面向机载软件适航审查的软件计划阶段的证据模型

袁 巍 吴 际 刘 超 杨海燕
(北京航空航天大学计算机学院 北京 100191)

摘 要 适航认证对机载软件至关重要,178C标准是重要的适航认证安全标准。Order8110.49指南提出了介入审查的适航认证方法,但是目前还未有对介入审查中软件计划阶段的178C目标符合性证据的研究。文中基于软件计划阶段需要审查的目标和DO-178C标准中针对目标的特征描述,提出了3种模型(标准证据模型、项目制品模型、项目相关证据模型),并通过项目相关证据模型向项目证据数据模型的转换生成证据信息检查单,确定了证据信息的来源。建立证据模型的审查方法为软件计划阶段的审查提供了证据收集的指导,减少了证据收集过程对审定方审定人员的依赖,提高了审查效率。最后,通过一个机载飞行显示器软件的案例说明了提出的证据模型具有可用性与有效性。

关键词 机载软件,适航审查,证据模型,DO-178C,Order8110.49

中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.09.007

Evidence Model Oriented to Airborne Software Airworthiness Review of Software Planning Stage

YUAN Wei WU Ji LIU Chao YANG Hai-yan
(School of Computer Science and Engineering, Beihang University, Beijing 100191, China)

Abstract Airworthiness certification is essential for airborne software. The 178C standard is an important safety certification standard for airworthiness. Order 8110.49 guideline sets out the method of airworthiness certification, but there is no research on the 178C target compliance evidence of the software planning phase involved in the review. Based on the objectives of the software planning phase and the feature description of the DO-178C standard, three models were proposed: standard evidence model, project-artifact model and project-related evidence model. And evidence information checklist is generated by converting the project-related evidence model to the project-related evidence data model to determine the source of the evidence information. The review method for establishing the evidence model provides guidance for the collection of evidence for the review of the software planning phase, reducing the reliance on the auditor's review process and improving the efficiency of the review. And the availability and validity of the proposed evidence model were llustrated by an Airborne-Flight-Display software.

Keywords Airborne software, Airworthiness review, Evidence model, DO-178C, Order8110.49

1 引言

机载软件是安全关键软件(Safety Critical Software)。许多国家都规定安全关键机载设备的软件系统或与飞机交联的软件系统都必须进行认证,否则不得使用。

安全认证通过提取和列举一系列的证据来证明系统满足相关的安全性目标或要求,进而证明系统达到了可以让人接受的安全程度[1]。目前,应用最广泛的是美国航空无线电技术委员会(RTCA)提出的航空工业的软件标准《机载系统和设备合格审定中的软件考虑》RTCA DO-178B[2]。但是,随着当前软件代码开发新技术的不断出现,软件开发需要新的验证和认证方法,因此工业界和学术界联合建立了一套新的认证体系DO-178C[3],其已被美国联邦航空管理局认可。DO-178C针对不同安全等级的软件分别提出了一系列目标。机载软件审查人员需要依据软件生命周期过程中产生的各种数据和信息,通过一定的论证过程和证据说明其符合DO-178C中提出的目标。除此之外,郑军等人[4]指出,DO-178C中提出了一系列新技术,包括基于模型开发的技术手段、面向对象以及相关技术等内容。DO-178C对软件开发生命周期中的所有活动以及活动达到的目标都有相关的规定。

为监控机载软件整个生命周期的过程,确保其符合DO-178C中的相关要求,FAA在Order8110.49——《机载软件批准指南》[5]中提出了178C过程目标符合性方法,并引入了介入审查(SOI)的概念,其在第3节"Software Review Process"

到稿日期:2017-07-28 返修日期:2017-09-19 本文受民用飞机专项科研项目(MJ-S-2013-10)资助。

袁 巍(1991-),男,硕士,主要研究领域为软件安全性与可靠性,E-mail:yuan_wei@buaa.edu.cn;吴 际(1974-),男,博士,副教授,主要研究领域为软件安全性与可靠性等,E-mail:wuji@buaa.edu.cn(通信作者);刘 超(1958-),男,博士,教授,主要研究领域为软件测试等;杨海燕(1974-),女,硕士,讲师,主要研究领域为软件工程等。

中对 SOI 审查进行了明确的定义。SOI 审查分为 4 个审查阶段:软件计划阶段审查(SOI#1)、软件开发阶段审查(SOI#2)、软件验证阶段审查(SOI#3)和软件最终阶段审查(SOI#4)。为每个阶段分配一定的 178C 目标,通过收集证据论证是否符合所分配的 178C 目标要求,并且只有上一个阶段审查的目标均符合目标要求才能进入下一个阶段的审查。

软件计划阶段审查是整个评审过程的首个审查过程,软件计划的优良直接关系到软件开发过程和软件测试过程的进度和效率。安全证据(Safety Evidence)是指为安全目标论证提供支持的数据、信息和制品[6]。审查人员从软件生命周期数据中抽取证据以论证符合 178C 安全目标,但是在适航审查过程中存在着 178C 目标和相关描述比较抽象、实际项目中证据信息不明确、证据的收集依赖于审查人员对安全标准的主观理解、收集到的证据的有效性和充分性较低以及可能产生冗余证据等问题。

因此,本研究的目标是:依据安全标准,分析软件计划阶段的目标、进行的活动和产生的软件生命周期数据,构建目标到软件项目制品的证据模型,建立证据模型到证据数据模型的转换,以获得证据信息检查单,确定证据信息的来源,减少证据收集过程对审查人员的依赖。

2　相关研究

FAA Order 8110.49 中引入了介入审查的概念,定义了软件计划阶段、软件开发阶段、软件验证阶段和软件最终审定阶段 4 个阶段的审查过程来论证 178C 目标的符合性。崔利杰等[7]分析了 A 级软件的软件计划阶段审查过程,指出计划过程建立了各种软件计划、标准、流程、活动、方法,以及开发、检验、控制、保证所需要的工具;在制定软件计划阶段审查要点时提出了内容完整性——审查要点与 178C 保持一致,前后一致性——不仅包括计划与标准的一致性,还包括计划和标准与系统安全性分析结果、实施过程输出数据的一致性,定义明确性和 178C 中提出的特殊考虑。朱宇蒙等[8]指出软件计划阶段评审(SOI#1)是几次适航评审的基础,SOI#1 评审判定软件计划、标准、符合性矩阵是否满足符 178B,SOI#2—SOI#4 评审判定软件研制过程的实施是否遵循软件计划、标准、符合性矩阵;他同时指出民用飞机的主制造商往往对供应商进行若干次工程审查,主要包括计划阶段评审(PPR)、初步设计评审(PDR)、关键设计评审(CDR)、测试就绪评审(TRR)和符合性评审(CR)。PPR 评审在 SOI#1 之前进行,并给出 PPR 工程评审与计划阶段评审的关系。

统一建模语言(Unified Modeling Language)为设计人员提供使用图形化描述多种模型的方法,包括类模型、用例模型和活动图模型等。其中,类模型提供了描述数据静态结构的方法,现有的 UML 建模工具包括 IBM Rational Architect,StarUML等,支持可视化的 UML 模型图绘制,以及各种不同视图间的分组管理等,侧重于对模型的描述,但缺少对模型实例对象(objects)的处理功能。模型化开发框架为程序员提供通过图形建模方法将模型自动生成代码的途径,一个典型的例子是 Eclipse Modeling Framework[9](EMF)。EMF 支持用户通过图形界面建立 UML 类模型,并自动生成对应的 Java 类源码文件,在一定程度上支持模型化编程。

一般的安全性证据可以分为基于过程的证据和基于产品的证据。Nair 等[10]对安全性证据做出了完整的归纳,指出基于过程的证据多数是根据软件生命周期活动收集的过程活动信息。基于软件项目产品的证据主要是收集软件活动产生的软件制品(software artifact)信息。

Panesar-Walawege 提出了基于模型的证据收集方法,通过对安全标准抽取标准证据模型(Standard Evidence Model,SEM)来指导评估人员进行证据收集[11]。SEM 包含了安全标准的安全证据概念类型和结构,这些概念是通过分析安全标准获取的,并不依赖于具体的项目数据制品的领域概念,具有较好的可复用性[12];但由于其并不包含某些在特定项目的原始数据中出现的具体属性和结构,因此 SEM 的实用性不强。项目相关证据模型(Project-Specific Evidence Model,PSEM)是对 SEM 在特定项目背景下的具体化,既描述了安全标准所定义的证据类型和关系,又包含了原始数据的属性、类型和结构。PSEM 一方面对 SEM 进行细化,一方面依赖项目制品模型(Project Artifact Model,PAM),后者定义了过程原始制品的类型和数据结构。

3　机载软件适航计划阶段的审查过程

按照 DO-178C 的目标和 8110.49 规定,软件适航的审查流程分为以下 4 个阶段。

1)软件计划阶段审查应在初始软件计划过程完成时(即在完成并评审了大部分的计划和标准时)发起。这一审查一般被称为介入阶段(SOI#1)。

2)软件开发阶段审查通常在 50%以上的软件开发数据(即需求、设计和代码)完成并评审后发起。这一审查一般被称为 SOI#2。

3)软件验证阶段审查通常应在 50%以上的软件验证和测试数据完成并评审后发起。这一审查一般被称为 SOI#3。

4)软件最终审定阶段审查应该在确定了软件最终版本,完成了软件验证,进行了软件符合性审查,软件准备好并正式提出系统审查申请后发起。这一审查一般被称为 SOI#4。

软件计划阶段是软件生命周期中的初始阶段,主要任务是规划并定义用于开发软件生命周期数据的软件计划、标准、程序、活动、方法和工具。计划阶段评审主要是通过评审软件计划阶段的数据和系统安全性评估的相关结论,来评估计划阶段过程对 DO-178C 文件附录的表 A-1、表 A-8(1-4)、表 A-9(1)和表 A-10(1-2)目标的符合性。SOI#1 评审的软件生命周期数据及对应的要求如表 1 所列。

表 1　SOI#1 评审的软件生命周期数据

Table 1　Software life cycle data in first review stage of SOI

软件生命周期数据	DO-178C 章节
软件合格审定计划(PSAC)	11.1
软件开发计划(SDP)	11.2
软件验证计划(SVP)	11.3
软件验证结果(SVR)	4.6,11.14
软件构型管理计划(SCMP)	11.4
软件质量保证计划(SQAP)	11.5
软件需求、设计、编码标准	11.6,11.7,11.8

软件计划阶段评审是一个迭代过程，一般采用远程评审和现场评审相结合的评审方式。在整个评审阶段，相关软件生命周期数据的评审可能会持续进行几轮。当评审方认可并批准相关软件生命周期数据后，SOI≠1 评审方可退出。

4 证据模型

在模型驱动开发中，为了提高模型的可复用性，研究人员提出了平台无关模型（Platform-Independent Model，PIM）和平台相关模型（Platform Specific Model，PSM）的相关概念[13]。其中，PIM 是独立于特定平台特性而用于描述软件需求的业务逻辑的模型，PSM 是 PIM 在特定平台上的具体化。

基于这一模型分层的思想，根据模型与标准和具体项目的相关程度的不同，本文以 A 级软件计划阶段审查目标为前提条件，定义了如下 3 类与证据收集相关的模型。

1）标准证据模型。标准证据模型定义了论证安全标准中的安全目标所需的证据类型和结构，提取 SEM 的过程不依赖于特定项目的原始数据特征。

2）项目制品模型。该模型描述和定义了特定项目软件过程中所产生的原始制品的类型和数据结构。

3）项目相关证据模型。项目相关证据模型是在特定项目制品上对标准证据模型进行具体化，该模型一方面定义了论证安全标准所需的证据信息类型和关系，另一方面包含了原始制品所特有的数据类型和结构。

在建立项目相关证据模型的基础上，给出一种项目相关证据模型向项目证据数据模型转换的方法，并生成证据信息检查单。

4.1 标准证据模型

4.1.1 软件生命周期过程的证据模型

如第 3 节所述，在软件计划阶段审查 DO-178C 文件附录的表 A-1、表 A-8(1-4)、表 A-9(1)和表 A-10(1-2)目标的符合性。

表 A-10 中的 1)和 2)目标的表述如下：

1) Communication and understanding between the applicant and the certification authority is established.

2) The means of compliance is proposed and agreement with the Plan for Software Aspects of Certification is obtained.

目标 1)是建立申请者与审定方的联络，目标 2)是提出符合性方法并制定软件合格审定计划，两目标与其他目标的联系较少，因此与其他目标分离并单独列出，在标准证据模型中加入相关证据。

表 A-1 中目标 1)－3)是检查软件生命周期过程的符合性目标，其表述如下：

1) The activities of the software life cycle processes are defined.

2) The software life cycle(s), including the inter relationships between the processes, their sequencing, feedback mechanisms, and transition criteria, is defined.

3) Software life cycle environment is selected and defined.

建立的软件生命周期过程的证据模型如图 1 所示。

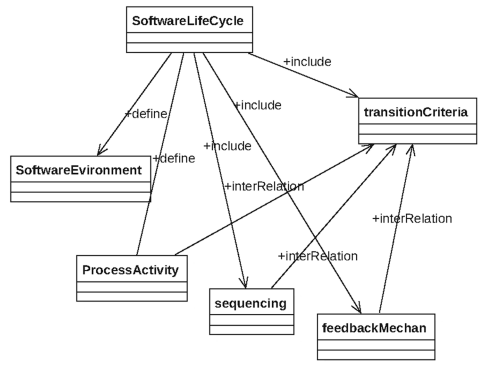

图 1 软件生命周期过程的证据模型

Fig. 1 Evidence model of software life cycle process

软件生命周期中定义了软件生命周期环境、过程活动，其与软件生命周期序列、反馈机制和迁移准则存在包含关系。过程活动、软件生命周期序列和反馈机制分别与迁移准则存在内在关系。

4.1.2 软件计划过程的证据模型

软件计划阶段关于计划和标准的目标有表 A-1 中的第 4)－7)个目标和表 A-9 中的第 1 个目标，其表述为：

1) Software development standards are defined.

2) Software plans comply with this document.

3) Additional considerations are addressed.

4) Development and revision of software plans are coordinated.

5) Assurance is obtained that software plans and standards are developed and reviewed for compliance with this document and for consistency.

建立的证据模型如图 2 所示。

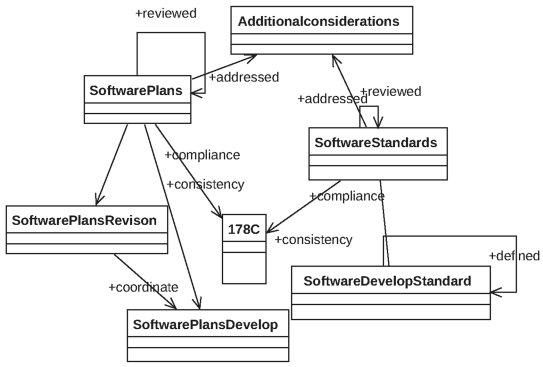

图 2 软件计划过程的证据模型

Fig. 2 Evidence model of software planning process

软件计划和软件标准经过评审后需符合 178C 标准并与 178C 保持一致，软件计划的更改和软件计划的开发需保持协同关系，软件计划中需定义软件开发标准，软件计划和软件标准需要解决额外考虑的问题。

4.1.3 软件配置管理过程的证据模型

检查软件配置管理过程的目标有表 A-8 中的第 1)－4)个目标，其描述为：

1) Configuration items are identified.

2) Baselines and traceability are established.

3)Problem reporting,change control,change review,and configuration status accounting are established.

4)Archive,retrieval,and release are established.

分析以上 4 个目标,软件配置管理过程的证据模型如图 3 所示。

软件配置管理过程活动包括识别配置项、建立基线、追踪关系、问题报告、更改控制、配置状态报告、存档、检索和发布信息。

由以上 3 种证据模型和表 A-10 中的 2 个目标的证据组合而成的模型则是软件计划阶段审查的标准证据模型,其结构如图 4 所示。

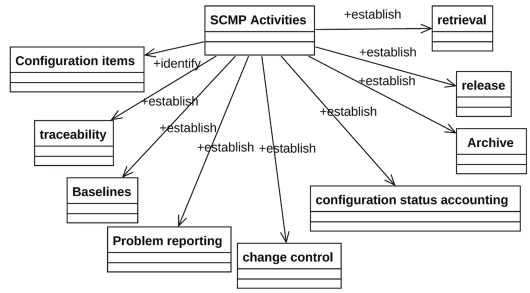

图 3　软件配置管理过程的证据模型

Fig. 3　Evidence model of software configuration management process

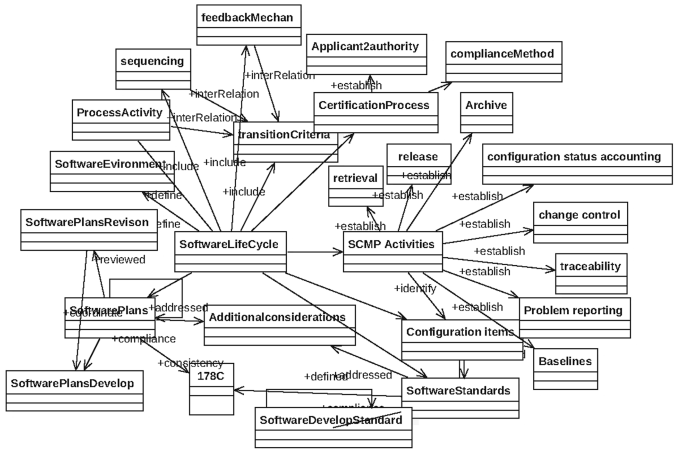

图 4　软件计划阶段审查的标准证据模型

Fig. 4　Standard evidence model of software planning stage review

4.2　项目制品模型

项目制品是指在具体的软件项目中产生的需求文档、测试文档、代码等软件生命周期数据。经第 3 节分析,计划阶段审查的主要项目制品有:软件合格审定计划(PSAC)、软件开发计划(SDP)、软件验证计划(SVP)、软件验证结果(SVR)、软件配置管理计划(SCMP)、软件质量保证计划(SQAP)、软件需求标准(SRS)、软件设计标准(SDS)、软件编码标准(SCS)。根据 178C 第 11 章对以上项目制品的定义,可以得出如图 5 所示的项目制品模型。

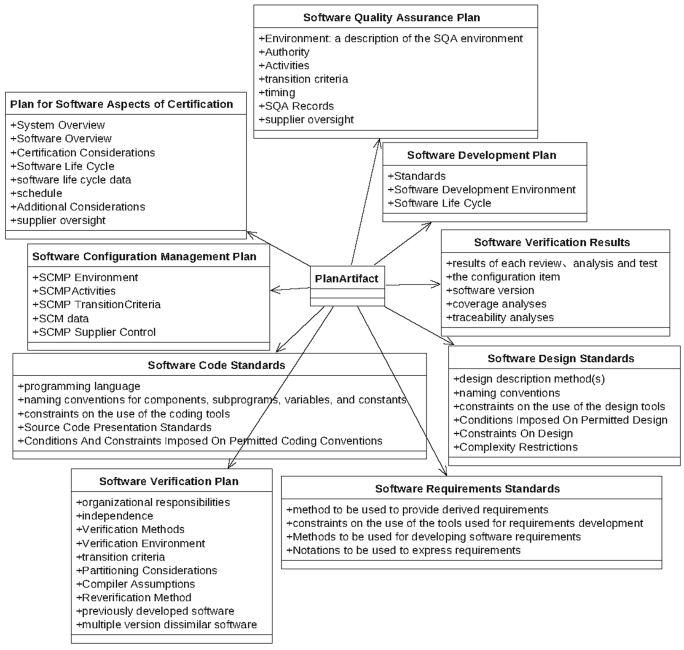

图 5　项目制品模型

Fig. 5　Project artifact model

PlanArtifact 类代表计划阶段软件制品，它与上述 9 类制品直接关联，每类制品的属性在类的成员属性中表示，如 SQAP(Software Quality Assurance Plan)中有 7 种属性，分别为软件质量保证环境描述(Environment：a description of the SQA environment)、授权当局(Authority)、软件质量保证活动(Activities)、迁移准则(transition)、时间选择(timing)、软件质量保证记录(SQA Records)、供应商监管(supplier oversight)。根据项目制品性质，每一种属性可以继续展开为一个类。由于篇幅原因，在此不再展开论述。

4.3 项目相关证据模型

项目相关证据模型是建立在标准证据模型上，根据项目制品模型对标准证据模型的细化。下面给出软件计划阶段的 3 类项目相关证据模型。

通过图 5 的项目制品模型对图 1 的软件生命周期证据模型进行细化，可以获得如图 6 所示的软件生命周期过程的项目证据模型。在 SoftwareLifeCycle 类中添加了两个引用类型：文档 PSAC 中的 Software Life Cycle 属性和 SDP 文档中的 Software Life Cycle 属性。在 transitionCriteria 类中增加了引用类型 SQAP 文档中的 transitionCriteria 属性和 SVP 文档中的 transitionCriteria 属性。其他类的增加属性如图 6 所示。

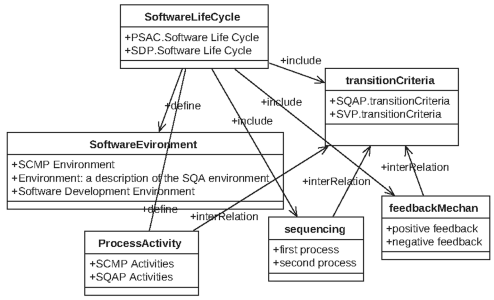

图 6 软件生命周期过程的项目相关证据模型

Fig. 6 Evidence model of software life cycle process in specific project

限于篇幅，图 7 和图 8 分别给出了软件计划过程的项目相关证据模型和软件配置管理过程的项目相关证据模型。通过对图 6－图 8 这 3 个项目相关证据模型的分析可合成最终的软件计划阶段项目证据模型，如图 9 所示。

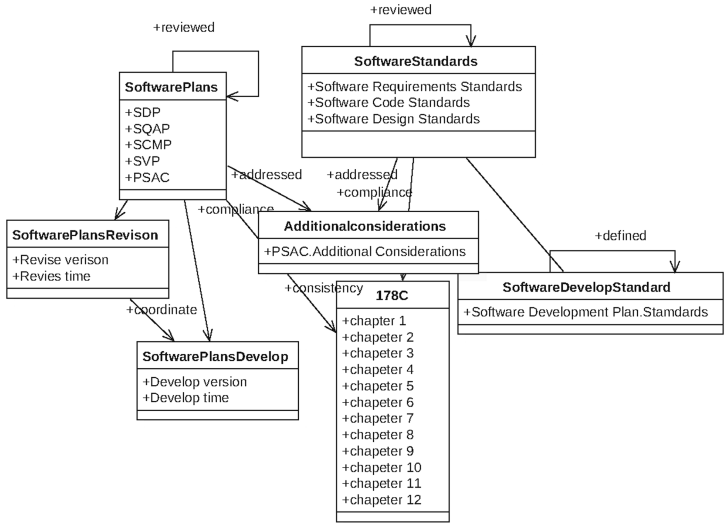

图 7 软件计划过程的项目相关证据模型

Fig. 7 Evidence model of software planning process in specific project

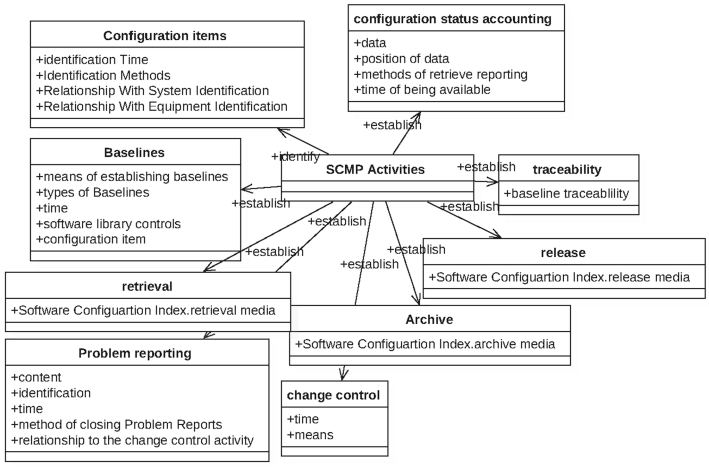

图 8 软件配置管理过程的项目相关证据模型

Fig. 8 Evidence model of software configuration management process in specific project

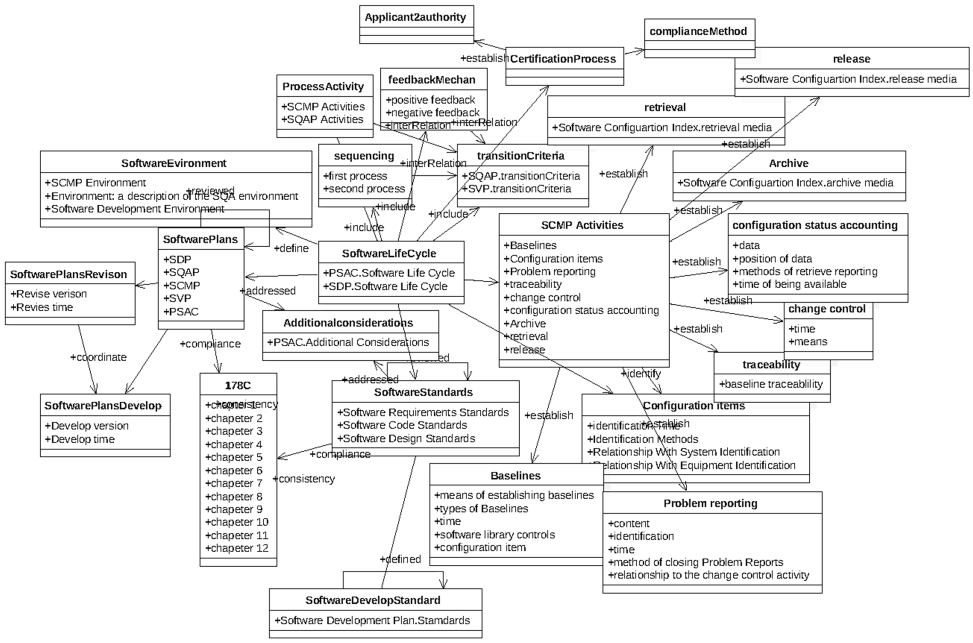

图 9　软件计划阶段的项目相关证据模型

Fig. 9　Evidence model of software planning stage in specific

4.4　项目相关证据模型向项目证据数据模型的转换

由 4.1 节－4.3 节的建模方法可以得到软件计划阶段的项目证据模型,但是无法直接通过证据模型获得想要的证据。本节从模型向代码转换的角度,给出一种项目证据模型的实例化方法,以获得证据数据模型,生成的证据信息检查单也为下一步对证据的自动化管理和验证提供了一种实现途径。

现有的一些非开源的建模工具可以支持模型向代码的转换,但是无法表示模型中元素之间的相互关联关系,可扩展性较低。下面介绍证据模型向证据数据模型转换的过程。如图 10 所示,starUmL 工具生成的模型文件格式为此模型工具特有的 mdj 格式文件,但是查阅 starUml 的相关资料发现,其结构是一种复杂的 JSON 格式的数据。因此,可以利用 Java 编程语言对 JSON 格式数据的支持来解析证据模型文件。

图 10　starUml 模型格式

Fig. 10　Data format of starUML model

4.4.1　构建证据树

通过上述分析,尝试使用一种建立树形结构的方式将证据模型中的关联关系表示出来。图 11 是树节点的结构图,如 Tree:TreeNode,左边是命名,右边是 Tree 的类型。一个 Tree 由 Content 和 Childlist 组成,Content 代表节点的属性值,Childlist 代表节点的子属性集合。

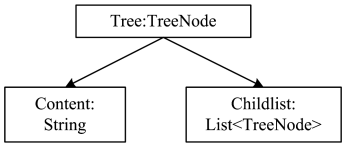

图 11　证据模型数据结构

Fig. 11　Data format of starUML model

图 12 给出了构建证据树的流程。构建证据树的算法如下:

1)新建一棵树,初始化树根节点。

2)读取顶层模型类成员属性。

3)判断类成员属性是否为空,若为空则退出,证据树构建完成,否则进入步骤 4)。

4)判断类成员属性是否已经被访问,若被访问则返回步骤 2),读取下一个顶层模型类属性;否则进入步骤 5)。

5)将类成员属性加入树节点中。

6)判断类成员属性是否还有子属性,若没有子属性则返回步骤 2),继续读取下一个成员属性;否则进入步骤 7)。

7)遍历子属性,将子属性加入树节点中。

8)返回步骤 6)。

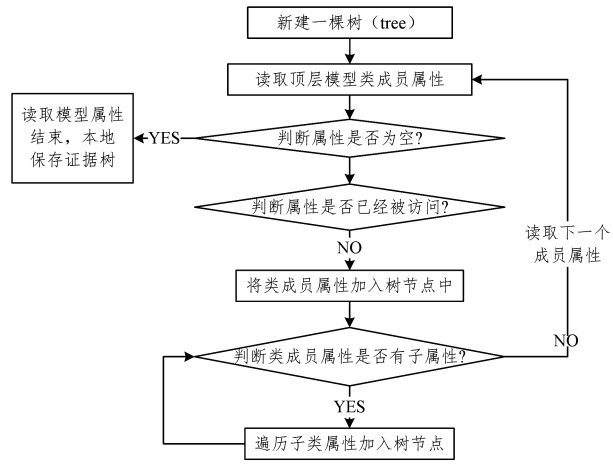

图 12　证据树流程图

Fig. 12　Flowchart of evidence tree

通过上述构建证据树的方法,可以将证据模型中的证据信息完整地保存到本地文件中,本文称这棵证据树为证据数据模型。读取证据信息的操作通过读取证据树完成。

4.4.2 证据树向证据信息检查单的转换

4.4.1 节得到的证据树是由 Java 代码实现的，如图 13 所示，在 Tree 中建立一个 searchEvi 方法来循环查找需要的证据。图 14 为 SOI＃1 的项目证据模型，粗线框内为 SCMP Activities 类。例如若要查找 SCMP Activities 的证据，则将 key 赋值为 SCMP Activities(Software Configuration Management Plan Activities)，并将其作为输入条件，则输出的证据如图 15 所示，SCMP 活动的证据信息链为 1，2，3…分层展开的结果，展开的层次越多表明证据信息越详细。通过调取具体的证据信息，来生成具体的证据信息检查单，以确定证据的来源。

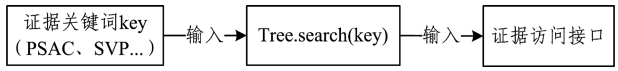

图 13 证据接口

Fig. 13 Evidence interface

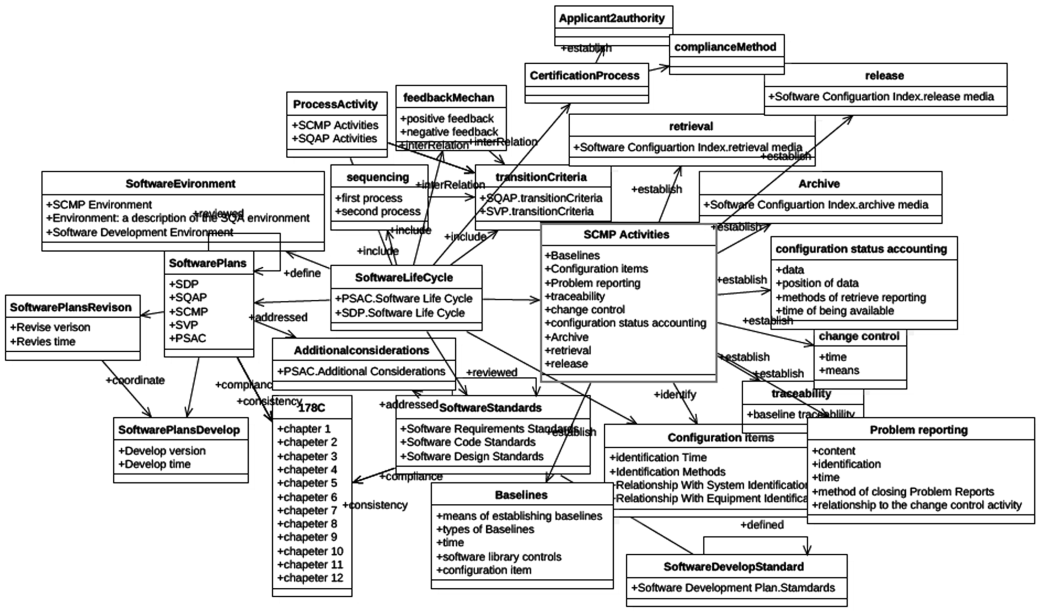

图 14 SCMP Activities 类图

Fig. 14 SCMP Activities class

读取成功!
1-SCMP Activities
2-Baselines
3-means of establishing baselines
3-tpyes of baselines
3-time
3-software library controls
3-configuration item
2-Configuration items
3-identification time
3-identification methods
3-relationship with system identification
3-relationship with equipment identification
2-Problem reporting
3-content
3-identification
3-time
3-method of closing problem reports
3-relationship to the change control activity
2-Traceability
3-baseline traceability
2-Change control

图 15 SCMP Activities 证据信息检查单

Fig. 15 Evidence information checklist of SCMP Activities

5 案例研究

本文以某机载飞行显示器软件为例，在对其进行软件适航第一阶段审查时，依据本文提出的建模方式，成功地建立了针对该机载软件的计划阶段审查目标的证据模型，并获得了证据数据模型，确定了该软件适航审查第一阶段的证据来源。

机载飞行显示器软件提供给审查的软件制品有：PSAC、高层需求、SCI、配置管理计划、(soi1)编码标准、soi1 软件开发计划——11.2 软件开发环境、设计标准——6.7 堆栈、软件需求标准——2.5 需求属性、验证计划——8 编译器假设、质量保证计划——2.2QA 职权。根据第 4.1 节的方法建立软件计划阶段的证据标准模型。根据 4.2 节的方法，在建立项目制品模型的过程中首先要筛选出软件计划阶段要审查的制品，如表 2 所列，去掉在这一阶段无需审查的高层需求和 SCI 两个制品，整理每个制品的子项个数，建立项目制品模型。根据 4.3 节的方法，关联证据标准模型和项目制品模型，建立项目相关证据模型。根据 4.4 节的证据模型向数据模型的转换方法，将模型文件转换成包含具体证据信息的证据检查单。根据证据检查单对软件第一阶段的关于 178C 附录表 A-1(1-7)、表 A-8(1-4)、表 A-9(1)和表 A-10(1-2)共计 14 个目标的符合性进行评审。证据检查单上的每一项内容关联着对应的 178C 目标，通过评审者填写项目制品的页码数和制品内容确定是否存在证据，来判定检查单关联的 178C 目标是否满足，并确定制品中缺少的证据制品。案例结果表明，在该飞行显示器软件项目中通过了 6 个目标，未通过目标 8 个，可以发现制品中支持目标符合性的证据所缺失的制品项有 45 项，无法明确支持目标的制品项有 20 项。将其与由审定方审查人员在同等条件下收集证据和审查的结果进行比较与分析，结果表明本文方法对证据收集和审查的准确度能够达到同等以上水平。

以上案例说明,本文提出的软件适航审查第一阶段的证据模型能够被成功地应用到实际的适航评审工作中,能够确定证据的来源,并能减少对审定方审定人员的依赖,提高了证据收集的准确度和适航审查的效率。

表 2　某机载飞行显示器的软件制品

Table 2　Airborne flight monitor software artifact

项目制品名称	制品简称	制品子项数
PSAC	PSAC	42
soi1 软件开发计划——11.2 软件开发环境	SVP	78
配置管理计划	SCMP	74
质量保证计划——2.2QA 职权	SQAP	42
验证计划——8 编译器假设	SVP	36
软件需求标准——2.5 需求属性	SRS	18
设计标准——6.7 堆栈	SDS	44
(soi1)编码标准	SCS	27

结束语　本文提出了面向机载软件适航审查软件计划阶段的证据模型,通过模型化驱动架构的方法建立证据模型,并给出证据模型向证据数据模型的转换,生成证据信息检查单,以达到确定 178C 目标证据来源的目的。本文提出了 3 种模型,分别为标准证据模型、项目制品模型和项目相关证据模型,覆盖了软件计划阶段审查涉及到的主要目标。证据数据模型为证据信息的可追踪和管理提供了一种途径。最后,以真实的工业案例说明了利用本文提出的建立证据模型的方式来确定证据来源的可行性。接下来的工作重点在于将建立证据模型的方法扩展到适航审查的其他几个阶段与研究证据的自动化管理和验证技术的工作中。

参 考 文 献

[1] BOZZANO M, VILLAFIORITA A. Design and Safety Assessment of Critical Systems[M]. Auerbach Publications, 2010.

[2] RTCA DO-178B. Software considerations in airborne system and equipment certification[S]. Washington D. C. : RTCA, 1992.

[3] RTCA DO-178C. Software considerations in airborne system and equipment certification[S]. Washington D. C. : RTCA, 2008.

[4] ZHENG J, HUANG Z Q, XU B F. Current progressand prospectsof airworthinesscertificationstandards[J]. Computer Engineering and Design, 2012, 33(1): 204-208.

[5] FAA Order 8110.49. Software approval guidelines[S]. Washington D. C., 2003.

[6] WEAVER R, DESPOTOU G, KELLY T, et al. Combining Software Evidence: Arguments and Assurance[C]// SIGSOFT Software. England, 2004: 152-160.

[7] CUI L J, REN B, LI Z. Airborne Software Airworthiness Review Based on DO-178B/C [J]. Journal of Command and Control, 2016, 2(1): 84-88.

[8] ZHU Y M, JIN P, SUN Q Y, et al. Research of airborne software plan phase review [J]. Aeronautical Science & Technology, 2014, 25(8): 5-8.

[9] STEINBERG D, BUDINSKY F, PATERNOSTRO M, et al. Eclipse Modeling Framework[M]. US: Addison-Wesley Professional, 2008: 62-210.

[10] NAIR S, DE LA VARA J L, SABETZADEH M, et al. Classification, Structuring, and Assessment of Evidence for Safety -- A Systematic Literature Review[C]// 2013 IEEE Sixth International Conference on Software Testing, Verification and Validation(ICST). 2013: 94-103.

[11] FALESSI D, SABETZADEH M, BRIAND L, et al. Planning for Safety Evidence Collection: A Tool-Supported Approach Based on Modeling of Standards Compliance Information[C]// IEEE Software. 2011: 849-860.

[12] PANESAR-WALAWEGE R K. Using Model-Driven Engineering to Support the Certification of Safety-Critical Systems[D]. Norway: University of Oslo, 2012.

[13] STAHL T. Model-Driven Software Development: Technology, Engineering, Management [M]. New York: John Wiley & Sons., 2006: 20-50.

(上接第 51 页)

分配到不同的物理机上,从而提高用户通信的平均带宽,改善用户体验。

参 考 文 献

[1] FELTER W, FERREIRA A, RAJAMONY R, et al. An updated performance comparison of virtual machines and linux containers [C]// 2015 IEEE International Symposium on Performance Analysis of Systems and Soft-ware(ISPASS). IEEE, 2015: 171-172.

[2] SEO K T, HWANG H, MOON I, et al. Performance comparison analysis of linux container and virtual machine for building cloud [C]// Networking and Communication. 2014: 105-111.

[3] VERMA A, PEDROSA L, KORUPOLU M, et al. Large-scale cluster management at Google with Borg[C]// Proceedings of the Tenth European Conference on Computer Systems. ACM, 2015: 18.

[4] 张阜兴. 知乎万级规模容器平台架构和实战[EB/OL]. (2016-11-18) [2017-07-11]. http://www.infoq.com/cn/presentations/platform-architecture-and-combatof-zhihu-container-platform.

[5] WANG H, SHI P, ZHANG Y. Jointcloud: A Cross-cloud copperation Architecture for integrated internet Service Customization[C]// IEEE, International Conference on Distributed Computing Systems. IEEE, 2017: 1846-1855.

[6] CUI W, ZHAN H, LI B, et al. Cluster as a Service: a Container based Cluster Sharing Approach with multi-user Support[C]// 2016 IEEE Symposium on Service-Oriented System Engineering (SOSE). IEEE, 2016: 111-118.

[7] BERNSTEIN D. Containers and cloud : From lxc to docker to kubernetes[J]. IEEE Cloud Computing, 2014, 1(3): 81-84.

[8] BURNS B, Grant B, Oppenheimer D, et al. Borg, omega, and kubernetes[J]. Communications of the ACM, 2016, 59(5): 50-57.

[9] MARMOL V, JNAGAL R, HOCKIN T. Networking in containers and container clusters[J/OL]. https://www.mendeley.com/research-papers/networking-containers-container-clusters1.

[10] The Kubernetes Authors. Kubernetes OpenVSwitch GRE/VxLAN networking [EB/OL]. [2017-07-08]. https://kubernetes.io/docs/admin/ovs-networking.

[11] MERKEL D. Docker: lightweight linux containers for consistent development and deployment[OL]. http://docs.docker.com.

[12] Docker Inc. Docker Documentation [EB/OL]. [2017-07-10]. https://docs.docker.com.