中应该尽可能地推广形式化方法的应用,得到准确而无二义的软件安全需求是保证软件安全性的关键.

3 面向标准的机载软件开发过程

在机载软件开发的过程中,首先依据软件架构对软件的各个模块进行软件等级的分配,如何分配需根据系统安全性分析的结果进行.在分配软件安全等级之后,根据适航认证标准中对应等级的目标对软件模块进行开发,并且在此过程中采用测试、仿真、验证等方法对软件的安全需求进行验证.本节首先对机载软件的安全等级分配的相关规定以及研究工作进行总结,在此基础上,给出了面向标准的软件开发过程以及相应的验证方法.

3.1 基于架构的软件等级划分

目前,在机载软件适航认证的过程中普遍使用的国际标准是航空无线电技术委员会(RTCA)发布的 DO-178B,该标准定义了机载软件开发过程当中各阶段软件制品所要达到的安全目标.实际上,并非所有的机载系统软件构件对飞机及乘客安全的影响都相同.例如,控制飞机高度的软件失效比存储和显示飞机地理信息的软件失效后果更严重,因为前者严重影响空中飞行及着陆阶段的安全,而后者虽给飞行员带来不便,但可以通过其他方式读取数据来消除影响.适航认证标准关注软件失效对系统安全性的影响,并将这些影响分成了 5 类:灾难性的、危险的/严重的、较重的、较轻的、无影响的[5].其中,失效状态为灾难性的,即,阻止航空器继续安全飞行和着陆的失效状态.失效状态为危险/严重的,即,降低航空器的性能或者机组人员克服不利操作状态的能力的失效状态.失效状态为较重的,即,可能降低航空器的性能或机组人员克服不利操纵状态的能力的失效状态.失效状态为较轻的,即,不会严重降低航空器安全性及又有机组人员的活动在他们的能力内能很好地完成其活动的失效状态.失效状态为无影响的,即,不影响航空器的工作性能或不增加机组人员工作量的失效状态.

软件等级是根据在系统安全性评估过程中确定的软件对潜在失效状态的影响来定义的,对于每个分类,存在着不同的认证等级,见表 1,分别为 A,B,C,D,E.其中,A 是最严格的软件等级,需要提交最多的文档用于证明软件制品与适航标准当中所规定认证目标的一致性;B 次之;随后是 C 和 D;E 表示对飞行安全没有影响.与之类似,应用于工业领域的国际标准 IEC61508 将软件分成 SIL1,SIL2,SIL3,SIL4 这 4 个安全集成等级[67].

Table 1 Software level of DO-178B

表 1 DO-178B 中的软件等级

Level	Condition	Effect of anomalous behavior
A	Catastrophic failure	"…prevent continued safe flight and landing…"
B	Hazardous/Severse failure	"…serious or potentially fatal injuries to a small number of … occupants…"
C	Major failure	"…discomfort to occupants, possibly including injury…"
D	Minor failure	"…some inconvenience to occupants…"
E	None	"… no effect on aircraft operational capability or pilot workload…"

DO-178B 中未明确说明软件开发和保障过程如何执行,但是规定了这些必须包含特定的活动,如走查、测试必须产生特定的文档,开发和确保多方面的计划、需求和设计的描述,且需求和代码测试之间必须满足可追踪性.DO-178B 描述了 66 个如上述类型的目标,软件等级为 A 必须满足所有的目标,软件等级为 B 必须满足其中的 65 个目标,软件等级为 C 必须满足其中的 57 个目标,软件等级为 D 必须满足其中的 28 个目标.系统安全性评估过程要先确定与特定系统中的软件部件有关的软件等级,而不考虑系统设计.当确定软件等级时,必须标明失效的影响.

目前,针对软件的安全等级的分配方面已经存在一系列的研究工作.文献[68]提出了一种使用模糊风险图和收集不精确和不确定的专家意见的主观评判系统的软件集成等级分配的定量方法.文献[69,70]基于已长期在大规模系统开发中应用的软件可靠性增长模型,提出了一种计算软件安全集成等级的方法.文献[71]提出了基于安全性的分析方法 Hip-HOPS,为提供组合安全关键功能的复杂架构网络子系统和构件分配安全集成等级的方法.文献[72]提出了一种新的为构件分配软件安全集成等级的方法,将安全集成等级的分配问题作为整数线性规划问题解决,并且允许根据特定的目标(包括来自 PHA,FTA 的约束以及安全工程师的偏好)对安全等级的分配进行优化.

3.2 基于模型的软件开发过程

在机载软件开发过程中,必须满足适航认证标准中的对应的目标.DO-178B 制定了软件生命周期各个过程的目标,阐述了达到目标所需进行的活动.DO-178B 规定,在获得分配给软件的系统需求之后,具有以下 4 个开发过程:① 软件需求过程.在该过程中,将从系统需求中提取软件高层需求,包括软件的功能、性能、接口和与安全性有关的需求等.② 软件设计过程.该过程负责设计软件的低层需求以及体系结构.③ 软件的编码过程.对低层需求进行实现,得到软件的源码.④ 软件集成过程.这一过程把源代码集成、编译并链接成为在目标环境上执行的代码.DO-178B 规定了在每一个过程中软件的安全目标,在软件开发的每个阶段,必须满足适航认证标准

所规定的认证目标.

目前,在软件安全性开发领域广泛采用基于模型的开发方法,最新的标准 DO-178C 中添加了针对基于模型开发方法应用的相关规定.模型驱动的方法强调模型在软件系统开发过程中的主体地位,以模型来引导开发,针对像机载系统这样需求复杂、安全关键的软件,通过建立规范的高层系统模型及文档,以模型来引导开发,可以为各个开发阶段提供全局统一的视图和指导.目前,国内外学术界和产业界已经对基于模型的开发方法进行了大量相关研究.

在理论研究方面,文献[73,74]总结了国际国内目前对于 MDA 的研究现状,指出了目前 MDA 领域中存在的问题和发展的道路,并且研究了 UML 扩展机制中衍型(stereotype)的精确定义,在此基础上给出了应用指导.文献[75]总结了目前模型转换技术的研究现状并对其进行了分类,并且介绍了一种模型转换的编织架构,提出用代码生成的方式解决模型转换语言之间的异构性.文献[76]介绍了一种基于 MDA 的 UML 顺序图到状态图的转换方法.文献[77]针对嵌入式软件的体系结构模型转换问题进行了研究.文献[78,79]对基于模型驱动的方法的基本理论进行研究,并讨论了相关方法在工业界的应用机制.

在产业界,法国 Esterel 公司的 SCADE 工具集提供了支持适航审定标准 DO-178B 的认证方法和技术,MathWorks 以及波音、空客等公司也都开发了一系列针对适航标准符合性验证的工具和产品.这些产品在欧洲 EC135 以及 EC155 系列直升飞机的自动飞行系统、空中客车 Airbus A340 和 A380 的控制系统以及波音 Boeing 787 的着陆导航系统等安全关键系统中得到了良好的应用[80,81].此外,比较典型的几个在航空领域应用的基于模型驱动的软件工具包括 TOPCASED[82],SPICES[83],ASSERT[84].TOPCASED 是由空中客车公司提出并实际应用的项目,采用模型驱动开发和形式化验证结合、多种开源工具集成的思想,支持复杂嵌入式实时系统的设计、开发与实现,以保证系统的质量属性并降低开发时间和成本.SPICES 是由欧盟 EUREKA-ITEA 支持的项目.针对航空工业界具体应用,提出对 AADL 语言进行扩展,以支持一些新概念和属性的描述,如功耗约束、对 ARINC653 标准的支持等,并标准化为 AADLV 2.0,研究成果应用于空中客车的下一代航空电子原型系统、法国地球观测卫星的嵌入式控制器、Thales 公司的某个航空设备等系统中.ASSERT 是欧盟支持的项目,目标是对传统嵌入式实时系统的开发过程进行改进,在生命周期中的每一个环节,都需要进行严格的确认和验证(validation and verification,简称 V&V)才能进入下一个环节.此外,国内也出现了商业化的 MDA 工具,由楚凡科技开发的 MDA 工具 Trufun Kant,是中国第一个基于 UML 的模型驱动架构开发工具,也是全球第一款中文 MDA 开发工具. Trufun Kant 覆盖了软件开发的各个环节,不仅可以通过 UML 进行需求捕获、系统分析、系统设计,同时可以进行代码生成、编译、调试、运行、打包和部署,以模型驱动整个软件开发,真正实现了以模型为中心的软件开发新模式.

3.3 软件验证技术在开发过程中的应用

传统的机载系统开发采用开发文档管理等方法,从工程化的角度来保障软件安全性,并且在研发中后期采用评审、仿真测试等方法来保障软件的安全性.在基于模型的软件开发过程中,在整个软件开发的过程中都必须根据软件所分配到的软件安全等级来决定采用哪种软件验证方法对软件开发过程中的所得到的软件制品进行验证.常用的软件制品验证方法包括:① 静态分析,以检查单或类似辅助手段为指导的定性分析;② 测试,开发并采用测试用例对给定的输入检测软件正确性;③ 模型检查,自动地对系统的功能属性定量地可重复性验证,主要包括模型检查、运行时检查等方法.这几种验证方法的优缺点总结见表 2,在具体的软件开发过程中,需根据软件等级决定采用对应的软件验证方法对软件制品进行验证.

总体而言,面向标准的软件开发过程包括两个方面:一方面,通过对软件等级划分得到软件等级;另一方面,根据软件的安全等级得到软件开发过程中所必须达到的安全目标.此外,为了检查对应安全目标是否已经达到,必须根据软件等级选择对应的软件验证方法对安全目标进行验证.图 3 给出了完整的面向标准的软件开发以及验证过程.

在面向标准的软件开发过程中,一方面,基于模型驱动的软件开发方法目前在自动化方面仍需进行进一步的研究;另一方面,基于软件体系结构的软件等级划分目前还未形成统一的划分和检测方法.此外,当前在验证

各个软件是否达到软件等级要求的目标方面,在对应的标准中,针对每一个软件等级都规定了对应的安全目标,但是并未规定实现这些目标所需采用的技术方法以及具体的流程,使得工程人员在具体的实施起来比较困难.

Table 2 Advantages and disadvantages of software verification methods

表 2 软件验证方法的优缺点

方法	优点	缺点
静态分析	易发现浅层次错误,适用性强	自动化程度低,路径覆盖率低,易漏查深层次错误,不能证明无错
测试	有一定自动化程度,技术门槛低,适用性强	不完全自动化,仅部分路径覆盖,易漏查并发错误,不能证明无错
模型检查	自动化程度高,全路径覆盖检错能力强,可发现深层次错误,可证明无错	状态空间爆炸,基于系统简化模型,技术门槛高

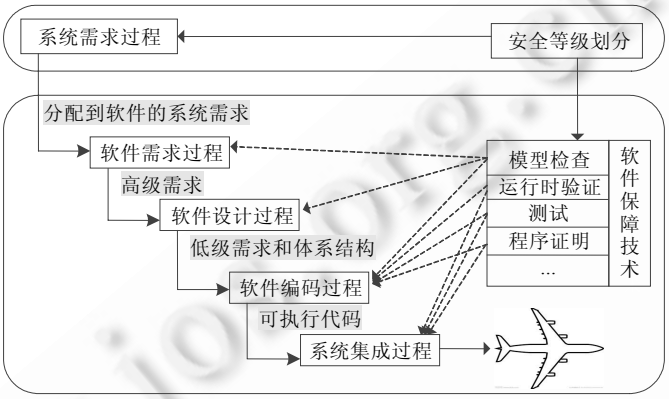


Fig.3 DO-178B development process

图 3 DO-178B 开发过程

4 软件安全需求形式化验证方法

在软件开发的各个阶段,一方面需要根据软件等级所规定的安全目标对软件制品进行验证,另一方面,需要针对从系统需求中获取的软件安全需求对各个阶段的软件制品进行验证.由于在最新的适航认证标准 DO-178C 中新增了基于模型的开发和形式化方法方面的内容,且当前在工业应用中采用形式验证已经成为安全关键软件的安全性的主要保障技术,因此,本节对基于模型开发的形式验证技术的理论以及工具进行了总结.

基于模型的形式化验证技术与代码审查、仿真和测试等方法相比,可以在开发早期对设计模型进行验证,尽早发现错误,并对模型自动生成的代码进行严格分析和验证,多层次纠错.系统验证的一般过程如下:给出一个系统以及该系统必须满足的规约,通过某种技术验证该系统是否满足给定的规约,而且验证过程一般是自动的.使用形式验证的好处在于,可以准确地描述系统的正确性和安全性性质,验证过程不须人工干预,错误定位非常准确等.形式化验证技术基于全路径覆盖的穷举式检查,易于发现软件中的算法错误、并发错误等隐藏缺陷,并且可以极大地提高验证的覆盖率和可信度.欧洲宇航防务集团(EADS)、法国空中客车公司(Airbus)、法国国家科学研究中心(LAAS-CNRS)、法国国家航天航空研究中心(ONERA)等著名研究机构联合开展了对航空航天软件的形式化验证技术研究,并在相应的航空器生产中进行了实践[85,86].表 3 中列出了部分国外机构采用模型驱动的开发和验证技术进行航空产品研发的情况和效果[87].由该表可以看出,工业界目前已经对形式验证技术进行了成功的应用.

图 4 给出了基于模型驱动的软件开发过程中的软件安全需求形式验证方法的示意图,主要包括模型检验、运行时验证、程序证明.目前,国内外在理论和工具方面都已有一系列的研究.文献[88]对于模型检查的基础理论展开了详细的论述.文献[89]对运行时验证展开了深入研究,并提出了相应的理论方法.美国喷气推进实验室(JPL)的 Holzmann 等人设计了 SPIN 模型检查器,对火星探测器的控制系统进行了验证[90].国内学者近年来也对

相关理论进行了持续关注,中国科学院的学者对模型检测的理论、方法和应用进行了全面的研究和综述[91].北京航空航天大学的学者对混成 CPS 系统的理论和应用进行了综述[92].北京大学的研究团队对运行时验证的理论展开了探索[93].南京大学研究团队对混成系统中的模型检测理论开展了大量的研究[94].南京航空航天大学的团队对基于场景规约的形式化分析与验证理论方法[95]开展了部分相关工作.此外,近年来,国内研究者也对形式化方法与领域特征的结合给予了关注.文献[96]采用形式化方法对铁路系统的基于动作序列的软件行为的需求模型进行了验证.文献[97]针对汽车行业的软件系统进行形式化分析与验证.文献[98]采用 Petri 网模型对机载系统中除冰软件系统的形式化分析与验证.文献[99]采用形式化方法对机载软件安全等级依赖的合法性进行了分析.文献[100]采用模型验证工具对飞机襟缝翼控制系统进行了形式验证,并且发现了其中的软件错误.文献[101]阐述了欧空局(ESA 资助的项目 COMPASS)对形式化方法在安全关键领域的应用,并且综合采用形式化方法进行形式化建模、形式规约以及安全性验证分析.但是,由于模型检测通常是穷尽搜索模型中的所有状态,当模型所包含的变量增多或者变量的值域范围变大时,都会导致状态空间的爆炸.当前,虽然存在一些方法可以在某些特定条件下的状态空间进行约简,但模型检测中仍缺乏有效的通用方法来处理大部分条件下的状态空间爆炸问题.定理证明[102]利用逻辑系统对软件系统进行描述,通过基于公理或者推理规则组成的形式化系统进行系统具有安全性质规约验证.目前,定理证明的研究主要是提高定理证明的自动化程度和证明的效率,包括基于归结的方法[103,104]、基于表的方法[105]、自然演绎法[106]以及基于扩展规则的方法[107].定理证明方法可以用来验证无限状态系统,但自动化程度相对较低,要求使用者掌握大量的逻辑知识,对逻辑推理有深入的了解,限制了定理验证的使用范围.

Table 3 Situation and effectiveness of the model-driven development and verification technology applied in some foreign institutions

表 3 部分国外机构采用模型驱动的开发和验证技术的情况和效果

公司名称	产品	模型驱动和验证技术使用率	效果
空中客车	A320 客机 A340 客机 A380 客机	70%的电传操纵控制系统 70%的自动飞行控制系统 50%的计算机显示装置	软件错误率降低 20 倍 开发周期缩短
欧洲直升机公司	EC-155/135,自动驾驶系统	90%的自动驾驶系统	开发周期缩短 50%
通用电气&洛克希德·马丁	FADEDC,发动机操控装置	使用率数据未公开	开发周期缩短 50%
美国太空武器公司	DCX 火箭	使用率数据未公开	开发成本降低 50~75% 开发周期和风险降低
霍尼韦尔商业航空系统	Primus Epic,飞行控制系统	60%的自动飞行控制系统	开发效率提高 5 倍 编码错误率低于 0.1% 通过 FAA 认证

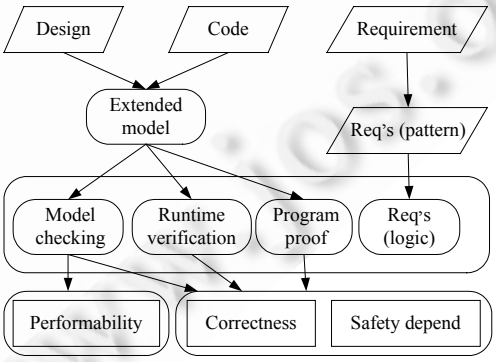


Fig.4 Formal verification methods of software

图 4 软件形式化验证方法

目前,基于模型的形式验证方面已经存在一系列比较成熟的工具,包括 SMV[108],Spin[109],UPPAL[110],

CBMC[111],BLAST[112],SLAM[113]等.其中,SMV 是使用符号模型算法来验证 CTL 公式的程序.它使用有限状态自动机作为建模语言,模型和规约被描述在输入文件中,模型描述语言是一个简单的并行赋值.Spin 是支持异步进程的设计和验证的系统.它关注证明进程交互的正确性,进程交互在 Spin 中可以通过同步原语、管道的异步通信,共享变量来实现.UPPAL 是一个对实时系统进行建模、确认和验证的综合工具平台.它将实时系统建模为时间自动机,并扩展其数据类型包括有界整数、数组等.CBMC 是一种可以验证 ANSI-C 和 C++语言的模型检测器.它可以验证数组越界、缓冲区溢出、指针安全、异常和用户自定义断言等,还可以验证 ANSI-C 和 C++代码是否与模型相匹配.BLAST 也是基于模型检测的验证 C 语言的模型检测器.它使用谓词抽象对 C 代码进行抽象,提供对程序的自动化验证.对于一段 C 语言程序,它证明程序是满足预先设定的安全属性,或者找到一条违反安全属性的执行路径,可以供开发者进行分析.SLAM 是一种基于模型检测的形式化验证工具.它用于检测软件接口是否满足规定的关键行为属性,帮助开发人员设计出高可靠和功能符合需求的软件系统.它使用了谓词抽象对程序状态空间进行约简.运行时验证方面目前也存在很多工具,如 Tracematches[114],JavaMOP[115],LARVA[116]等.在定理证明方面,PVS 是原型验证系统(prototype verification system)[117].

虽然当前软件验证技术已经在工业界进行了部分应用,但是形式化方法本身的复杂性对工程人员提出了很高的要求,使得当前普及使用很困难.另外,形式化验证大部分基于系统状态执行,使得状态空间爆炸的问题在软件系统的形式验证过程中显得尤其突出,需要采用安全关键软件的特点来进行特定方面的软件状态空间的约简.

5 面向安全认证的需求可追踪性研究及其证据收集

在介绍了软件安全需求的获取与描述、面向标准的软件开发过程和软件安全需求验证方法这 3 方面的软件安全性分析方法之后,本节对软件安全性过程中根据机载软件的适航认证标准的要求获取合适的软件安全证据的方法和技术进行综述.这些软件证据将成为机载软件通过相应的适航认证的前提.

我们首先给出安全性认证的基本概念.认证工作一般由第三方开展,通过一系列验证活动确认产品遵循了相应的设计标准,能够满足对应的安全需求.在进行机载软件安全性保障的同时,收集软件证据说明软件符合给定的安全性需求,或者说明软件满足特定的适航标准要求,是进行机载软件安全认证的前提.目前,针对软件适航认证方面国内外已经存在一系列的研究.文献[118]针对民航领域的软件安全的适航认证问题提出了一种静态的软件安全审计方法,用于支持机载软件的适航认证.文献[119]针对民航工业的不同方面的安全性问题进行了综述.文献[120]针对软件适航提供案例进行了研究.国内近年来也对适航审定进行了初步的探索研究,如,文献[28]对软件安全性及其相关安全性保障方法进行了综述,文献[121]将 DO-178B 和 GJB5000A 进行了比较.

证据收集作为适航认证的前提,可以分为两个方面:① 软件开发过程的证据收集;② 软件安全需求验证过程的证据收集.下面,我们将从这两个方面对过程管理和证据收集进行综述.

随着模型驱动开发技术的广泛应用和 UML 在嵌入式系统中应用的日趋增长,将安全信息集成到软件和系统开发过程中的方法和技术近年来得到了更多的关注.文献[122,123]对安全关键系统软件安全认证的现状给出了综述,并且针对航空适航认证给出了详细的总结.文献[124]通过对 DO-178B 中的目标进行分析,建立了支持安全认证的概念原模型,目的是将基于 UML 的软件设计与 DO-178B 背景下的安全理由结合起来,从而在开发与认证的过程中促进更好的交流和理解.文献[125]提出了一种基于 IEC 61508 标准的概念模型,即软件开发过程中的安全证据模型.软件提供商可以根据该元模型在开发的过程中进行证据收集,便于后期进行安全认证.另外,文献[126–128]分别针对不同的应用领域结合 UML 来对系统的安全性属性进行相应的建模,便于对模型的安全属性进行验证.文献[129]扩展 UML 类图的元模型以支持非功能属性的建模,并且对所建立的模型进行可满足性与一致性验证.此外,文献[130]对系统静态结构的软件构件安全依赖的合法性采用表格形式进行说明;文献[131]基于 IEC 61508 标准对构件之间的安全等级依赖的问题进行了部分描述.

软件安全需求验证过程的证据收集可以概括为验证的验证(verification of the verification activities),它对本文中所描述的软件安全需求验证活动给予评价.目前,在学术界和工业界都已经存在相关的研究工作.文献[132]

提出了一种安全性测试方法.该方法使用故障树模型(FTA),根据标识风险的严重度及引起基本故障事件的数目实现了测试用例的生成、选择及优先级的确定.其次,也存在一些采用模型验证的方法为机载软件的安全认证提供证据的相关工作.文献[133]通过将模型检测工具集成到开发环境当中,为满足 DO-178B 的认证目标提供证据;法国 Esterel 公司开发的 SCADE 套件提供了支持部分适航认证标准的认证方法和技术,在欧洲 EC135 以及 EC155 系列直升飞机的自动飞行系统、空中客车 Airbus A340 和 A380 的控制系统以及波音 Boeing 787 的着陆导航系统等安全关键系统中得到了部分应用[134,135].

6 总结与展望

作为当前的研究热点,机载软件的安全性问题在近几年得到了广泛的研究.本文对机载软件安全性分析标准、方法及工具进行了综述,以机载软件的适航标准为基础,从机载软件安全性需求的获取与描述、面向标准的软件开发过程、软件安全需求验证这 3 个方面对当前学术界的研究工作以及工业界的相关应用进行了综述,并且针对面向安全认证的过程管理及其证据收集进行了总结.

虽然机载软件安全性的研究已经取得了一定的成果,但是针对适航认证标准的要求进行机载软件的安全性分析,仍有很多问题需要解决,体现在以下几个方面:

(1) 当前的适航认证标准是以目标为导向的,为当前的符合适航标准的机载软件的安全性评估带来了一定的困难:一方面,对于开发与验证过程,DO-178B 标准依然执行目标导向原则,规定了开发与验证进程的目标,但并没有指定要使用什么样开发和验证技术方法.因此,对于实际中的软件开发和安全评估人员而言可操作性不强;另一方面,DO-178B 标准中所给出的目标比较抽象,对于初次接触者而言,难以明确开发过程中 DO-178B 中所要求达到的目标.

(2) 在软件安全需求的获取与描述方面存在以下问题:机载软件的故障危害分析方法目前多关注系统层面,从机载系统中得到针对机载软件的安全需求以及对机载软件的安全性进行分析的方法目前尚存在不足.目前,故障危害分析工作集中在系统层面,而如何从系统的故障危害分析过程中得到软件相关的故障危害危以及得到对应的软件需求方面的工作较少.在系统级安全性分析的相关标准(SAE-ARP-4754/4761)中,将系统的安全性分析分为概念设计、功能设计、详细设计、设计的确认与验证这 4 个阶段,并对应提供了主要采用传统的功能危险性分析(functional hazard assessment,简称 FHA)、故障模式及其影响分析(failure modes and effects analysis,简称 FMEA)、故障树分析(failure tree analysis,简称 FTA)、共因分析(common cause analysis,简称 CCA)这 4 类有效的安全需求分析方法.但在机载嵌入式软件中,错误隐蔽性较高,且考虑到软件自身所特有的生命周期,无法简单地套用系统级故障危害分析的流程与方法.

(3) 在面向标准的软件开发过程中,主要存在以下问题:对于针对适航认证标准的机载软件适航安全等级分配,现有的工作尚未形成一整套的适航安全等级分配以及保证其分配与适航标准要求之间一致的机制.模型驱动方法依然处于发展中,平台无关模型 PIM(platform independent model)和平台相关模型 PSM(platform specific model)的定义尚未标准化,PIM 转换成 PSM 过程只能半自动化,使用模型驱动工具来辅助手工转换,这使得模型驱动开发在航空领域中的应用存在一定的局限性.由于最新的适航标准对模型驱动在实际机载软件开发中的应用进行了规范,因此整个的软件开发的过程必须遵从模型驱动开发来进行.

(4) 软件安全需求验证方面主要存在以下几个问题:目前,在机载软件制造与开发过程中,主要采用代码审查、仿真和测试相结合的方法来进行确认和验证.这些方法尽管操作门槛低而且能够有效地发现显性错误,但无法对软件全路径覆盖,也难以发现如并发、递归等引起的深层次错误.

(5) 当前,机载软件开发过程中尚未形成完整的安全认证证据收集方法,使得在项目开发之后,为了通过相应的适航认证标准的认证,需要补充大量的软件安全证据.如前所述,DO-178B 是目标导向的,规定了机载软件开发以及验证过程中所必须满足的一些目标,但是并未提供详细的开发和验证方法说明.

针对以上问题,在理论和方法研究方面可以做以下探索:

- 对于问题(2),研究从机载系统中得到针对机载软件的安全需求以及对机载软件的安全性进行分析的

方法;研究从系统的故障危害分析过程中得到软件相关的故障危害危以及得到对应的软件需求.

- 对于问题(3),研究针对具体软件分析其适航安全等级划分与适航标准要求之间的一致性的理论和方法.另外,研究开发过程的自动化也是研究重点,开发过程是不可能省略的,但可以借助于开发工具来提高开发过程的自动化程度,就是使用计算机帮助部分或者完全地实现某些开发过程.然而,要实现开发过程自动化的前提条件是要让计算机理解软件开发的需求,即,要用数字化、形式化的方式来描述软件需求.例如,为实现软件编码过程的自动化,自动地从低级需求得到源代码,必须要用形式化的方式来描述低级需求而不能用自然语言.形式化描述除了能够帮助实现开发过程的自动化以外,还能避免自然语言的二义性,检查各阶段软件描述的一致性和完备性.
- 对于问题(4),研究验证过程的自动化.对于验证过程,同样可以采用验证工具来提高自动化程度.要实现验证过程的自动化,尤其是软件需求过程和软件设计过程的验证的自动化,也要求使用形式化的语言来描述高级需求和低级需求.
- 针对问题(5),研究建立可追踪性元模型等,然后针对特定的安全需求,收集在软件的开发以及验证过程中所有的安全证据,说明当前的机载系统满足该安全需求.

在技术保障方面可以做如下探索:

- 对于问题(1),需要提供更加详细的适航认证标准指南,针对软件开发以及验证各阶段的目标,分析相应技术及方法,为将来在实际中进一步实施相应的开发和验证标准奠定基础.
- 对于问题(2),需要针对机载软件的特征采用合适的故障危害分析方法.
- 对于问题(3),开发有效的模型驱动开发工具,以提高开发过程的自动化.
- 对于问题(4),开发不同安全性需求验证工具,达到验证过程省略的目标.根据 DO-178B 中的说明,当使用某开发工具来实现某开发过程时,该开发过程的验证工作可以省略,其前提条件是所用开发工具本身通过了同样级别或更高级别的质量认证.这样,开发过程的验证转化成了开发工具的验证,同时把开发结果的认证转化为开发工具的认证.由于开发工具可以复用,所以对开发工具的严格验证和质量认证就替代了很多个项目的验证和质量认证.
- 针对问题(5),给出在实际的软件项目中针对适航认证标准的认证目标,对机载软件的安全性分析过程中的证据进行收集,作为机载软件安全认证的证据提交给适航认证方,以便通过机载软件的适航认证.针对适航认证标准,提供一套完整的有效软件安全证据收集流程.

References:

[1] Athalye P, Maksimovic D, Erickson R. High-Performance front-end converter for avionics applications. IEEE Trans. on Aerospace and Electronic Systems, 2003,39(2):462−470. [doi: 10.1109/TAES.2003.1207258]

[2] Learmount D. Never again. Flight Int'l, 2012,182(5367):32−35.

[3] Afshar A, Majid Hajyhosseinloo MD, Ali Eftekhari MD. A report of the injuries sustained in Iran Air Flight 277 that Crashed near Urmia, Iran. Archives of Iranian Medicine, 2012,15(5):317−319. [doi: 012155/AIM.0014]

[4] Yin YF, Liu B. Research on formal verification technique for aircraft safety-critical software. Journal of Computers, 2010,5(8): 1152−1159. [doi:10.4304/jcp.5.8.1152-1159]

[5] DO-178B/ED-12B. Software considerations in airborne systems and equipment certification. RTCA/EUROCAE, 1992.

[6] RTCA. DO-178C software considerations in airborne systems and equipment certification. Washington: Radio Technical Commission for Aeronautics, Inc., 2008.

[7] McDermid JA, Pumfrey DJ. A development of hazard analysis to aid software design. In: Proc. of the COMPASS'94. IEEE Computer Society, 1994. 17−25. [doi: 10.1109/CMPASS.1994.318470]

[8] Firesmith DG. Engineering safety-related requirements for software intensive systems. In: Proc. of the Int'l Conf. on Software Engineering (ICSE 2006). New York: ACM Press, 2006. 1047−1048. [doi: 10.1145/1134285.1134498]

[9] Ayoub A, Kim B, Lee I, Sokolsky O. A safety case pattern for model based development approach. In: Proc. of the NASA Formal Method. LNCS 7226, Berlin, Heidelberg: Springer-Verlag, 2012. 141−146. [doi: 10.1007/978-3-642-28891-3_14]

[10] Jacklin S, Lowry M R, Schumann JM, Gupta P. Verification, validation, and certification challenges for adaptive fights-critical control system software. In: Proc. of the American Institute of Aeronautics and Astronautics (AIAA) Guidance, Navigation, and Control Conf. and Exhibit. 2004. 16−19.

[11] Hawkins RD, Kelly TP. Software safety assurance—What is sufficient? In: Proc. of the 4th IET Int'l Conf. on System Safety. 2009. 1−6. [doi: 10.1049/cp.2009.1542]

[12] MIL-STD-882D. Standard practice for system safety program requirements. Military: Department of Defense, 1996.

[13] NASA-STD-8710.13. Software safety. Washington, 2004.

[14] USAF. MIL-STD-1574A. System safety program for space and missile system. Arlington: Department of Defence, 1979.

[15] Leveson NG. Software safety: Why, what, and how. Computing Survey, 1986,18(2):125−163. [doi: 10.1145/7474.7528]

[16] National Aeronautics and Space Administration. NASA-STD-8719.13A-2003 software NASA technical standard. Washington: National Aeronautics and Space Administration, 2003.

[17] GJB900-90. General program for system safety. 1990 (in Chinese). http://www.gjb.com.cn/standardshow.php?id=682411

[18] GJB/Z 142-2004. Guide for military software safety analysis. 2004 (in Chinese). http://www.gjb.com.cn/standardshow.php?id=682743

[19] SAE-ARP-4754. Certification considerations for highly-integrated or complex aircraft systems. SAE Int'l, 1996. http://standards.sae.org/arp4754/

[20] AC 20-115B. Document RTCA/DO-178B. FAA, 1992. https://www.faa.gov/regulations_policies/advisory_circulars/index.cfm/go/document.information/documentID/22122

[21] AC 20-171. Alternatives to RTCAlDO-178B for software in airborne systems and equipment. FAA, 1993. http://www.faa.gov/regulations_policies/advisory_circulars/index.cfm/go/document.information/documentID/698460

[22] AC 20-148. Reusable software components. FAA, 2004. http://www.faa.gov/regulations_policies/advisory_circulars/index.cfm/go/document.information/documentID/22207

[23] Lutz R. Software engineering for safety: A roadmap. In: Proc. of the Conf. on Future of Software Engineering. New York: ACM Press, 2000. 213−226. [doi: 10.1145/336512.336556]

[24] Hauge HJ. A survey of software safety. Technical Report, Department of Computer and Information Science, Norwegian University of Science and Technology, 2001.

[25] McDermid JA, Pumfrey DJ. Software safety: Why is there no consensus? In: Proc. of the ISSC 2001. Huntsville: System Safety Society, 2001.

[26] Medikonda BS, Panchumarthy SR. A framework for software safety in safety-critical systems. ACM SIGSOFT Software Engineering Notes, 2009,34(2):1−9. [doi: 10.1145/1507195.1507207]

[27] McDermid JA. Software safety: Where's the evidence? In: Proc. of the 6th Australian Workshop on Safety Critical Systems and Software, Vol.3. Australian Computer Society, Inc., 2001. 1−6.

[28] Fan XG, Chu WK, Zhang FM. Surveys of software safety. Computer Science, 2011,38(5):8−13 (in Chinese with English abstract). [doi: 10.3969/j.issn.1002-137X.2011.05.002]

[29] Yang SP, Sang N, Xiong GZ. Research on safety testing and evaluation technology of safety critical software. Chinese Journal of Computers, 2004,27(4):442−450 (in Chinese with English abstract). [doi: 10.3321/j.issn:0254-4164.2004.04.002]

[30] Fang BX, Lu TB, Li C. Survey of software assurance. Journal on Communications, 2009,30(2):106−117 (in Chinese with English abstract).

[31] Wu WH, Kelly T. Safety tactics for software architecture design. In: Proc. of the 28th Annual Int'l Computer Software and Applications Conf. 2004. [doi: 10.1109/CMPSAC.2004.1342860]

[32] Zhao TD. Safety design analysis and verification. Beijing: National Defense Industry Press, 2011 (in Chinese).

[33] Xu XJ, Bao XH, Lu MY, Chang W. A study and application on airborne software safety requirements elicitation. In: Proc. of the 2011 9th Int'l Conf. on Reliability, Maintainability and Safety (ICRMS). 2011. 710−716. [doi: 10.1109/ICRMS.2011.5979357]

[34] Walia GS, Carver JC. A systematic literature review to identify and classify software requirements errors. Information and Software Technology, 2009,51(7):1087−1109. [doi:10.1016/j.infsof.2009.01.004]

[35] Rausand M, Høyland A. System Reliability Theory: Models, Statistical Methods and Applications. 2nd ed., Wiley, 2004.

[36] IEC 60812: Analysis techniques for system reliability. In: Proc. of the Failure Mode and Effect Analysis (FMEA). Int'l Electrotechnical Commission, 1991.

[37] IEC 61822: Hazard and operability studies (HAZOP studies)—Application guide. Int'l Electrotechnical Commission, 2001.

[38] Rausand M, Hoyland A. System Reliability Theory: Models, Statistical Methods, and Applications. 2nd ed., New York: Wiley InterScience, 2003.

[39] Andrews JD, Dunnett SJ. Event-Tree analysis using binary decision diagrams. IEEE Trans. on Reliability, 2000,49(2):230−238. [doi: 10.1109/24.877343]

[40] IEC 61078: Analysis techniques for dependability—Reliability block diagram method. Int'l Electrotechnical Commission, 1991.

[41] Ortmeier F, Schellhorn G. Formal fault tree analysis-practical experiences. Electronic Notes in Theoretical Computer Science, 2007,185:139−151. [doi: 10.1016/j.entcs.2007.05.034]

[42] Cha S, Yoo J. A safety-focused verification using software fault trees. Future Generation Computer Systems, 2012,28(8): 1272−1282. [doi: 10.1016/j.future.2011.02.004]

[43] Čepin M, Mavko B. A dynamic fault tree. Reliability Engineering & System Safety, 2002,75(1):83−91. [doi: 10.1016/S0951-8320(01)00121-1]

[44] Hansen KM, Ravn AP, Stavridou V. From safety analysis to safety requirements. IEEE Trans. on Software Engineering, 1998, 24(7):573−584. [doi: 10.1109/32.708570]

[45] Schellhorn G, Thums A, Reif W. Formal fault tree semantics. In: Proc. of the Integrated Design and Process Technology (IDPT 2002). 2002. 1−8.

[46] Kaiser B, Zocher A. Bdd complexity reduction by component fault trees. In: Proc. of the European Safety and Reliability Conf. (ESREL 2005). Rotterdam: Balkema Publishers, 2005. 1011−1019.

[47] Kaiser B, Gramlich C, Forster M. State/event fault trees—A safety analysis model for software-controlled systems. Reliability Engineering & System Safety, 2007,92(11):1521−37. [doi: 10.1016/j.ress.2006.10.010]

[48] Walker M, Papadopoulos Y. Qualitative temporal analysis: Towards a full implementation of the fault tree handbook. Control Engineering Practice, 2009,17(10):1115−25. [doi: 10.1016/j.conengprac.2008.10.003]

[49] Palshikar GK. Temporal fault trees. Information and Software Technology, 2002,44(3):137−150. [doi: 10.1016/S0950-5849(01)00223-3]

[50] Walker M, Bottaci L, Papadopoulos Y. Compositional temporal fault tree analysis. In: Proc. of the 26th Int'l Conf. on Computer Safety, Reliability, and Security (SAFECOMP 2007). Berlin, Heidelberg: Springer-Verlag, 2007. 106−119. [doi: 10.1007/978-3-540-75101-4_12]

[51] Domis D, Hofig K, Trapp M. A consistency check algorithm for component-based refinements of fault trees. In: Proc. of the ISSRE 2010. 2010. 171−180. [doi: 10.1109/ISSRE.2010.23]

[52] Galileo dynamic fault tree analysis tool. 1999. http://www.cs.virginia.edu/~ftree/

[53] Relex software. 1985. http://www.relex.com

[54] Fault tree+. 1986. http://www.isograph-software.com/2011/software/reliability-workbench/fault-tree-analysis/

[55] Allenby K, Kelly T. Deriving safety requirements using scenarios. In: Proc. of the 5th IEEE Int'l Symp. on Requirements Engineering. 2001. 228−235. [doi: 10.1109/ISRE.2001.948563]

[56] Troubitsyna E. Elicitation and specification of safety requirements. In: Proc. of the 3rd Int'l Conf. on Systems. IEEE Computer Society, 2008. 202−207. [doi: 10.1109/ICONS.2008.56]

[57] Ma L, Huang ZQ, Xu BF, Chen Z. A fault tree analysis method supporting model checking. Computer & Digital Engineering, 2013,41(2):257−260 (in Chinese with English abstract).

[58] Xiang J, Ogata K, Kong W, Futatsugi K. From fault tree analysis to formal system specification and verification with ots/cafeobj. Information and Media Technologies, 2007,2(2):448−460.

[59] Nejati S, Sabetzadeh M, Falessi D, Briand L, Coq T. A SysML-based approach to traceability management and design slicing in support of safety certification: Framework, tool support, and case studies. Information and Software Technology, 2012,54(6): 569−590. [doi: 10.1016/j.infsof.2012.01.005]

[60] Lamsweerde A. Formal specification: A roadmap. In: Proc. of the Conf. on the Future of Software Engineering. New York: ACM Press, 2000. 147−159.

[61] Koh K Y, Seong PH. SMV model-based safety analysis of software requirements. Reliability Engineering & System Safety, 2009, 94(2):320−331. [doi: 10.1016/j.ress.2008.03.025]

[62] Cha S, Yoo J. A safety-focused verification using software fault trees. Future Generation Computer Systems, 2012,28(8): 1272−1282. [doi: 10.1016/j.future.2011.02.004]

[63] Yoo J, Kim T, Cha S, Lee JS, Son HS. A formal software requirements specification method for digital nuclear plant protection systems. Journal of Systems and Software, 2005,74(1):73−83. [doi: 10.1016/j.jss.2003.10.018]

[64] Hu X, Yang CH, Huang MS. Construction and estimation of requirement knowledge ontology for avionics electronics system software. Computer Engineering, 2013,39(3):56−62 (in Chinese with English abstract). [doi: 10.3969/j.issn.1000-3428.2013.03.012]

[65] Li Z, Liu B, Miao H, Yin YF. Modeling and verification of software safety requirement based on ontology. Journal of Beijing University of Aeronautics and Astronautics, 2012,38(11):1445−1449 (in Chinese with English abstract).

[66] Liu C, Wang Y, Jin Z. Eliciting dependability requirements: A knowledge-based approach. Acta Electronica Sinica, 2010,38(2A): 188−193 (in Chinese with English abstract).

[67] Functional safety of electrical/electronic/programmable electronic safety related systems. IEC 61508, Int'l Electrotechnical Commission, 1998. http://cds.cern.ch/record/441660

[68] Simon C, Sallak M, Aubry J. SIL allocation of SIS by aggregation of experts' opinions. In: Proc. of the Safety and Reliability Conf. (ESREL 2007). 2007. 753−761.

[69] Fujiwara T, Kimura M, Satoh Y, Yamada S. A method of calculating safety integrity level for IEC 61508 conformity software. In: Proc. of the 2011 IEEE 17th Pacific Rim Int'l Symp. on Dependable Computing (PRDC). IEEE Computer Society Press, 2011. 296−301.

[70] Fujiwara T, Estevez JM, Satoh Y, Yamada S. A calculation method for software safety integrity level. In: Proc. of the 1st Workshop on Critical Automotive Applications: Robustness & Safety (CARS 2010). New York: ACM Press, 2010. 31−34.

[71] Papadopoulos Y, Walker M, Reiser MO, Weber M, Chen D, Törngren M, Servat D, Abele A, Stappert F, Lonn H, Berntsson L, Johansson R, Tagliabo F, Torchiaro S, Sandberg A. Automatic allocation of safety integrity levels. In: Proc. of the 1st Workshop on Critical Automotive Applications: Robustness & Safety (CARS 2010). New York: ACM Press, 2010. 7−10.

[72] Mader R, Armengaud E, Leitner A, Steger C. Automatic and optimal allocation of safety integrity levels. In: Proc. of the Reliability and Maintainability Symp. (RAMS). IEEE Computer Society Press, 2012. 1−6.

[73] Jiang YB, Shao WZ, Zhang L, Ma ZY. On the formal definition and analysis of stereotype in UML. Acta Electronica Sinica, 2003,31(B12):2101−2105 (in Chinese with English abstract). [doi: 10.3321/j.issn:0372-2112.2003.z1.036]

[74] Jiang YB, Shao WZ, Zhang L, Ma ZY, Meng XW, Ma HH. On the classification of UML's meta model extension mechanism. 2004. 54−68. http://link.springer.com/chapter/10.1007/978-3-540-30187-5_5.

[75] Wang XB, Wang HM, Wu QY, Shi DX. A weaving framework for model transformation. Ruan Jian Xue Bao/Journal of Software, 2006,17(6):1423−1435 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/17/1423.htm [doi: 10.1360/jos171423]

[76] Cui M, Li XD, Zheng GL. Formal analysis on UML real-time activity diagram. Chinese Journal of Computers, 2004,27(3): 309−346 (in Chinese with English abstract). [doi: 10.3321/j.issn:0254-4164.2004.03.008]

[77] Zhu Y, Huang ZQ, Cao ZN, Zhou H, Liu YP. Method for generating software architecture models from formal specifications. Ruan Jian Xue Bao/Journal of Software, 2010,21(11):2738−2751 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/3701.htm [doi: 10.3724/SP.J.1001.2010.03701]

[78] Wang Y, Zhou X, Dong Y, Li C. A MDA-based approach for general embedded software simulation platform. In: Proc. of the 8th Int'l Conf. on Embedded Computing, ScalCom (EmbeddedCom 2009). IEEE Computer Society, 2009. 20−25.

[79] Malek S, Mikic-Rakic M, Medvidovic N. A style-aware architectural middleware for resource-constrained, distributed systems. IEEE Trans. on Software Engineering, 2005,31(3):256−272. [doi: 10.1109/TSE.2005.29]

[80] Insaurralde CC, Seminario MA, Jimenez JF, Giron-Sierra JM. Model-Driven system development for distributed fuel management in avionics. Journal of Aerospace Computing, Information, and Communication, 2013,10(2):71−86. [doi: 10.2514/1.53714]

[81] Mohagheghi P, Dehlen V. Where is the proof?—A review of experiences from applying MDE in industry. In: Proc. of the ECMDA-FA 2008. LNCS 5095, 2008. 432−443. [doi: 10.1007/978-3-540-69100-6_31]

[82] The TOPCASED Website. http://www.topcased.org/

[83] The SPICES ITEA Project Website. http://www.spices-itea.org/public/news.php

[84] The ASSERT Project Website. http://www.esa.int/TEC/Software_engineering_and_standardisation/TECJQ9UXBQE_0.html

[85] Cofer D. Model checking: Cleared for take off. In: Proc. of the SPIN 2010. LNCS 6349, 2010. 76−87. [doi: 10.1007/978-3-642-16164-3_6]

[86] Bochot T, Virelizier P, Waeselynck H, Wiels V. Model checking flight control systems: The airbus experience. In: Proc. of the 31st Int'l Conf. on Software Engineering (ICSE 2009). IEEE, 2009. 18−27. [doi: 10.1109/ICSE-COMPANION.2009.5070960]

[87] Whalen M. Why we model using MBD effectively in critical domains. 2013. http://2013.icse-conferences.org/documents/publicity/MiSE-WS-Whalen-slides.pdf

[88] Clarke EM, Grumberg O, Peled DA. Model Checking. The MIT Press, 2000.

[89] Delgado N, Gates AQ, Roach S. A taxonomy and catalog of runtime software-fault monitoring tools. IEEE Trans. on Software Engineering, 2004,30(12):859−872. [doi: 10.1109/TSE.2004.91]

[90] Holzmann GJ. The SPIN Model Checker: Primer and Reference Manual. Addison-Wesley Professional, 2003.

[91] Lin HM, Zhang WH. Model checking: Theories, techniques and applications. Acta Electronica Sinica, 2002,30(12A):1907−1912 (in Chinese with English abstract).

[92] Liu J, Lü JD, Quan Z, Zhan NJ, Zhao HJ, Zhou CC, Zou L. A calculus for hybrid CSP. In: Proc. of the APLAS 2010. Berlin, Heidelberg: Springer-Verlag, 2010. 1−15. [doi: 10.1007/978-3-642-17164-2_1]

[93] Shao J, Deng F, Wang QX. A model-based software system monitoring approach. Journal of Computer Research and Development, 2010,47(7):1175−1183 (in Chinese with English abstract).

[94] Zhao CZ, Dong W, Sui P, Qing ZC. Construction techniques of anticipatory monitors for parameterized LTL. Ruan Jian Xue Bao/Journal of Software, 2010,21(2):318−333 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/3738.htm [doi: 10.3724/SP.J.1001.2010.03738]

[95] Hu J, Yu XF, Zhang Y, Li XD, Zheng GL. Scenario-Based consistency verification of component-based real-time system designs. Ruan Jian Xue Bao/Journal of Software, 2006,17(1):48−58 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/17/48.htm [doi: 10.1360/josl70048]

[96] Du JW, Xu ZW, Jiang F. Research on verification of behavior requirement patterns based on action sequences. Journal on Communication, 2011,32(1):94−105 (in Chinese with English abstract).

[97] Shi JQ, He JF, Zhu HB, Fang HX, Huang YH, Zhang XX. ORIENTAIS: Formal verified OSEK/VDX real-time operating system. In: Proc. of the ICECCS 2012. IEEE, 2012. 293−301.

[98] Li Z, Liu B, Lu MY, Yin YF. Modeling and verification of deicing software safety requirement based on expanded Petri net. Journal of Beijing University of Aeronautics and Astronautics, 2012,38(1):64−68 (in Chinese with English abstract).

[99] Xu BF, Huang ZQ, Hu J. Model-Driven safety dependence verification for component-based airborne software supporting airworthiness certification. Acta Aeronautica et Astronautica Sinica, 2012,33(5):796−808 (in Chinese with English abstract).

[100] Chen Z, Gu Y, Huang Z, Zheng J, Liu C, Liu ZY. Model checking aircraft controller software: A case study. Software: Practice and Experience, 2013. [doi: 10.1002/spe.2242]

[101] Esteve MA, Katoen JP, Nguyen VY, Postma B, Yushtein Y. Formal correctness, safety, dependability, and performance analysis of a satellite. In: Proc. of the 34th Int'l Conf. on Software Engineering (ICSE 2012). IEEE, 2012. 1022−1031.

[102] Moser M, Ibens O, Ietz R, Steinbach J, Goller C, Schumann J, Mayr K. SETHEO and ESETHEO-the-CADE-13 systems. Journal of Automated Reasoning, 1997,18(2):237−246. [doi: 10.1023/A:1005808119103]

[103] Hustadt U, Motik B, Sattle U. Reasoning in description logics with a concrete domain in the framework of resolution. In: Proc. of the 16th European Conf. on Artificial Intelligence (ECAI 2004). Amsterdam: IOS Press, 2004. 353−357.

[104] Aitken JS, Reichgelt H, Shadbolt N. Resolution theorem proving in reified modal logics. Journal of Automated Reasoning, 1994, 12(1):103−129. [doi: 10.1007/BF00881845]

[105] Fitting M. Types and Tableau. Berlin: Springer-Verlag, 2000.

[106] Dag P. Natural Deduction. New York: Dover Publications, 2006.

[107] Sun JG, Li Y, Zhu XJ, Lü S. Anovel theorem proving algorithm based on extension rule. Journal of Computer Research and Development, 2009,14(1):9−14.

[108] McMillan KL. Symbolic Model Checking. Kluwer Academic Publishers, 1993.

[109] Holzmann GJ. The model checker spin. IEEE Trans. on Software Engineering, 1997,23(5):279−295. [doi: 10.1109/32.588521]

[110] Larsen KG, Petterson P, Wang Y. UPPAAL in a nutshell. Journal of Software Tools for Technology Transfer, 1997,1(1-2): 134−152.

[111] Clarke E, Kroening D, Lerda F. A tool for checking ANSI-C programs. In: Proc. of the TACAS 2004. Springer-Verlag, 2004. 168−176.

[112] Henzinger TA, Jhala R, Majumdar R, Sutre G. Lazy abstraction. In: Proc. of the 29th ACM Symp. on Principles of Programming Languages. 2002. 58−70. [doi: 10.1007/s100090050010]

[113] Ball T, Majumdar R, Millstein T, Rajamani S. Automatic predicate abstraction of C programs. In: Proc. of the ACM SIGPLAN 2001 Conf. on Programming Language Design and Implementation (PLDI 2001). 2001. 203−213. [doi: 10.1145/381694.378846]

[114] Allan C, Avgustinov P, Christensen AS, Hendren L, Kuzins S, Lhotak O, de Moor O, Sereni D, Sittampalam G, Tibble J. Adding trace matching with free variables to AspectJ. In: Proc. of the OOPSLA 2005. New York: ACM Press, 2005. 345−364. [doi: 10.1145/1103845.1094839]

[115] Chen F, Jin DY, Meredith P, Rosu G. Monitoring oriented programming—A project overview. In: Proc. of the ICICIS 2009. New York: ACM Press, 2009. 72−77.

[116] Colombo C, Pace GJ, Schneider G. LARVA-Safer monitoring of real-time Java program. In: Proc. of the 7th IEEE Int'l Conf. on Software Engineering and Formal Methods. IEEE Computer Society, 2009. 33−37. [doi: 10.1109/SEFM.2009.13]

[117] Owre S, Rushby JM, Shankar N. PVS: A prototype verification system. In: Proc. of the CADE'92. LNAI 607, 1992. 748−752. [doi: 10.1007/3-540-55602-8_217]

[118] Dodd I, Habli I. Safety certification of airborne software: An empirical study. Reliability Engineering & System Safety, 2011, 98(1):7−23. [doi: 10.1016/j.ress.2011.09.007]

[119] Bala I, Sharma SK, Sunand K. Exploring raw safety aspects in aviation industry. Computer Engineering and Intelligent Systems, 2013,4(1):80−97.

[120] Wassyng A, Maibaum T, Lawford M, Bherer H. Software certification: Is there a case against safety cases? In: Calinescu R, Jackson E, eds. Proc. of the 16th Monterey Workshop: Foundations of Computer Software—Modeling, Development, and Verification of Adaptive Systems. LNCS 6662, 2011. 206−227. [doi: 10.1007/978-3-642-21292-5_12]

[121] Kong J, Yan HH. Comparisons and analyses between RTCA DO-178B and GJB5000A, and integration of software process control. In: Proc. of the ICACTE 2010. IEEE Computer Society, 2010. V6367−V6372. [doi: 10.1109/ICACTE.2010.5579827]

[122] Kornecki A, Zalewski J. Certification of software for real-time safety-critical systems: State of the art. Innovations in Systems and Software Engineering, 2009,5(2):149−161. [doi: 10.1007/s11334-009-0088-1]

[123] Huhn M, Hungar H. UML for software safety and certification. In: Proc. of the MBEERTS. LNCS 6100, 2010. 201−237. [doi: 10.1007/978-3-642-16277-0_8]

[124] Zoughbi G, Briand LC, Labiche Y. A UML profile for developing airworthiness-compliant (RTCA DO-178B) safety-critical software. In: Proc. of the ACM/IEEE Int'l Conf. on Model Driven Engineering Languages and Systems. 2007. 574−588. [doi: 10.1007/978-3-540-75209-7_39]

[125] Panesar-Walawege RK, Sabetzadeh M, Briand L. Characterizing the chain of evidence for software safety cases: A conceptual model based on the IEC 61508 standard. IEEE, 2010. 335−344. [doi: 10.1109/ICST.2010.12]

[126] Bernardi S, Merseguer J, Petriu DC. Adding dependability analysis capabilities to the MARTE profile. In: Czarnecki K, Ober I, Bruel JM, Uhl A, Ïolter M, eds. Proc. of the MODELS 2008. LNCS 5301, Heidelberg: Springer-Verlag, 2008. 736−750. [doi: 10.1007/978-3-540-87875-9_51]

[127] Jürjens J. Developing safety-critical systems with UML. In: UML 2003—The Unified Modeling Language. Berlin, Heidelberg: Springer-Verlag, 2003. 360−372. [doi: 10.1007/978-3-540-45221-8_31]

[128] Object Management Group. UML profile for modeling and analysis of real-time and embedded systems (MARTE). 2008. http://www.omg.org/spec/MARTE/

[129] Zhang Y, Mei H. Non-Functional attributes oriented description and verification in UML class diagrams. Ruan Jian Xue Bao/Journal of Software, 2010,20(6):1457−1469 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/3455.htm [doi: 10.3724/SP.J.1001.2009.03455]

[130] Zoughbi G, Briand L, Labiche Y. Modeling safety and airworthiness (RTCA DO-178B) information: Conceptual model and UML profile. Software and Systems Modeling, 2010,10(3):1−31.

[131] Hause MC, Thom F. An integrated safety strategy to model driven development with SysML. In: Proc. of the 2nd Institution of Engineering and Technology Int'l Conf. on System Safety. London: IEEE, 2007. 124−129.

[132] Kloos J, Hussain T, Eschbach R. Risk-Based testing of safety-critical embedded systems driven by fault tree analysis. In: Proc. of the IEEE 4th Int'l Conf. on Software Testing, Verification and Validation (ICST 2011). Berlin: IEEE Computer Society, 2011. 26−33.

[133] Cofer D. Model checking: Cleared for take off. In: Proc. of the SPIN 2010. LNCS 6349, Berlin, Heidelberg: Springer-Verlag, 2010. 76−87. [doi: 10.1007/978-3-642-16164-3_6]

[134] The Scade Website. http://www.esterel-technologies.com/products/scade-suite/

[135] Bochot T, Virelizier P, Waeselynck H, Wiels V. Model checking flight control systems: The airbus experience. In: Proc. of the 31st Int'l Conf. on Software Engineering (ICSE 2009). IEEE, 2009. 18−27. [doi: 10.1109/ICSE-COMPANI ON.2009.5070960]

附中文参考文献:

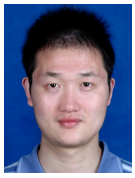
[17] GJB900-90.系统安全性通用大纲.1990. http://www.gjb.com.cn/standardshow.php?id=682411

[18] GJB/Z 142-2004.军用软件安全性分析指南.2004. http://www.gjb.com.cn/standardshow.php?id=682743

[28] 樊晓光,褚文奎,张凤鸣.软件安全性研究综述.计算机科学,2011,38(5):8−13. [doi: 10.3969/j.issn.1002-137X.2011.05.002]
[29] 杨仕平,桑楠,熊光泽.安全关键软件的防危性测评技术研究.计算机学报,2004,27(4):442−450. [doi: 10.3321/j.issn:0254-4164.2004.04.002]
[30] 方滨兴,陆天波,李超.软件确保研究进展.通信学报,2009,30(2):106−117.
[32] 赵廷弟.安全性设计分析与验证.北京:国防工业出版社,2011.
[57] 马琳,黄志球,徐丙凤,陈哲.支持模型检测的故障树生成方法研究.计算机与数字工程,2013,41(2):257−260.
[64] 胡璇,杨春晖,黄茂生.航电系统软件需求知识本体构建及评价.计算机工程,2013,39(3):56−62. [doi: 10.3969/j.issn.1000-3428.2013.03.012]
[65] 李震,刘斌,苗虹,殷永峰.基于本体的软件安全性需求建模和验证.北京航空航天大学学报,2012,38(11):1445−1449.
[66] 刘春,王越,金芝.基于知识的软件可信性需求获取.电子学报,2010,38(2A):188−193.
[73] 蒋严冰,邵维忠,张路,麻志毅.UML 中衍型的精确定义与分析.电子学报,2003,31(B12):2101−2105. [doi: 10.3321/j.issn:0372-2112.2003.z1.036]
[75] 王学斌,王怀民,吴泉源,史殿习.一种模型转换的编织框架.软件学报,2006,17(6):1423−1435. http://www.jos.org.cn/1000-9825/17/1423.htm [doi: 10.1360/jos171423]
[76] 崔萌,李宣东,郑国梁.UML 实时活动图的形式化分析.计算机学报,2004,27(3):339−346. [doi: 10.3321/j.issn:0254-4164.2004.03.008]
[77] 祝义,黄志球,曹子宁,周航,刘亚萍.一种基于形式化规约生成软件体系结构模型的方法.软件学报,2010,21(11):2738−2751. http://www.jos.org.cn/1000-9825/3701.htm [doi: 10.3724/SP.J.1001.2010.03701]
[91] 林惠民,张文辉.模型检测:理论、方法与应用.电子学报,2002,30(12A):1907−1912.
[93] 邵津,邓芳,王千祥.一种基于模型的软件系统监测方法.计算机研究与发展,2010,47(7):1175−1183.
[94] 赵常智,董威,隋平,齐治昌.面向参数化 LTL 的预测监控器构造技术.软件学报,2010,21(2):318−333. http://www.jos.org.cn/1000-9825/3738.htm [doi: 10.3724/SP.J.1001.2010.03738]
[95] 胡军,于笑丰,张岩,李宣东,郑国梁.基于场景构件式实时软件设计的一致性检验.软件学报,2006,17(1):48−58. http://www.jos.org.cn/1000-9825/17/48.htm [doi: 10 1360/josl70048]
[96] 杜军威,徐中伟,江峰.基于动作序列的行为需求模型验证的研究.通信学报,2011,32(1):94−105.
[98] 李震,刘斌,陆民燕,殷永峰.基于扩展 Petri 网的除冰软件安全需求建模和验证.北京航空航天大学学报,2012,38(1):64−68.
[99] 徐丙凤,黄志球,胡军.面向适航认证的模型驱动机载软件构件的安全性验证.航空学报,2012,33(5):796−808.
[129] 张岩,梅宏.UML 类图中面向非功能属性的描述和检验.软件学报,2010,20(6):1457−1469. http://www.jos.org.cn/1000-9825/3455.htm [doi: 10.3724/SP.J.1001.2009.03455]

黄志球(1965－),男,江苏南京人,博士,教授,博士生导师,CCF 高级会员,主要研究领域为软件工程,服务计算.
E-mail: zqhuang@nuaa.edu.cn

徐丙凤(1986－),女,博士生,CCF 学生会员,主要研究领域为软件工程,软件安全性分析与验证.
E-mail: xubingfeng@nuaa.edu.cn



阚双龙(1988－),男,博士生,主要研究领域为软件工程,形式化方法,嵌入式软件分析与验证.
E-mail: kanshuanglong@nuaa.edu.cn

胡军(1973－),男,博士,副教授,CCF 会员,主要研究领域为软件工程,嵌入式系统分析与验证,Cyber Physical Systems,形式化方法.
E-mail: hujun@nuaa.edu.cn

陈哲(1981－),男,博士,副教授,CCF 会员,主要研究领域为软件工程,软件验证,形式化方法.
E-mail: zhechen@nuaa.edu.cn