基于系统思维的软件安全性需求开发框架

褚文奎 丛 伟 樊晓光 顾文灿

(空军工程大学工程学院 西安 710038)

摘 要 糟糕的软件需求是导致安全性关键系统发生灾难性事故的最主要原因。为解决需求开发问题,建构了一个系统建模与系统分析相结合、基于系统思维的软件安全性需求开发框架。针对系统模型的特定等级特定领域,提出了集成安全性分析的需求开发方法。该方法既能最大限度地约束安全性需求缺陷,防止其向同一分析等级内的其它领域或下一分析等级传播,并尽早重新生成安全性需求,又能够不断生成证据,支持安全性论据的构建。

关键词 软件安全性,系统思维,需求工程,安全性分析,综合航电系统

中图法分类号 TP311,TP309 **文献标识码** A

System Thinking Based Development Framework for Software Safety Requirements

CHU Wen-kui CONG Wei FAN Xiao-guang GU Wen-can

(Institute of Engineering, Air Force Engineering University, Xi'an 710038, China)

Abstract Poor software requirement for safety-critical systems (SCSs) is identified as a major root cause of catastrophic accidents. A system thinking based development framework for software safety requirements was built with system modeling and system analysis. For a particular analysis domain in a particular analysis level, a development method integrated with safety analysis was presented to develop software safety requirements. With the method, safety critical errors in software requirements are neither likely to propagate through to other analysis domains in the same analysis level nor likely to the subsequent analysis level. New safety requirements will be derived as early as errors are found in the safety analysis process. Safety evidence will be generated in the process to support the building of safety arguments.

Keywords Software safety, System thinking, Requirement engineering, Safety analysis, Integrated modular avionics system

1 引言

在航空领域中,越来越多的航电软件承担着安全性关键的指挥、控制、监视功能。比如,美军四代战机 F-22、F-35 包含了数百万行软件代码[1],担负飞行控制、任务计算、通信导航等功能。这些安全性关键的航电软件一旦出现故障,将造成灾难性后果。2004 年一架 F-22 战机就曾因飞行控制软件故障而坠毁[2]。确保软件不会作为诱发因素造成安全性关键系统发生灾难性事故,这是软件安全性的研究初衷[3]。在影响软件安全性的众多因素中,文献[4]表明,软件需求缺陷导致的安全性问题已占到 60%~80%。而单元测试发现的软件缺陷和错误中需求类占到 70%[5]。F-22 项目办公室认为这一数字甚至高达 85%。这都说明糟糕的软件安全性需求是导致事故发生的主要根源。

评价软件安全性需求开发质量的三个常用指标是正确性(correctness)、完整性(completeness)和一致性(consistency)。开发的软件安全性需求如果不正确、不完备,抑或相互冲突,都会影响软件安全性。本文着重研究软件需求开发阶段的安全性需求开发方法,旨在自软件开发之初就考虑软件安全性问题,确保开发出正确的、完整的、一致的软件安全性需求,保证软件安全性。

2 相关研究

航电软件开发目前普遍遵循 DO-178B 标准[6]。它规定,航电软件的高级需求由航电系统生命周期产生的系统需求、系统结构等进行开发。在航电软件设计过程中,反复迭代求精高级软件需求,从而开发出软件体系结构以及可以直接进行源代码实现的低级需求。数十年来,DO-178B 饱受批评和指责,版本升级压力巨大,其中一个重要原因就在于它没有陈述针对软件危险失效的安全性分析活动。与英国国防部军用系统开发标准 DS 00-56(第四版)[7]相比,DO-178B 既没有致力于安全性管理,也没有安全性需求识别、安全性审计实施、安全性记录维护等方面的要求。

从需求工程视角而言,软件需求的模糊特征使其可以在不同的抽象级别上进行求精[8],并最终形成了一个指导系统设计的目标结构图。两种典型方法是自动规约中的知识采集

本文受国家自然科学基金(61172083)和总装备部国防预研基金(9140A17020307JB3201)资助。

褚文奎(1980-),男,博士,讲师,CFF 会员,主要研究方向为航电系统综合化技术、军用航电软件安全性,E-mail:chuwenkui@126.com;丛 伟(1973-),女,博士,讲师,主要研究方向为综合航电系统;樊晓光(1965-),男,博士,教授,CCF 会员,主要研究方向为综合航电系统结构;顾文灿(1965-),男,硕士,副教授,主要研究方向为综合航电系统。

(knowledge acquisition in automated specification, KAOS)和非功能性需求框架(non-functional requirement framework, NFRF)。在KAOS目标结构中,高级目标由软件系统及其外部环境的初始文档析出,然后反复求精,直到落实到相应智能体(比如操作者、硬件设备、软件)上。目标求精过程通过合并AND/OR分解模板或领域特定的求精模板进行表示。由于KAOS将所有的目标都当作了硬目标,只能以形式化方式进行规约和实现,因此它不太适合安全性等以可接受标准进行实现的软目标[9]。

相较于KAOS主要用于功能需求一类的硬目标而言,NFRF主要用于质量需求的表达和分析。它将非功能需求作为可能会被违反的软目标,并认为只有通过特定的设计技术才能予以实现。NFRF提供了能够利用图形结构求精和实现质量需求的统一模型。NFRF的缺点在于它未能提供有效的机制阐明质量需求[9]。

软件安全性研究的先驱者、美国麻省理工学院的Leveson教授主持开发了意图规约(intent specification)[10]结构,用于捕获软件系统需求。该结构分为7个等级,每一级都从人、系统、环境、验证与确认(verification and validation, V&V)角度设计、管理系统规约,说明了系统规约的内容、方法以及原因。需要指出的是,意图规约的每一级并不是对上一级信息的细化,而是从一个新视角重新审视系统。意图规约的这种层次化需求开发方法为本文分等级、分领域研究基于系统思维的软件安全性需求开发框架提供了思想借鉴。不过,本文采用的不是多视角,而是自上而下的细化求精思路。

为本文提供软件需求开发并进行安全性分析的另一个启示来自于FAA推荐的针对运载火箭的软件安全性需求开发方法[11]。该方法分为5个步骤:①定义能满足系统安全性需求的软件开发方法,制定系统安全性项目计划和软件开发计划;②识别安全性关键的系统功能并进行优先级排序,明确安全性工作的侧重点;③借助系统功能有关知识以及系统设计标准、安全标准及已有经验教训等,定义出系统顶级安全性需求,识别出与这些系统功能相关的危险,比如计算错误、数据错误、逻辑错误、接口错误、环境错误、硬件错误等;④评估并给出相应的风险减轻策略,比如故障探测、隔离、容忍、恢复等;⑤进行有效性确认和验证。

3 软件安全性需求开发的系统思维与开发框架

3.1 系统思维

软件系统之所以存在安全隐患,是因为它存在脆弱性——即系统需求缺陷、设计缺陷等的外在反映。这些缺陷如果被激活形成系统故障或失效将可能影响系统安全性。传统事故链模型认为事故是由于单个系统构件失效造成的。实际上大多数事故则是由多个系统构件之间的非正常交互引起的[12]。随着安全性关键系统的软件密集化程度越来越高,系统构件之间的交互越来越复杂,致使系统脆弱性更难以定位和预测。解决此问题的一种思路是采用系统论的有关思想,比如系统论事故模型与过程STAMP[13]将事故认为是在系统设计、开发、运行上强加了不妥当的安全性控制或约束造成的。

与STAMP模型试图利用系统论解释事故为何发生不同,本文试图借鉴系统论的观点,在需求开发阶段借助系统思维从更宽广的视角研究软件的上下文以及软件构件之间的交互,分析、确定软件承担的全部系统安全性职责,确保开发出正确的、完整的、一致的软件安全性需求。

3.2 开发框架

在软件需求分析阶段需要关注并解决的安全性相关问题包括:①系统上下文。安全性是一个系统问题,软件安全性只能在系统上下文中进行分析。由此就必须准确建立系统和环境之间的交互模型,以便确定与系统、软件等相关的风险能否被接受;②失效行为。系统必须努力维护其安全行为,即便是在部分构件失效或环境行为背离期望的情况下。

客观而言,在软件需求开发阶段能否或在多大程度上解决上述安全性问题,关键在于能否建立恰当的需求开发模型并提供有力的系统分析支持。为此,本文建构一个集成系统建模和系统分析的、基于系统思维的软件安全性需求开发框架,如图1所示。

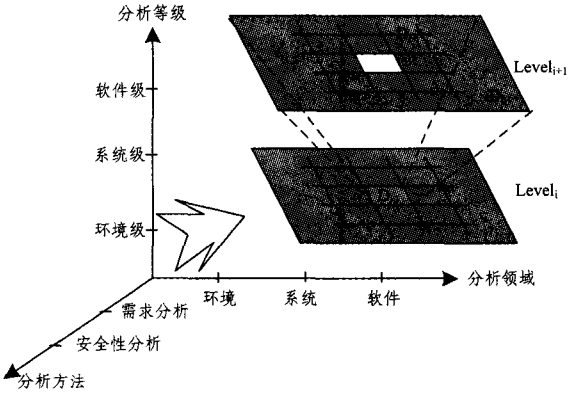

图1 基于系统思维的软件安全性需求开发框架

系统模型捕获系统运行环境、运行机制、构件交互规则等,为系统分析提供上下文;系统分析捕获系统需要提供的服务和需要满足的约束,便于求精系统模型。该框架具有3个维度,阐释如下:

• 分析等级——表征了模型的层次性。可以粗略地分为环境级、系统级、软件级。每个分析等级包含了若干个分析领域。

• 分析领域——描述了某个分析等级内特定的分析范围和目标,能够为分析等级提供领域知识。其中,环境可视为一个大系统,软件可视为一个子系统。

• 分析方法——针对系统模型某个层级某个领域进行系统分析的方法,是开发包括安全性需求在内的所有需求并求精系统模型的推动力。

其中,分析等级与分析领域实际上是对软件(包括软件构件)上下文的一种描述,而分析方法则提供了系统分析的手段。

3.3 系统建模

系统论的一个基本思想是将系统视为一个层次化的结构。系统每一级都对下一级的活动加以限制,定义相应的行为准则,从而控制其行为。而层次本身由运行在层级接口的控制过程进行表征。

假设环境和软件及其求精部分(比如构件)都可以分别视为一个系统,那么系统建模的对象包括环境、系统、软件、构件等。建模系统运行环境的目的在于刻画环境强加在系统结构及其行为上的约束,以便开发系统在环境中集成的关键需求。

建模系统的本质在于刻画系统的结构和行为，为需求开发和分析提供上下文。系统结构可以由系统的组成构件进行建模，系统的行为则是一个包含了每个构件行为以及构件之间交互行为的集合。其中交互行为可通过语法和语义两个方面进行描述。语法方面是指构件之间的接口定义，语义方面是指在接口部分观测到的行为。刻画系统行为的基本方法是使用黑盒模型仅描述系统的输入输出。

以综合航电系统为例。假设综合航电系统由 3 个基本构件组成，分别是物理设备（比如空空导弹）、控制系统（比如悬挂物管理系统）和飞行员。分别对这些构件进行相应领域的分析，能够求精这些构件并识别出其它构件。比如对于控制系统级进行领域分析，除了识别控制系统是一个计算机系统之外，还能够识别物理设备、飞行员与控制器之间的接口构件，分别是设备接口（比如传感器、激励器）和人机接口。根据安全性需求开发框架，可以建立综合航电系统的分析等级和分析领域。比如环境级定义综合航电系统的范围，系统级识别综合航电系统的基本构件及其交互关系。

4 集成了安全性分析的需求开发过程

4.1 基本思路

基于系统思维开发软件安全性需求的一个基本出发点是构建软件的环境模型，建立自环境、经系统到软件的层次化分析结构。针对建构的综合航电系统模型，可以建立如图 2 所示的基于系统思维的需求分析结构。

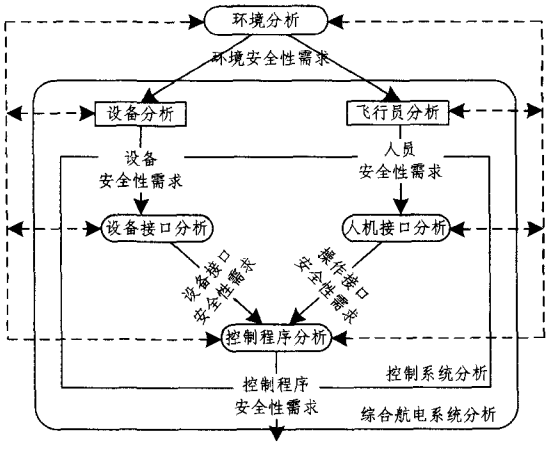

图 2 航电软件安全性需求分析结构

其中，闭环描述了分析阶段（每个分析阶段对应一个分析领域），箭头则表示了分析阶段之间的信息流（分析阶段之间的关系取决于分析领域之间的依赖关系）。各分析阶段的安全性目的定义如下：

• 环境分析阶段——生成综合航电系统的安全性目标声明，确定能够导致事故的系统失效行为。

• 设备分析阶段——识别可能的危险以及与安全性需求相关的设备属性。

• 设备接口分析阶段——描述了设备与控制系统之间的接口，规定设备接口必须具备的行为。

• 飞行员分析阶段——识别飞行员哪些操作能够导致系统失效行为，给出相应预防措施；

• 人机接口分析阶段——描述了飞行员与控制系统之间的接口，规定接口必须具备的行为。

• 控制程序分析阶段——根据构件属性及构件之间的交互作用建立一个控制程序的顶级组织。

利用上述分析结构，能逐步建立航电软件的环境模型，明确软件承担的系统安全性职责。同时也可以看出，为了开发综合航电系统软件（此处是控制程序）的安全性需求，需要从设备和飞行员两个视角进行分析。

针对每个分析等级内的每个分析领域，为开发软件安全性需求，本文采用一种集成安全性分析和需求分析的方法，如图 3 所示。

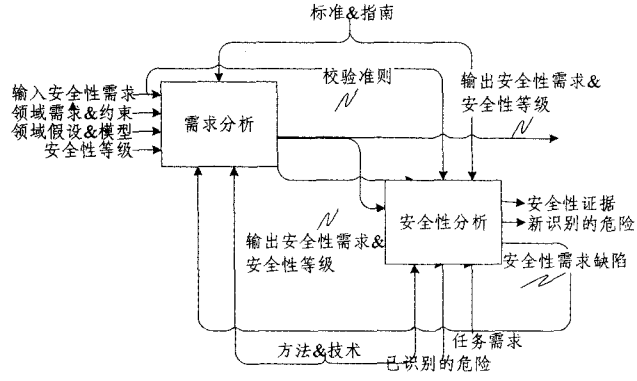

图 3 集成了安全性分析的软件安全性需求开发方法

其中，需求分析用来开发安全性需求，安全性分析用来检验由需求分析开发得到的安全性需求是否是风险可接受的。图 3 中各引脚的含义解释如下：

• 输入安全性需求——是上一分析等级开发的安全性需求。经过求精或缺陷修复后，开发出反映领域实体安全行为的输出安全性需求。

• 输出安全性需求——描述领域实体安全行为。

• 领域需求——是指附加在领域实体上的功能性需求或非功能性需求，是用户的任务需求在具体领域实体上的分解。领域约束是领域需求的一种特殊形式，是强制性需求。

• 领域假设——刻画有关领域实体行为的初步设定，但不一定准确。

• 领域模型——描述领域内构件及构件间的交互行为。这些模型能够在需求分析过程中求精，并成为输出安全性需求的一部分。

• 安全性等级——指出安全性需求在既定条件下利用既定措施满足或实现领域的安全性特征的可能性。

• 安全性需求缺陷——表明输出安全性需求与输入安全性需求不一致，或表明输出安全性需求未达到期望安全性等级。

• 标准和指南——比如 DO-178B、ARP4754 等，提供强制性安全性需求约束以及可用于需求分析、安全性分析的方法和技术。

• 校验准则——用于检查输出安全性需求是否达到了期望安全性等级。

• 任务需求——描述了用户对软件的顶级需求。

• 安全性证据——由安全性分析生成，用以确认输出安全性需求已经达到所需安全性等级，并与输入安全性需求一致。

• 危险——由安全性分析生成，说明输出安全性需求可能导致的危险。

4.2 需求分析

针对当前分析领域（如图 2 中的控制系统），需求分析以

前一个分析领域(如图 2 中的综合航电系统)开发的安全性需求为主要输入,以强加在领域实体上的需求和约束、领域实体的行为假设、领域模型和安全性需求应具有的安全性等级为辅助输入,在遵循某些标准或指南的基础上,依据某些分析方法,生成针对当前领域的安全性需求以及应具有的安全性等级。新开发的安全性需求是否满足期望的安全性等级,则需要根据指定的校验准则对其进行安全性分析。

值得说明的是,若提供和接收输入安全性需求的领域分别位于上下两个分析等级,如前者对应综合航电系统级,后者对应控制系统级,那么这个需求分析过程本质上是综合航电系统级安全性需求在控制系统上的求精过程。若提供和接收输入安全性需求的领域位于同一个分析等级,如前者是人机接口,后者是控制程序,那么这个需求分析过程就是将人机接口安全性需求传递并保持到控制程序上的过程,即是开发控制程序的安全性需求的过程。

针对某个分析领域进行需求分析,需要:①识别领域内的相关实体(这些实体可进一步分解);②规定实体行为及其之间的交互;③确认领域实体行为及其交互行为与领域行为相一致。

上述实质上是一个领域求精的过程,领域被求精为若干个领域实体,这些领域实体又可以视为新的领域并进一步分解。在刻画实体行为方面,可以从正常与异常两个角度进行,以便提供一个完整的行为空间,这将另文详述。当然并不是所有实体行为都能够产生安全性后果,这需要根据某些标准和指南,利用某些方法和技术进行分析。不过,规定实体的异常行为将有助于确定领域行为如何在其实体失效的情况下依然保持健壮性或何时启动例外处理程序来确保领域行为。毫无疑问,这有利于提高系统安全性。

一种普遍的观点是,采用形式化技术能够有效地提高软件产品的可信性。就需求分析阶段而言,如果采用形式化技术,那么就要求这些技术的特征和表达能力最好与需求分析活动相适应。从建模系统的视角看,目前的形式化技术大体上可以划分为两类:一类刻画系统应具备的属性,表达了对系统行为方面的必要约束,比如 KAOS、NFRF;另一类通过建立仿真模型,刻画系统运行机制,比如系统的非确定性机制和并发性等,可采用的技术包括时间 Petri 网、状态转移图等。

4.3 安全性分析

在每个分析领域中,若仅执行需求分析求精或开发安全性需求,是很难确定输出安全性需求是否能够实现相应的安全性等级的,或难以确定输出安全性需求是否仍然存在需求缺陷。由此,有必要对其进行安全性分析。

安全性分析以需求分析的输出安全性需求为主要输入,以任务需求、校验准则、已识别危险、期望安全性等级为辅助输入,在遵循某些标准和指南下,采用某些安全性分析方法和技术,①利用任务需求检验输出安全性需求的有效性;②验证输出安全性需求与输入安全性需求是否一致;③验证输出安全性需求是否实现期望的安全性等级。若安全性分析结果表明输出安全性需求与输入安全性需求、任务需求是一致的,并且实现了期望的安全性等级,那么给出相应的证据,以便于后期安全性认证。如果安全性分析结果表明输出安全性需求与输入安全性需求或任务需求不相一致,或者未能实现期望的安全性等级,那么识别输出安全性需求中存在的缺陷,特别是能够导致危险后果的缺陷,并反馈到需求分析过程,以重新生成包含修复了缺陷的安全性需求。这种在每个分析等级内所有分析领域内进行安全性分析的好处在于,尽早减少风险且抑制风险传播,有助于节省后续系统认证代价。

目前可用的安全性分析方法有很多,比如 PHA、HAZOP、PSSA、FTA、FMEA 等。以是否使用数据作为度量依据,这些方法可分为定量分析(比如基于失效率的 FTA 等)和定性分析(比如 PHA、HAZOP 等)两类。由因果推理关系看,已有安全性分析方法又可以分为归纳分析(比如 FMEA 等)和演绎分析(比如 FTA 等)两类。鉴于一方面现有标准多使用定性的术语表示安全性需求目标,另一方面在软件需求开发阶段获取详细的数据(比如某项软件需求缺陷被激活并导致危险后果的概率)要么比较难(很多数据都是假设的),要么数据不太可靠,此处采用定性的安全性分析方法,旨在:①确保系统不会由正常状态进入危险状态;②识别安全性需求规约中的缺陷,并分析这些缺陷可能导致的危险后果。

为了实现上述安全性分析目的,本文提出两个定性的安全性分析活动:初步分析和脆弱性分析。二者之间的交互关系如图 4 所示。

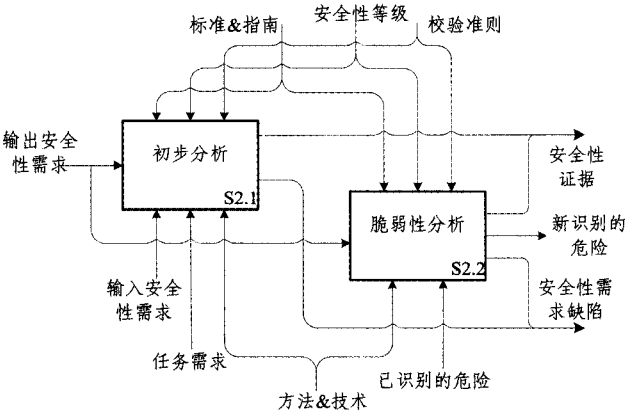

图 4 两种定性的安全性分析方法

4.3.1 初步分析

在明确定义的环境下,初步分析活动将验证和有效地确认输出安全性需求。需要完成的工作包括:①若提供和接收输入安全性需求的两个领域位于同一级,那么初步分析验证输入安全性需求和输出安全性需求的一致性,确保同级安全性需求不会相互冲突;②若提供和接收输入安全性需求的两个领域分别位于上下级,那么初步分析验证输出安全性需求是否符合输入安全性需求,确保上下级安全性需求的一致性;③利用校验准则和安全性等级,验证输出安全性需求是否存在缺陷和影响系统的安全行为;④检查确认输出安全性需求是否准确反映了强加在系统上的需求和约束,并确保安全性需求不与任务需求相冲突;⑤若验证和确认过程未能发现任何问题,即说明输出安全性需求满足需要、开发合理,那么就生成相应的证据。反之,则识别输出安全性需求中存在的缺陷以及相应的预防或修正措施,并反馈给需求分析。

4.3.2 脆弱性分析

前已述及,系统之所以会存在安全性问题是因为系统本身具有脆弱性。由此脆弱性分析旨在检查输出安全性需求中是否存在能够影响系统安全性的缺陷。一旦识别此类缺陷,

(下转第 418 页)

方法研究[J]. 计算机工程与设计,2011,32(1):133-137

[2] 宋现锋,刘军志. QoS 支持下的 GIS 服务链最优化问题研究[J]. 电子科技大学学报,2010,39(2):298-301

[3] 段玉倩,贺家李. 遗传算法及其改进[J]. 电力系统及其自动化学报,1998,10(1):39-52

[4] Ma Sai,Li Min-ruo,Du Wei-chang. Service Composition for GIS [C]// Proceedings-2008 IEEE Congress on Services,SERVICES 2008. PART 1,2008:168-175

[5] 刘书雷,刘云翔,张帆,等. 一种服务聚合中 QoS 全局最优服务动态选择算法[J]. 软件学报,2007,18(3)

[6] 蔡美玲,高春鸣. 基于树型编码的遗传算法在 Web 服务选择中的应用[J]. 计算机工程与应用,2007,43(31):214-218

(上接第 415 页)

就将其反馈到需求分析中,重新开发消除了缺陷的安全性需求,以降低系统进入危险状态的可能性。

归纳分析方法和演绎分析方法都可以用于输出安全性需求的脆弱性分析。较高的分析等级一般比较适合采用演绎分析方法,此时安全性需求信息比较通用、宏观、不涉及细节。而归纳分析方法则更适合对较低分析等级内的分析领域进行分析,此时有关安全性需求组成要素及其联系的信息比较具体、特殊。PHA 和 HAZOP 是两种可用于项目需求规约阶段的方法,并且这两种方法都可用于归纳分析或演绎分析。PHA 能够识别关键的系统功能和严重的系统危险;而 HAZOP 使用一些引导词表明规约如何被违背以及违背是否会导致危险等,是一种更为详细、系统的分析方法,可认为是一种重量级的 PHA。考虑到脆弱性分析活动是在项目早期的需求开发阶段开展的,此处给出如图 5 所示的支持 PHA、HAZOP 方法应用的脆弱性分析过程。

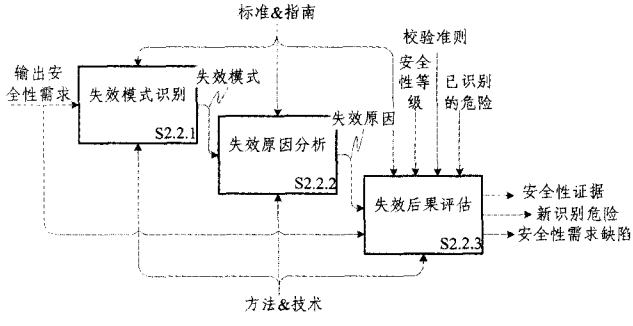

图 5 支持 PHA、HAZOP 的脆弱性分析活动

一旦脆弱性分析识别出具有能够导致危险后果的缺陷或者能够引起新的危险,就会将其提交给需求分析,以修正安全性需求。若脆弱性分析结果表明,输出安全性需求既不具有缺陷,也不会引入新的危险状态,那么将给出证据,表明输出的安全性需求已经满足校验准则。

结束语 本文采用系统建模与系统分析相结合的方式、基于系统思维建构了一个软件安全性需求开发框架,提出了一种集成了安全性分析的需求开发过程。该方法能够最大程度地约束安全性需求缺陷,防止其向同一分析等级内的其它领域或下一分析等级传播,同时有助于尽早地考虑重新生成安全性需求,减弱后续安全性分析的复杂性。此外,这种集成需求分析和安全性分析的策略能够不断生成证据,支持安全性论据和案例的构建,有助于减少后续认证代价。

尽管本文期望在需求开发阶段,采用需求分析和安全性分析相结合的方法来确保安全性需求开发的完整性、正确性、一致性,但毋庸置疑的是,该目标能否实现还受到下述 3 个约束的影响:①脆弱性分析并不能保证安全性需求依赖的所有假设都能得到严格遵守;②即使对安全性需求进行验证和确认,仍然会驻留一些未被发现的缺陷;③已有经验表明,验证工作本身也可能存在缺陷。可见,若可能,还应尽可能早地开展安全性评估活动,度量安全性需求的风险性,预测软件对系统风险的贡献。

参 考 文 献

[1] 褚文奎,张凤鸣,樊晓光. 综合模块化航空电子系统软件体系结构综述[J]. 航空学报,2009,30(10):1912-1917

[2] 422nd Test and Evaluation Squadron. Executive summary: aircraft accident investigation,F/A-22 S/N 00-4014[EB/OL]. http://www.f-22raptor.com/pdf/af.exsum_f22crash.pdf

[3] 樊晓光,褚文奎,张凤鸣. 软件安全性综述[J]. 计算机科学,2011,30(5):812-818

[4] Lutz R R. Analyzing software requirements errors in safety-critical,embedded systems[C]// Proceedings of the International Conference on Software Requirements. IEEE,1992:53-65

[5] McDermid J A. Software Safety:Where's the evidence? [C]// 6th Australian Workshop on Industrial Experience with Safety Critical Systems and Software(SCS 2001). Brisbane:Australian Computer Society,2001,CRPIT 3:1-6

[6] RTCA. DO-178B—1992 Software Considerations in Airborne Systems and Equipment Certifications[S]. Washington DC:Radio Technical Commission for Aeronautics,Inc,1992

[7] MoD. DEF-STAN 00-56 issue 4—2007 Safety Management Requirements for Defence Systems[S]. London:Ministry of Defence,2007

[8] 王拥军. 需求工程中的不确定性研究[D]. 西安:西北工业大学,2001

[9] Wu W. Architectural reasoning for safety-critical software applications[D]. Heslington:University of York,2007

[10] Leveson N G. An approach to designing safe embedded software [M]. London:Springer Verlag,2002,LNCS 2491:15-29

[11] Murray D P,Hardy T L. Developing safety-critical software requirements for commercial reusable launch vehicles[EB/OL]. http://www.faa.gov/about/office_org/headquarters_offices/ast/reports_studies/media/DMurray_SW%20REQTS_IAASS07_FINAl.pdf/

[12] Gharajedaghi J. System thinking:managing chaos and complexity:a platform for designing business architecture[M]. Boston:Elsevier,1999

[13] Leveson N G. A Systems-Theoretic Approach to Safety in Software-Intensive Systems[J]. IEEE Transactions on Dependable and Secure Computing,2004,1(1):66-86