基于RUCM的软件安全性需求描述方法

吴 雪 刘 超 吴 际
(北京航空航天大学计算机学院 北京 100191)

摘 要 随着软件在安全关键系统中的应用越来越广泛、承担的安全关键功能越来越多,软件的安全性需求变得越来越重要,成为系统安全性的一个重要的决定性因素。软件安全性需求的正确描述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。然而现有的安全性需求通常被混同于一般功能性描述中,缺乏独立、规范和明确的描述,缺乏对于故障、失效与安全性需求之间相互关系的描述机制,导致在实际应用中缺乏对安全性需求进行准确描述的方法。设计了一种基于结构化模版和约束规则的安全性需求规约,即基于RUCM的安全性需求描述方法Safety RUCM,该方法以RUCM建模方法为基础,通过扩展用例规约模板和限制规则,添加故障描述模版以及数据描述模版,使其能够支持故障相关描述以及相应的安全性需求描述并形成安全性需求规约,最后通过某机载操作系统的案例研究验证了Safety RUCM建模方法的可行性。

关键词 安全性需求,RUCM,数据字典,故障
中图法分类号 TP311.5 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.12.015

Safety Requirements Description Method Based on RUCM

WU Xue LIU Chao WU Ji
(School of Computer Science and Engineering, Beihang University, Beijing 100191, China)

Abstract Safety requirements have commanded increasing attention as software is playing a more and more important role in today's safety critical systems. The extraction and description of software safety requirement are the key element of the whole software safety work. The subsequent software design and realization and test process will reference to software safety requirement. Nevertheless, most safety requirements are described in ordinary functional specification, lack of independent and normative description, especially the relationship between safety requirements and fault, failure. As a result, there is little practical guidance on how to describe safety requirements. So this paper designed a safety requirements specification—Safety RUCM, which is based on restricted use case modeling RUCM, and extended its specification template and restriction rules by adding fault specification and data specification in order to support the fault related information. We used this specification to describe an operating system safety requirement. Result shows that this specification is practicable.

Keywords Safety requirements, Restricted use case modeling, Data dictionary, Fault

1 前言

随着软件在安全关键系统(Safety Critical System)的应用越来越广泛、承担的安全关键功能越来越多,软件的安全性需求变得越来越重要,它是系统安全性的一个重要的决定性因素。如在汽车、核能、火车等安全性关键系统领域,特别是航空领域等,软件要求很高的安全性,当系统的安全性无法得到满足时,往往会导致严重后果,带来巨大损失。安全性需求的确定与准确描述是确保软件安全的关键,因此对安全性需求的确定与准确描述具有必要性和迫切性。

安全性需求(Safety Requirements)指的是为防止非安全状态、降低故障产生的后果所采取的必要行动和约束[1]。软件安全性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[2]。

但是上述定义都较为抽象,在实际项目中,需求分析人员总是被到底什么才是安全性需求、如何来描述安全性需求这样的问题所困扰。鉴于此,本文试图对安全性需求进行分类、细化,并采用一种方法来清晰地、无二义性地描述安全性需求。

RUCM(Restricted Use Case Modeling)方法[11]是一种基于用例建模的需求建模方法,由用例图、改进的结构化的用例规约模版以及一组限制规则构成。实验证明,RUCM方法可以有效地降低需求规约的二义性,同时又保留自然语言易理解和易使用的优点。因此RUCM方法在软件需求建模方面

到稿日期:2015-02-03 返修日期:2015-05-07

吴 雪(1987-),女,硕士,主要研究方向为软件需求、软件测试,E-mail:wuxue890107@163.com;刘 超(1958-),男,博士,教授,博士生导师,主要研究方向为软件测试、面向对象技术、软件开发环境等;吴 际(1974-),男,副教授,主要研究方向为软件安全性与可靠性、嵌入式软件设计与验证、软件测试。

拥有显著优势。

RUCM是一种通用的需求描述方法，在描述安全关键软件的安全性需求时还存在不足，但是RUCM方法具有良好的扩展能力，可以通过扩展RUCM来支持安全性需求描述[12-14]。

2 安全性需求

2.1 安全性需求的分类

目前，航空安全领域所参考的重要标准是航空无线电技术委员会(RTCA)发布的DO-178C[3]。该标准定义了机载软件开发过程中各阶段软件制品所要达到的安全目标。该标准指出系统的安全性评估过程主要是确定系统可能的故障状态并对其进行分类，然后通过明确系统对各种安全故障的避免措施构建所需的免疫应对能力来确定安全性需求，以确保系统的完整性。这些需求被指定给硬件和软件，通过提供故障避免、故障检测、故障容错等来排除或限制故障的影响，以保证系统的安全性。其他安全性标准如61508[4]、0055[5]等也都需要对故障检测、故障应对等安全特性进行描述。

而软件安全性需求来自于系统的安全性评估过程，由此可见，安全性需求的准确描述与故障息息相关。本文认为可以从故障的角度将安全性需求分成3类：故障避免(Fault Avoidance)、故障检测(Fault Detection)、故障容错(Fault Tolerance)，如图1所示。其中安全性需求中重点关注的故障是可能引起严重失效的故障。

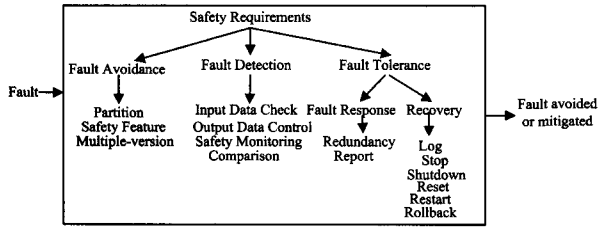

图1 从故障角度进行的安全性需求分类

故障避免是一种防患于未然的安全措施，系统识别出可能发生的故障后分析其发生原因，将发生条件破坏掉，使其不可能发生或者降低其发生概率。故障检测的安全性需求主要用来检测故障，以在故障发生后能以较短的时间发现该故障，从而阻止故障造成失效。故障容错的安全性需求主要体现的是失效安全性(fail-safety)和对系统的保护性(protective)，即在某些故障被触发(发生)后，系统仍然能够保持安全状态所需的安全性需求。故障容错类的安全性需求主要体现在故障应对与故障恢复上。故障应对行为包括向某些上级功能报告故障的发生、系统优雅降级、将某些功能设置为某些特定的状态或转换到备份的资源等。故障恢复主要指的是系统从故障中恢复的安全性需求，可能包括记录故障发生时的相关信息、停止某些功能的运行、重启某些功能、系统重配置或者回溯等。

2.2 现有的安全性需求描述中存在的问题

近年来，围绕安全性需求描述方法的研究有：基于场景的安全性需求描述方法[6]，该方法定义了一种图形化描述语言PSC(property sequence charts)来描述部件的线性时序逻辑特性；面向目标的安全性规约[7]方法，其采用面向目标的方法来识别并说明安全性需求系统风险，获取安全性目标，进而获得安全性需求；分级的安全性需求描述方法[8]，其对不同的安全性需求的重要性进行了区分；基于状态机的安全性需求描述方法[9,10]，其使用形式语言和状态机来描述系统的行为，通过添加安全性约束来描述系统的安全性需求。

在这些安全性需求描述方法中，对DO178C[3]、61508[4]、0055[5]等软件安全性相关标准中提到的故障检测、故障应对等安全特性缺少明确对应的描述。根据2.1节中的分析可知，安全性需求的准确描述与故障息息相关，故障避免、故障检测、故障处理等安全性需求是非常重要并需要明确指出的，因此本文提出了一种基于RUCM的安全性需求描述方法。

3 基于RUCM的安全性需求描述方法

针对以上安全性需求描述过程中存在的问题，本文设计了一种基于结构化模版和约束规则的安全性需求描述方法——基于RUCM的安全性需求规约Safety RUCM。该方法的步骤如下。

Step1 采用故障树分析FTA、功能危险分析FHA等方法来获得系统可能引发严重失效的故障。

Step2 使用本文提供的故障描述模版来描述该故障的相关信息。

Step3 从故障避免、故障检测、故障容错等方面来考虑控制该故障的安全性需求。

Step4 使用本文提供的安全性需求描述模版来描述安全性需求的相关信息，其中包括建立安全性需求与故障之间的联系以及使用数据字典来描述需求中的输入输出信息等。

Step5 检验安全性需求以及故障等信息的描述是否符合本文提出的安全性限制规则，如不符合，则应该重新检查描述的正确性或者说明信息的特殊性，即不符合规则但又合理的理由。

3.1 故障模型

故障是表达安全性需求所需的重要概念，在需求建模阶段，故障应该被显式地识别出来。故障在DO-178C中的定义为，故障是错误在软件中的表现，故障的发生可能导致失效[19]。失效在DO-178C[19]中的定义为，失效是指一个系统或者系统组件在特定的约束下不能够完成所需的功能。失效可能由故障的发生所导致。本文从故障的角度将安全性需求分成3类：故障避免(Fault Avoidance)、故障检测(Fault Detection)、故障容错(Fault Tolerance)，其中故障容错类安全性需求可进一步细分成故障应对和故障恢复类的安全性需求。因此，本文建立了故障模型，如图2所示。其中深色部分是RUCM中定义的概念，浅色部分是本文新识别的概念。

故障Fault可能引起失效Failure，故障可以被安全性需求SafetyUC避免、检测、容错。故障与安全性需求之间是一个多对多的关系，即一个故障可以被多个安全性需求避免、检测、容错；一个安全性需求也可以对多个故障进行避免、检测和容错。故障检测类安全性需求需要包含一类特殊的句子即FaultDetectionSentence，故障容错类安全性需求包含特殊的分支流：故障容错类的分支流FaultToleranceFlow及其子分支流故障应对分支流FaultResponseFlow和故障恢复分支流FaultRecoveryFlow。

安全性需求继承自原有的UseCase，SafetyUC通过建立起与故障之间的关系，间接地对失效进行了控制。SafetyUC还有一个属性SafetyType，用来表明SafetyUC的类型，Safe-

tyType 可以是故障避免类的 faultAvoid、故障检测类的 faultDetect 和故障容错类的 faultTolerance。

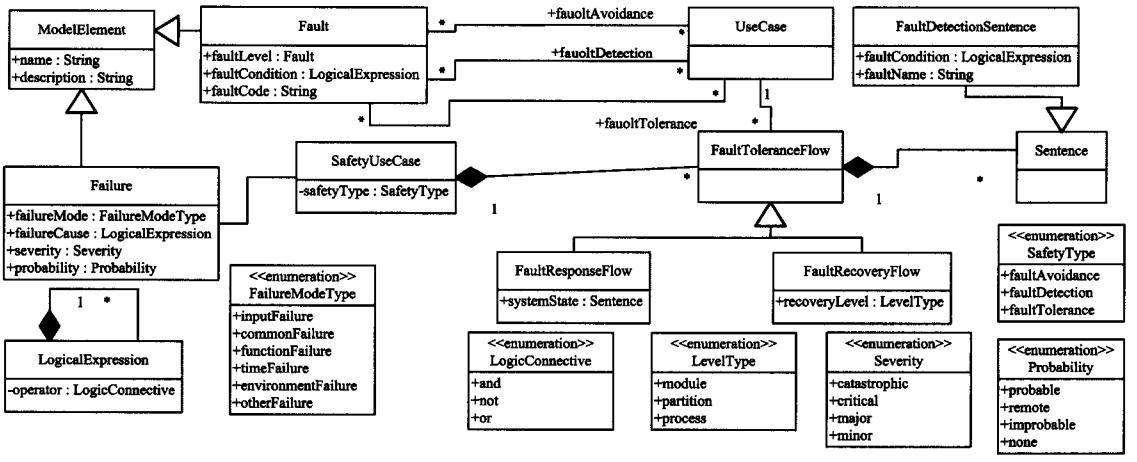

图 2　故障模型

故障 Fault 继承自 ModelElement，可从其父类 ModelElement获得属性 Name 和 Description，Name 表达的是该 Fault 的名字，Description 是对该 Fault 的概要描述。Fault 包含 faultLevel、faultCondition、faultCode 3 个属性。

faultLevel 表示故障等级，可以根据故障的影响来确定故障的等级。故障的等级是 LevelType 枚举类型，LevelType 有 3 种：Module(模块级)、Partition(分区级)、Process(进程级)，一个 module 级别的故障可能影响的是一个 module 中所有的分区，比如 module 初始化故障、特定系统功能执行中的故障、分区交换过程中的故障等；一个 partition 级别的故障影响的只是一个 partition，比如 partition 初始化过程中的故障、进程管理中的故障等；一个 process 级别的故障影响的是一个或多个进程，进程级别的故障有非法的 O/S 请求、进程执行故障(溢出、内存违反)等。

faultCondition：对故障存在状况的描述，即系统发生故障后的状态，亦即如果系统出现了该状态，说明故障已经发生。faultCondition 是用逻辑表达式 LogicExpression 来表示的，LogicExpression 的进一步细化表达见第 4 节安全性需求验证研究。

faultCode：故障码用来标识故障类型。虽然现有研究提出了对故障的分类研究，但本文关注需求描述，将不再对故障类型做进一步细化，允许使用字符串来描述。

FaultDetectionSentence：故障检测语句，主要用于描述系统进行故障检测的行为。故障检测类的安全性需求通常使用这类语句来描述，在特定的故障状态下能够说明某种故障的发生，这类安全性需求通常会使用到故障检测技术，该技术用来专门检测软件、硬件和环境中可能发生的故障。软件运行时若出现故障，检测软件能够立即做出响应。

FaultToleranceFlow：故障容错主要表达的是故障发生后，系统为保持安全状态所进行的一系列行为。故障容错可具体分为 FaultResponseFlow 和 FaultRecoveryFlow，其中 FaultResponseFlow 依赖于该故障被检测时的系统的操作状态，比如核心模块初始化、系统特定功能、分区交换、分区初始化或者进程管理、进程执行等，不同的系统操作状态下故障应对的需求可能不同。FaultRecovery 主要描述的是故障发生后的恢复行为，是系统返回正常操作模式的过程描述，确保发生故障的情况下，也能够通过一定的措施(如启动安全的备用选择)来保证系统的安全。

3.2　数据模型

部分安全关键软件应用于实时嵌入式系统，涉及对安全关键系统或设备的监控。这类软件从外界获得信息来进行分析、加工和处理，得出有用数据用以检测、决策或辅助决策。这些采集到的数据、分析处理过程中产生的中间数据和最终结果数据的内容和格式，其准确性、完整性等可能直接影响安全性，因此是必须关注的。0055 标准中指出如果安全性功能或安全特性依赖于数据的属性或者特征，那么数据的属性及其特征应该被显式地定义出来。DO178C 标准要求，软件应该正确应对输入数据故障以及阻止输出数据故障的发生。20438 标准也指出，应进行输入确认，即设置检查操作者输入数据的严格规则，拒收不正确的输入。而 RUCM 及其他规约并不支持数据相关的描述。因此，本文将数据字典与 RUCM 结合起来共同对安全性需求进行描述。

数据字典是系统中各类数据描述的集合，通过基于规则的数据定义方式来提高需求描述的准确性。同时，可读性规范和逻辑严谨也是数据字典的重要特点。数据字典支持描述基础数据和组合数据以及对数据的约束的定义，如图 3 所示。在需求分析阶段，数据字典一方面供分析人员来描述和定义系统需求层次需要识别的数据，另一方面与需求内容在一起构成了需求规格。

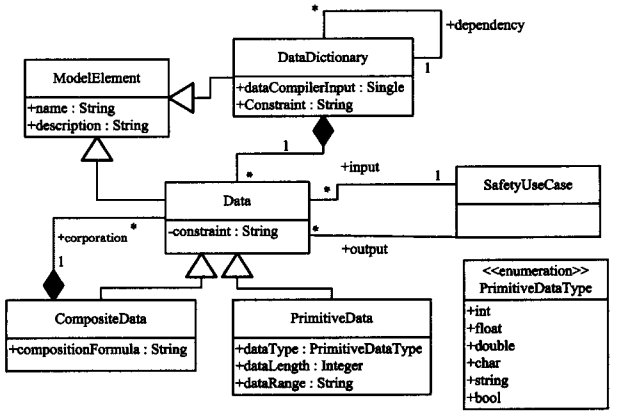

图 3　数据模型

定义绝大多数复杂事物，都是用已定义的基本事物的某

种组合来表示。从这个意义上说，定义是自顶向下的分解。所以数据字典中的定义就是对数据自顶向下的分解。

按照数据内容的组合特征来划分，数据字典中可以定义原子数据(PrimitiveData)和组合数据(CompositeData)。所谓原子数据就是无需再往下划分，且可以定义为给定的某些基本数据类型。组合数据则是由原子数据或者组合数据组成的数据，用来描述和表示需求规格中涉及的问题域概念。

为了更加简便地定义数据，数据字典提供了相应的定义操作符，如表1所列。

表1 数据字典中的使用符号

符号	含义	解释
=	被定义为	数据的定义规则
+	与	X=a+b，表示X由a和b组成
[…|…]	或	X=[a|b]，表示X由a或b组成
{…}	重复	X={a}表示，X由0或多个a组成
(…)	可选	X=(a)表示a可在X中出现，也可不出现
int，string，long等基本数据类型	基本数据元素	X=int，表示X为int类型的数据元素

3.3 安全性需求描述语句

在RUCM方法中定义了多种类型的语句来描述用例规约和事件流，针对安全性需求描述的需要，本研究工作对语句的类型进行了扩充，如图4所示(深色部分是RUCM中定义的概念，浅色部分是本文新识别的概念)。

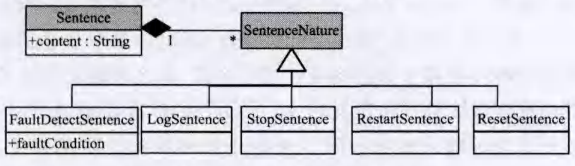

图4 故障角度进行的安全性需求分类

语句分为简单语句(SimpleSentence)和复合语句(ComplexSentence)。在RUCM的限制规则下，简单语句由主语、谓语、宾语构成；复合语句是由简单语句加上特定的关键词而形成的结构化语句，如IF-THEN-ENDIF分支语句。根据安全关键领域的软件系统的特点，需要为RUCM的语句类型进行扩充。

故障Fault继承自ModelElement，可从其父类ModelElement获得属性Name和Description，Name表达的是该Fault的名字，Description是对该Fault的概要描述。

FaultLevel表示的是故障的等级，可以根据故障的影响来确定故障的等级。

FaultCondition：对故障存在状况的描述，即触发故障的条件。

FaultCode：故障码用来标识故障类型。

FaultDetectionAction：故障检测行为，主要用于描述系统进行故障检测的行为，包含故障检测行为的需求是故障检测类的安全性需求。

FaultToleranceFolw：故障容错主要表达的是故障发生后，系统为保持安全状态所进行的一系列行为。故障容错可具体分为FaultResponse和FaultRecovery，其中FaultResponse依赖于该故障被检测时的系统的操作状态，比如核心模块初始化、系统特定功能、分区交换、分区初始化或者进程管理、进程执行等；FaultRecovery主要描述的是故障发生后的恢复行为，是系统返回正常操作模式的过程描述，是fail-safety的一个体现，即使在发生故障的情况下，也能够通过一定的措施(如启动安全的备用选择)来保证系统的安全。

3.4 用例规约模版与限制规则的扩展

这部分主要讲述对RUCM的规约模版与限制规则在Safety方面的扩展。

3.4.1 故障规约模版

首先根据领域模型中的故障相关模型，为RUCM添加故障规约模版，主要用于描述故障相关信息，模版如表2所列。

表2 故障规约模版

Fault Name	故障的名字
Fault Description	故障的概要描述
Fault Level	故障的级别
Fault Code	故障码
Fault Condition	故障条件
Cause Failure	可能引起的失效
Fault Avoidance	故障避免的需求
Fault Detection	故障检测过程
Fault Response	故障应对过程
Recovery Actions	系统从故障中恢复的需求

故障规约模版主要用来描述故障相关信息，包括故障的名字、故障的概要描述、故障的级别、故障码、故障条件、可能引起的失效、故障避免的需求、故障检测过程、故障应对和故障的恢复动作。Fault Condition应使用FaultConditionSentence来描述。其中Fault Avoidance、Fault Detection和Fault Response是安全性需求，在该故障模版中只需引用对应的安全性需求用例的名字即可。Recovery Actions是一类特殊的安全性需求，它引用RecoveryUseCase，它的事件流主要描述的是故障的恢复动作，主要由5类特殊的安全性语句来描述，包括LogSentence、ShutdownSentence、ResetSentence、StopSentence、RestartSentence。

3.4.2 数据字典模版

Safety RUCM扩展模版新增了一个数据字典模版，根据领域模型中的数据相关模型设计，其主要用于描述数据相关信息。数据字典模版如表3所列。

表3 数据字典模版

Data Dictionary Name	数据字典的名字
Data Dictionary Description	数据字典的概要描述
Dependency	数据字典对其他数据字典的依赖
Data Definition	数据定义1 数据定义2 ……

该表格的内容与领域模型中的数据相关模型一致，需要强调的是Data Definition中的每一行代表一个数据定义语句，所有的数据定义语句需要符合数据定义的语法规则，只有符合语法规则的语句才能够被自动识别出来。该数据字典模版中定义的数据将被用于安全性需求用例描述中。

3.4.3 安全性用例规约模版

安全性用例规约模版是对RUCM的标准用例规约模版的裁剪和扩展，其中的裁剪方式包括删除某些在嵌入式安全关键领域不必要的字段，基本的扩展方式包括增加新的字段和表格，或增加新的描述语法。扩展后的模版如表4所列。

Safety RUCM扩展模版增加了一个新的Input域和Output域，主要用于描述用例的输入和输出，其中输入和输出均来自使用数据模版定义的数据字典中的数据，即模版中填写

的数据应该能够在数据字典模版中找到。在该模版的 Basic Flow 以及各 Alternative Flow 中可以使用 3.3 节中提供的安全性需求描述语句来描述需求的事件流。

表 4 安全性用例规约模版

UseCase Name	用例名称，通常以动词开头	
Brief Description	用例内容的简要描述	
Precondition	用例的前置条件，即用例执行前必须满足的条件	
Dependency	依赖关系，描述与其他用例之间的包含、扩展关系	
Generalization	泛化关系，描述与其他用例之间的泛化关系	
Input	用例的输入	
Output	用例的输出	
Safety Type	指出安全性需求用例的类型	
Basic Flow	基本事件流，描述完全执行用例时的事件流程	
	Steps	事件流步骤
	PostCondition	基本流的后置条件
	特定分支流	
Specific	RFS	引用基本流中的步骤号码
Alternative Flows	Steps	事件流步骤
	PostCondition	分支流的后置条件

3.4.4 安全性限制规则

限制规则能够有效地降低自然语言的二义性，同时又保持自然语言易理解和易使用的特点，是 RUCM 方法中的重要组成部分。在安全性扩展的 Safety RUCM 方法中，标准 RUCM 中定义的 26 条限制规则仍然适用，但需要增加一系列新的限制规则来降低安全性相关文本的二义性。本研究工作为 Safety RUCM 定义了 10 条扩展规则，Safety-R1－Safety-R6 是用于填写用例规约模版的限制规则，SafetyR7－SafetyR11 是 4 条关键词规则。

Safety-R1：故障规约模版中 Fault Condition 必须使用满足逻辑表达式 Logic Expression 的语法规则的句子来描述。该规则保证 Fault Condition 能够被解析成逻辑表达式，便于后期对故障条件的一致性进行验证。

Safety-R2：故障规约模版中 Cause Failure 表示的是故障可能引起的失效，这里必须填写的是已经在模型中存在的失效的名字。该规则保证系统能够找到该失效，并且建立故障与该失效的联系。

Safety-R3：故障规约模版中 Fault Avoidance、Fault Detection、Fault Response、Fault Recovery 表示的是避免与该故障相关的安全性需求，这里填写的必须是已经在模型中存在的安全性需求的名字或者为空。该规则保证系统能够找到该安全性需求，并且建立故障与该安全性需求的联系。

Safety-R4：故障规约模版中，如果 Fault Avoidance 为空，那么 Fault Detection、Fault Response、Fault Recovery 要求均不为空。该规则保证的是，针对某故障，如果没有故障避免类的安全性需求，则一定要考虑故障检测以及故障应对和恢复的安全性需求。

Safety-R5：在安全性需求用例中，如果安全性需求用例的 Safety Type 是 FaultAvoid 类型，则需要明确指出要避免的故障是什么。该规则保证 FaultAvoid 类型的安全性需求能够明确要避免的故障。

Safety-R6：在安全性需求用例中，如果安全性需求用例的 Safety Type 是 FaultDetect 类型，那么必须出现故障检查类型的句子 When 〈fault condition〉 Fault 〈fault name〉 exist。该规则保证 FaultDetect 类型的安全性需求有实际的意义，的确进行了故障的检测，而不只是个标签。

Safety-R7：在安全性需求用例中，如果安全性需求用例的 Safety Type 是 FaultTolerance 类型，那么必须出现 FaultTolerance 类型的分支流，在该分支流中需要指出要容错的故障是什么。该规则保证 FaultTolerance 类型的安全性需求有实际的意义，的确进行了故障的检测，而不只是个标签。

Safety-R8：Fault Recovery Flow 数据恢复流必须包含 LogSentence、ShutdownSentence、ResetSentence、StopSentence、RestartSentence 这 5 类 Sentence 中的一种或多种。

Safety-R9：在填写数据字典模版时，Data Definition 中的每一个句子都应该符合 3.2 节中描述的数据定义的语法规则。该规则保证 Data Definition 中的每一个句子都能够被自动地识别，并且生成数据模型。

Safety-R10：When 〈fault condition〉 Fault 〈fault name〉 exists。

故障检测类的安全性需求中，当系统对故障进行检测时，必须用到此故障检测语句，该语句表示当〈fault condition〉被满足时，说明名为〈fault name〉的故障已经存在了。

Safety-R11：Log 〈fault name〉。

该语句表示在故障发生后记录故障名为〈fault name〉的故障信息。

4 案例研究

某机载操作系统是一个安全关键系统，其中进程是一个具有一定独立功能的程序关于某个数据集合的一次运行活动。它是操作系统动态执行的基本单元，既是基本的分配单元，也是基本的执行单元，很多安全关键的任务都是通过进程的执行来完成的。如果在需求阶段进程描述不清晰或者组成不明确，可能造成后期的设计和实现存在严重的问题，所以，在操作系统的需求规约中进程的属性及其特征应该被显示地定义出来，因此可以采用本文提出的用例规约模版中的数据模版对进程进行描述，如表 5 所列。

表 5 某机载操作系统的数据字典

Data Dictionary Name	某机载操作系统
Data Definition	进程＝进程名＋入口地址＋栈大小＋基本优先级＋周期＋时间容量＋截止期类型＋当前优先级＋截止期＋进程状态//描述的是进程相关的信息； 进程名＝string //定义进程的名字，同一分区内的每个进程的名字唯一； 入口地址＝int//指明进程的启动地址，即入口函数地址； 栈大小＝int // 确定进程运行时栈的大小； 基本优先级＝int //进程在创建时给定的优先级； 周期＝bool //确定周期进程的周期或标明该进程为非周期进程； 时间容量＝int//指明进程完成执行过程所需要的时间上限； 截止期类型＝[周期|非周期]//定义进程相关的截止期类型； 当前优先级＝int/进程当前的优先级，分区操作系统使用该优先级对进程进行调度。初始化时设置为基本优先级； 截止期＝time//进程执行完毕的截止期，即进程应该在该时间点前执行完毕； 进程状态＝[休眠态|就绪态|运行态|等待态]； 处理结果＝[成功|失败]//截止期超时处理结果；

进程的截止期是进程的一个重要属性，进程应该在该时

间点前执行完毕；如果超出了截止期可能导致安全关键的任务不能按时执行完毕，引发严重后果，因此截止期超时是操作系统应该重点关注的一个安全性故障。该故障应该显式地描述出来，可用故障规约模版描述，如表 6 所列。

表 6　截止期超时

Fault Name	截止期超时
Fault Description	进程没有在给定的截止期内完成执行会产生截止期超时的故障
Fault Level	Process
Fault Code	Deadline_missed
Fault Condition	系统时间≥进程->截止时间 and 进程->进程状态==运行态
Cause Failure	上层应用关键任务失效
Fault Avoidance Fault Detection	健康监控
Fault Response	处理截止期超时
Recovery Actions	处理截止期超时

在表 6 中，首先应该描述截止期超时故障的基本信息，然后描述与该故障有关的安全性需求。依据本文提出的安全性需求的分类，可以从 3 个方面进行提取。首先，有没有安全性需求能够避免此故障。截止期的设置要依据进程实际完成的功能及与分区间其他进程的关系，即使如此，进程的实际执行时间也有可能因分区内的异步时间而导致截止期超时[15]，也就是该故障不可能被完全避免。然后，考虑有没有安全性需求来检测此故障，即在故障发生后系统能够及时获知该故障发生的相关信息。此操作系统主要采取健康监控来检测此故障的发生。最后应该考虑的是故障发生后系统如何应对以及如何从故障中恢复，通过这些故障应对和故障恢复的安全性需求来保证系统的安全性。截止期超时故障的应对和恢复都是由处理截止期超时这个安全性需求来完成的，如表 7 所列。

表 7　处理截止期超时

UseCaseName		处理截止期超时
Brief Description		主要用于截止期超时故障的处理
Precondition		截止期超时故障发生
Generalization		硬截止期超时处理
Input		进程
Output		处理结果
Safety Type		Fault Tolerance
Tolerance Fault		截止期超时
Basic Flow	Steps	1. If(进程软截止期超时)
		2. Log 截止期超时
		3. Restart 进程
		4. Else
		5. Log 截止期超时
		6. Invoke 硬截止期超时处理
		7. END IF
	PostCondition	返回成功

结束语　本文设计了一种基于结构化模版和约束规则的安全性需求规约 Safety RUCM，该方法以 RUCM 建模方法为基础，通过扩展用例规约模板和限制规则，添加故障描述模版以及数据描述模版，使其能够支持故障相关描述以及相应的安全性需求描述形成安全性需求规约，并通过某机载操作系统的案例研究验证了 Safety-RUCM 建模方法的可行性。本文未来打算对使用该方法描述的安全性需求进行验证，主要从故障处理的完整性、故障条件的一致性、数据使用的一致性等方面来展开，以期能够帮助需求分析人员尽早发现安全性需求存在的问题，改正问题后得到完整的、一致的安全性需求。

参 考 文 献

[1] Hauge H J. A Survey of Software Safety[R]. Trondheim: Department of Computer and Information Science at the Norwegian University of Science and Technology, 2001

[2] Wu W H, Kelly T. Safety tactics for software architecture design[C]// Proc. of the 28th Annual Int'l Computer Software and Applications Conf. 2004

[3] Software Considerations in Airborne Systems and Equipment Certification: RTCA DO-178C[S]. Washington DC: RTCA, Inc, 2011

[4] International Electrotechnical Commission. Functional safety of electrical/electronic/programmable electronic safety-related systems-Part 2: Requirements for electrical/electronic/programmable electronic safety systems: IEC-61508[S]. London: International Electrotechnical Commission, 2010

[5] Requirements for safety related software defence equipment in defence equipment: Def Stan 00-55[S]. UK: Ministry of Defence, 1997

[6] J unwei D, X Zhong-wei, M Meng. Verification of Scenario-Based Safety Requirement Specification on Components Composition [C]// 2008 International Conference on Computer Science and Software Engineering. IEEE, 2008, 2: 686-689

[7] Navarro, Elena, Sanchez P, et al. A goal-oriented approach for safety requirements specification[C]// 13th Annual IEEE International Symposium and Workshop on Engineering of Computer Based Systems, 2006(ECBS 2006). 2006: 27-30

[8] Bounds A C. Safety requirements specification for new safety systems in older nuclear facilities in the UK[C]// 6th IET International Conference on System Safety. 2011: 1-5

[9] Jo H-J, Uiwang R, Hwang J-G. Formal Requirements Specification in Safety-critical Railway Signaling System[M]// Transmission & Distribution Conference & Exposition: Asia and Pacific. 2009. 2009: 26-30

[10] Troubitsyna E. Elicitation and specification of safety requirements[C]// Proc. of the 3rd Int'l Conf. on Systems. IEEE Computer Society, 2008: 202-207

[11] Yue Tao, Briand L C, Labiche Y. A Use Case Modeling Approach to Facilitate the Transition Towards Analysis Models: Concepts and Empirical Evaluation[M]// Model Driven Engineering Languages and Systems. Springer Berlin Heidelberg, 2009: 484-498

[12] Yue Tao, Briand L C, Labiche Y. Automatically Deriving a UML Analysis Model from a Use Case Model[M]. Carleton University, 2010

[13] Yue Tao, Briand L C, Labiche Y. Automatically Deriving UML Sequence Diagrams from Use Cases: Technical Report[R]. Carleton University, Canada, 2010

[14] Yue Tao, Briand L C, Labiche Y. Facilitating the Transition from Use Case Models to Analysis Models: Approach and Experiments[J]. ACM Transactions on Software Engineering and Methodology (TOSEM), 2013, 22(1): 5

[15] Li Yun-xi, Shi Lei, Ren Xiao-rui. Design and Iimplementation of Process Managemen In Partition [J]. Aeronautical Computing Technique, 2005, 35(4): 12-15