软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2024,35(9):4265−4286 [doi: 10.13328/j.cnki.jos.007136]

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

基于 MTRDL 的自动飞行系统模式需求建模与验证方法*

徐 恒[1], 黄志球[1], 胡 军[1], 陶传奇[1], 王金永[2], 石 帆[1]

[1](南京航空航天大学 计算机科学与技术学院, 江苏 南京 210016)

[2](徐州工程学院 信息工程学院, 江苏 徐州 221018)

通信作者: 黄志球, E-mail: zqhuang@nuaa.edu.cn

摘 要: 在民机自动飞行过程中, 自动飞行系统模式转换是影响安全的重要因素, 随着现代民机机载系统的功能与复杂度的快速增长, 在需求阶段对自动飞行系统模式转换的安全性分析和验证成为重要的挑战. 飞行模式转换的复杂性不仅体现在自动飞行过程中必需的多重飞行模式之间的交互关系, 还体现在模式转换与外部环境之间复杂的数据与控制交联关系, 这些交联关系同时隐含了飞行模式转换的安全性质, 这些特征提高了形式化方法的应用难度. 提出一种领域特定的建模验证框架: 首先, 提出面向自动飞行系统模式转换的领域需求建模语言 MTRDL 和基于该语言扩展于 SysML 上的建模方法; 其次, 提出基于安全需求模板的安全性质辅助规约方法; 最后, 通过对某机型的若干条目化需求的实例研究, 证明所提方法在自动飞行系统模式转换需求验证中的有效性.

关键词: 自动飞行系统模式; 形式化方法; SysML 建模; 安全性质

中图法分类号: TP311

中文引用格式: 徐恒, 黄志球, 胡军, 陶传奇, 王金永, 石帆. 基于MTRDL的自动飞行系统模式需求建模与验证方法. 软件学报, 2024, 35(9): 4265–4286. http://www.jos.org.cn/1000-9825/7136.htm

英文引用格式: Xu H, Huang ZQ, Hu J, Tao CQ, Wang JY, Shi F. Requirement Modeling and Verification for Automatic Flight System Modes Based on MTRDL. Ruan Jian Xue Bao/Journal of Software, 2024, 35(9): 4265–4286 (in Chinese). http://www.jos.org.cn/1000-9825/7136.htm

Requirement Modeling and Verification for Automatic Flight System Modes Based on MTRDL

XU Heng[1], HUANG Zhi-Qiu[1], HU Jun[1], TAO Chuan-Qi[1], WANG Jin-Yong[2], SHI Fan[1]

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

[2](School of Information Engineering, Xuzhou University of Technology, Xuzhou 221018, China)

Abstract: During the automatic flight of civil aircraft, the transition of automatic flight system modes is an important factor affecting safety. With the rapid growth of functions and complexity of modern civil aircraft airborne systems, the safety analysis and verification of automatic flight system mode transition in the requirement phase has become an important challenge. The complexity of flight mode transition is not only reflected in the interaction among multiple flight modes necessary during the automatic flight but also in the complex data and control cross-linking relationships between the mode transition process and the external environment. Additionally, these cross-linking relationships imply the safety properties of the flight mode transition process, which increases the application difficulty of formal methods. This study proposes a domain specific modeling and verification framework. First, a modeling language MTRDL for transition requirements of automatic flight system modes and a modeling method based on extended SysML language are put forward. Secondly, the safety property-assisted protocol method based on safety requirement templates is proposed. Finally, the effectiveness of the method in the requirement verification of automatic flight system mode transition is proven by a case study of a certain aircraft's itemized requirements.

Key words: automatic flight sytem mode; formal method; SysML modelling; safety property

* 基金项目: 国家自然科学基金 (U2241216); 中央高校基本科研业务费专项资金 (NT2022027, NJ2022027); 河南省科技攻关项目 (222102210048)

本文由"形式化方法与应用"专题特约编辑曹钦翔副教授、宋富研究员、詹乃军研究员推荐.

收稿时间: 2023-09-11; 修改时间: 2023-10-30; 采用时间: 2023-12-13; jos 在线出版时间: 2024-01-05

CNKI 网络首发时间: 2024-05-13

自动飞行系统是民机飞行过程中辅助飞行员控制飞机的核心系统, 指引飞机按照预先设定的程序或者根据环境条件选择合适的飞行控制律来实现自动飞行[1]. 而自动飞行系统模式实现了飞行员和自动飞行系统人机接口的交互, 它不但描述了自动飞行系统的行为, 是对飞行任务的抽象规约, 还是连接飞行任务和具体飞行控制律之间的桥梁. 因此自动飞行系统模式转换逻辑不仅是自动飞行系统开发时的设计准则和要求, 也是飞行员操作和飞行手册的保障依据, 飞行模式转换的错误会导致严重的危害后果, 如模式设定错误引发的斯特拉堡 A320 空难[2]、自动油门模式混淆导致的韩亚 214 号空难[3]等. 自动飞行系统模式转换需求作为转换逻辑的载体, 其验证工作不仅是机载软件开发过程中的关键环节, 也是适航安全标准的要求[4].

机载软件需求主要采用自然语言描述, 而自然语言天然存在二义性和模糊性以及不完整性, 且难以进行自动化分析处理. 在使用自然语言描述需求进行开发的过程中, 工程师浪费了大量的时间和人力对需求文档进行人工审查, 以确保需求文档上下文的一致性和精确性, 同时由于缺少精确描述需求的方法, 在确认和验证需求时难以实现自动化过程[5]. 因此对系统需求进行精确的形式化建模和分析, 可以有效保障系统开发质量, 并支持在开发早期对需求展开分析和验证, 已成为学术界和工业界的共识[6–8]. 形式化方法是一种基于数理逻辑对软硬件系统进行描述的技术, 通过精确的概念定义系统的数学模型, 可以不运行系统进行验证. 传统的仿真测试方法主要用于系统设计后期的验证, 而形式化规约和验证技术不仅支持在系统设计前期进行验证, 而且能够通过穷尽系统状态空间对系统模型进行验证从而保证验证的正确性和完整性, 因此采用形式化方法对民机自动飞行系统模式转换需求展开建模验证是必要的[9].

在国际适航安全标准中, 对模型驱动工程和形式化方法用于基于需求的研发也提出了强制要求. 1992 年 RTCA 发布的 DO-178B《机载系统和设备合格审定中的软件考虑》标准中, 提出机载软件的研发过程需要重点保证包括系统需求、软件高级需求、软件低级需求等在内的需求正确性[10]. 在最新版的 DO-178C[11]中, 又进一步要求在软件开发过程中需要以高级需求和低级需求为核心展开多层级的安全性分析与验证工作, 在进一步的修订 DO-333[12]中提出形式化方法必须作为安全性验证手段之一. 然而, 这些安全性标准都只给出了适航审定的抽象目标, 不涉及具体技术方法. 如何应用相应的形式化方法理论用于实际的工程应用领域, 从而形成航空工程中适用且有效的方法和工具, 对机载软件需求进行分析与验证, 在国内外仍然都是一个非常大的挑战.

随着集成建模开发工具的发展, 航空领域开始追求需求、设计和验证阶段的统一建模语言和工具, 以形成系统的框架方法. 从需求建模方法来看, 机载软件领域当前采用的方法主要有 3 类. 第 1 类是以 UML[13]、SysML[14]、AADL[15]等为代表的图形化统一建模语言及其扩展, 如: 2013 年 Insaurralde 等人在 SmartFuel 项目中基于 AADL 对飞机燃油系统进行了建模验证[16]; 2020 年, 波音基于 SysML 建模语言提出了包含建模、模拟、设计、支付这 4 个阶段的 Diamond 模型; 2021 年, 美国柯林斯公司在 AADL 框架基础上综合了自动验证插件并用于 OpenUxAS 无人机项目[17]. 这类方法优点是面向复杂系统工程领域设计, 涵盖了系统架构的全生命周期且具备通用性和可扩展性, 缺点是由于半形式化语言在解释/使用上存在部分二义性, 不能直接支持形式化的属性验证[18]. 第 2 类是以 Simulink [19]和 SCADE [20]工具为代表的基于同步数据流的图形化建模语言, 如: 2005 年, 美国柯林斯公司基于 Simulink 对 ADGS-2100 自适应显示和制导窗口管理器进行了建模[21]; 2009 年, 法国空客基于 Lustre 语言和 SCADE 工具对飞行控制系统进行了建模和验证[22]. 这类方法优点是与设计联系密切且具有数学模型支持, 缺点是这类方法的模型通常包含明显的设计细节, 本质上是一份系统详细设计模型, 不具备满足 DO-178C 标准中多层级需求分析和验证的能力, 此外这些工具为国外商业软件其技术细节黑盒不可控[23]. 第 3 类是受限自然语言形式的需求建模方法, 即基于特殊语义限定词, 采用结构化方法形成模板化的自然语言, 通常用这类方法来描述的需求都是采用条目化的管理方法, 如: Zhang 等人提出规则约束用例规约中自然语言使用的限定用例建模方法 RUCM (restricted use case modeling)[24], 并成功从限定自然语言需求过渡到 UML 分析模型[25]; 南京航空航天大学团队提出一种限定自然语言需求模板, 并基于该模板自动生成 AADL[26]. 这类方法优点是更贴近实际工程中自然语言表达需求的习惯, 缺点是模板设计工程量巨大且在后续设计和验证中仍然要转换为其他建模语言. 无论是哪一类需求建模方法, 在将形式化方法应用于航空领域时, 主流工作都集中于如何构建更加可用和鲁棒的工具以支持可复用的形式化规约和验证, 推动形式化方法工具的集成和可复用库设施的构建[27].

本文工作的主要贡献在于: 提出一种模式转换领域专用的形式化需求建模语言 MTRDL, 和基于该语言的 SysML Profile, 并在该建模语言的基础上实现对安全性质的形式化验证. 该工作为形式化方法在航空工程需求验证的应用提供了有效范例.

本文第 1 节对自动飞行系统模式转换的关键特征和概念、模型检验方法等基础知识进行了概述. 第 2 节给出了面向自动飞行系统模式转换问题的需求建模语言 MTRDL 的形式化定义, 以及基于 FMSysML 的自动飞行系统模式转换建模方法. 第 3 节中给出了基于 FMSysML 的安全需求模板, 以及基于该模板的安全性质辅助规约方法. 第 4 节给出了实例研究与结果的分析讨论. 最后是相关研究分析与未来工作.

1 背景知识

1.1 自动飞行系统模式转换相关概念

自动飞行系统 (AFS) 按照预先设定的飞行程序或根据环境条件变化及飞行员指令选择合适的飞行模式来控制飞机自动飞行. AFS 控制飞机姿态、航向、导航、速度等, 因此飞行模式转换与 AFS 中的其他子系统之间有着复杂的交联关系; 这些子系统包括: 飞行管理系统 (FMS)、飞行指引仪 (FD)、自动驾驶仪 (AP)、飞行员人机接口 (CI)、飞机状态传感器数据 (ASSD) 等. 其中飞行控制面板 (FMCP) 和主飞行显示器 (PFD) 组成连接飞行员和自动飞行系统模式之间的人机接口. 飞行员可以通过 FMCP 中的按钮选择需要的飞行模式, 对飞行指引仪和自动驾驶仪进行打开或关闭操作等; 主飞行显示器 PFD 上需要正确及时的显示各类飞行模式相关的信息, 如: 飞行指引指令、飞机俯仰、横滚的控制指令、模式是否被选择以及自动驾驶仪是否已经开启的通知等.

飞行模式转换逻辑以及飞行控制律共同构成飞行导引系统 (FGS). 本质上, AFS 飞行模式可以定义为一组自动飞行系统互斥行为的系统配置, 每一个模式对应一种系统行为. AFS 包含的飞行模式种类繁多, 模式转换逻辑复杂. 从软件角度来看, 自动飞行系统模式转换逻辑可以看作是一组处理离散数据的算法集, 其关键功能是在 AFS 系统处于激活状态的任意时刻, 选择合适的飞行控制律来完成对飞机运动状态的控制.

如后文图 1 所示, 民机自动飞行系统模式至少包括 4 个维度: 垂直模式类、水平模式类、自动油门模式类和多轴模式类, 每一类模式又根据不同机型的飞行任务要求, 划分为若干具体模式, 每一个具体模式又细分为可能的多个子模式. 在每一个飞行阶段, 飞机的运行都是由多个飞行模式之间的组合来进行导引控制, 以飞机进近着陆阶段为例, 不同飞行阶段自动飞行系统模式之间的组合切换还和其他系统存在关联. 飞行模式转换的复杂性不仅体现在自动飞行过程中必需的多重飞行模式之间的交互关系, 还体现在模式转换与外部环境之间复杂的数据与控制交联关系, 这些交联关系同时隐含了飞行模式转换的安全性质. 因此, 自动飞行系统模式转换具有耦合兼容的多维度复杂静态结构; 交互合作和过渡切换的多层次模式动态组合; 以及需要安全风险分析和安全论证过程的安全攸关等领域特征.

1.2 时序逻辑与模型检验

给定系统模型 M 以及一个时序逻辑描述的系统属性 ϕ, M 是含有多个状态的标记迁移系统 $(S,\rightarrow,L)$, ϕ 随着 M 的状态演变可能会改变其真假. 计算树逻辑 (computation tree logic, CTL) 是一种常用的时序逻辑, 一个 CTL 公式可以用 Backus Naur 范式表达如下:

$$\phi := p \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \phi \rightarrow \phi \mid AX\phi \mid AF\phi \mid AG\phi \mid A[\phi U \phi],$$

其中, 每个时序连接词都是一对符号. A 表示“沿着所有路径”, X 表示“neXt 状态”, F 表示“某个未来状态”, G 表示“所有未来状态”, U 表示“直到”. 对模型 M 上的状态 s, $(M,s) \vDash \phi$ 被称为模型 M 在状态 s 满足 ϕ, 语义为:

(1) $(M,s) \vDash \phi$ 当且仅当 $p \in L(s)$.

(2) $(M,s) \vDash \neg\phi$ 当且仅当 $(M,s) \vDash \phi$ 不成立.

(3) $(M,s) \vDash \phi_1 \wedge \phi_2$ 当且仅当 $(M,s) \vDash \phi_1$ 且 $(M,s) \vDash \phi_2$.

(4) $(M,s) \vDash \phi_1 \vee \phi_2$ 当且仅当 $(M,s) \vDash \phi_1$ 或 $(M,s) \vDash \phi_2$.

(5) $(M,s) \vDash \phi_1 \rightarrow \phi_2$ 当且仅当 $(M,s) \vDash \phi_1$ 不成立或 $(M,s) \vDash \phi_2$.

(6) $(M,s)\vDash AX\phi$ 当且仅当对所有满足 $s\to s'$ 的 s', $(M,s')\vDash AX\phi$.

(7) $(M,s)\vDash AF\phi$ 当且仅当 s 延伸的所有路径 $s\to s_1\to\ldots$, 路径上存在某个状态 s_i, $(M,s_i)\vDash\phi$.

(8) $(M,s)\vDash AG\phi$ 当且仅当 s 延伸的所有路径 $s\to s_1\to\ldots$, 路径上的所有状态 s_i, $(M,s_i)\vDash\phi$.

(9) $(M,s)\vDash A[\phi_1U\phi_2]$ 当且仅当 s 延伸的所有路径 $s\to s_1\to\ldots$, 路径上存在某个状态 s_i, $(M,s_i)\vDash\phi_2$, 同时对于所有 $j<i$ 的状态 s_j, $(M,s_j)\vDash\phi_1$.

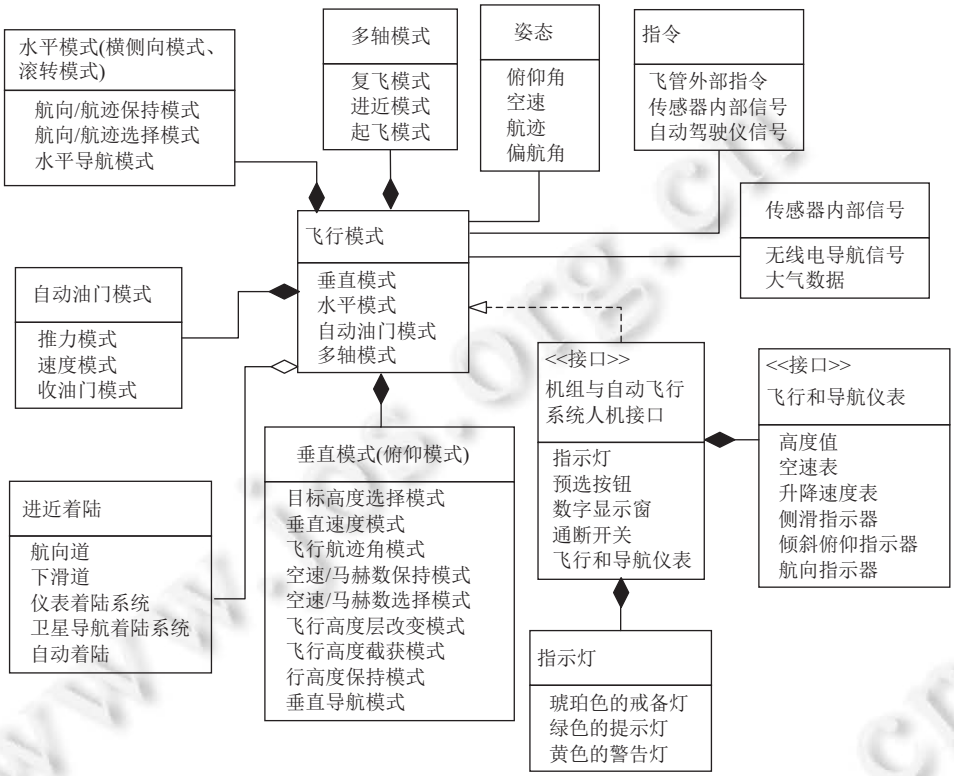

图 1 自动飞行系统模式概念图

模型检验以系统模型 M 和系统属性 ϕ 为对象, 检验 $(M,s_0)\vDash\phi$ 是否成立, 其中 s_0 是 M 的初始状态. 因此其是一种能够对有限状态系统进行形式化建模和自动验证的技术, 能够通过隐式不动点计算或显示状态搜索来验证系统是否满足规约性质, 如图 2 所示, 模型检验方法通过形式化建模语言构建系统模型, 通过时序逻辑公式描述系统性质规约, 最终使用模型检测工具自动验证系统模型是否满足性质规约, 若不满足, 则给出反例路径. 本质上是针对给定的模型和待验证的性质, 通过状态搜索自动判别模型在语义层次是否满足该性质. 因为自动化程度较高, 因此模型检验和时态逻辑在航空航天、轨道交通、核能等自动化领域中具有广泛的应用, 线性时态逻辑 (linear temporal logic, LTL)、计算树逻辑以及它们的扩展逻辑经常用于某些需要考虑任务的时态、时间或空间约束的场景中[28,29].

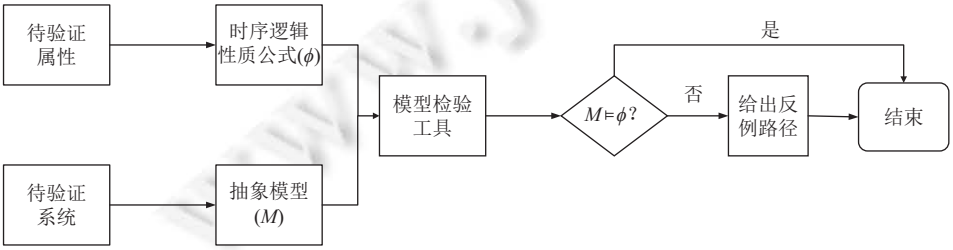

图 2 模型检验过程

2 自动飞行系统模式转换的需求建模

2.1 自动飞行系统模式转换需求的形式化模型定义

SysML 作为一种通用图形建模语言, 具有成熟的工具支持和扩展能力, 在航空工程中得到越来越多的应用. 然而其缺乏严格的形式化语法语义的定义, 属于半形式化语言, 且其通用性使得面向领域特定问题的建模不易使用. 本文提出一种对自动飞行系统模式转换需求的领域建模语言 MTRDL (mode transition requirement discription language), 该语言具有明确的形式化定义可以作为安全性验证的基础.

定义 1. 系统是一个元组 System = (MODULAR, MODE, RELATIONSHIP, PROCESS, COMM) , 其中,

(1) MODE 表示系统模式 mode 的集合.

(2) MODULAR 表示模块 modular 的集合.

(3) RELATIONSHIP 表示关系 relationship 的集合.

(4) PROCESS 表示模块的内部状态迁移过程 process 的集合.

(5) COMM 表示模块或模块间的通信 comm 的集合.

在自动飞行模式转换过程中, 模式的接通和切出需要和大量系统中存在的模块进行交互, 如图 3 所示, 部分模块只控制模式的转换, 如飞行控制面板 FMCP 等, 部分模块则与模式交互密切相互影响, 如自动驾驶仪 AP、飞行指引 FD 和自动油门 AT, 部分模块则只接收模式的反馈信息, 不影响模式的转换, 如飞行显示面板 PFD. 对系统的建模一方面需要对模式和模块进行建模, 包括其静态结构和内部的状态迁移过程, 另一方面需要对模式和模块间的交互进行建模.

定义 2. 一个模块是一个元组 modular = $(n_u, \Sigma_{im}, E_{im}, \Sigma_{ex}, E_{ex}, \Sigma_{en}, E_{en})$, 其中,

(1) n_u 表示模块的名称, 我们用 N_u 表示所有模块名称的集合.

(2) Σ_{im} 表示模块输入变量的集合.

(3) E_{im} 表示模块输入事件的集合.

(4) Σ_{ex} 表示模块输出变量的集合.

(5) E_{ex} 表示模块输出事件的集合.

(6) Σ_{en} 表示模块内部封装变量的集合.

(7) E_{en} 表示模块内部封装事件的集合.

modular 表示了系统中能够执行功能的部件, 为了体现模块与其他模块或模式的交互, modular 中的变量和事件被分为输入、内部封装、输出这 3 类: 输入变量和事件可以认为是一种中间变量, 其取值取决于其他模块或模式的输出变量或事件; 内部封装变量和事件定义了模块的状态, 它们的取值取决于输入变量和事件以及其他内部封装变量和事件; 输出变量和事件定义了到其他模式或模块的输出, 其取值取决于内部封装变量和事件. 只有输出变量和事件能够被其他模块或模式访问, 而输入和内部封装的变量和事件仅在模块内部处理过程中需要使用. 这种分类的好处在于能够层次化的表示模块的需求, 将模块的内部信息和可被其他模块或模式访问的信息区分开, 同时利用这种层次化的表示降低 process 的规模, 使自动飞行系统这种复杂系统更易建模, 在条目化需求更改时也更容易对模型进行更改. 以自动驾驶仪 AP 为例, Is_AP_Minconfig _Cond_Met 是一个输入变量, 表示飞机是否满足 AP 接通最小构型状态, 当它为真时需要满足 2 升降舵均有效、至少 1 个发动机有效等多个条件, 即它的取值取决于多个其他模块或模式的输出; 同时 AP_Engaged_State 是内部封装变量, 表示 AP Engage 状态, 当它为真时需要满足 AP Available 状态为 TRUE 且 FMCP 的 AP 接通请求信号为 TRUE, 即它的取值取决于内部变量 AP_Available_State 和一个输入变量 FMCP_Engage_AP_Signal; 输出变量 Is_AP_Engaged 表示 AP 是否接通, 是 AP 模块给到其他模块或模式的输出, 它的取值取决于内部封装变量 AP_Engaged_State, 和 AP 模块有通信的其他模块或模式可以访问该变量.

定义 3. 一个模式是一个元组 mode = $(n_d, \Sigma_{entry}, E_{entry}, \Sigma_{exit}, E_{exit}, \Sigma_{mode}, E_{func}, E_{em}, \Sigma_{out}, E_{out})$, 其中,

(1) n_d 表示模式的名称, 我们用 N_d 表示所有模式名称的集合.

(2) Σ_{entry} 表示模式接通逻辑的输入变量集合.

(3) E_{entry} 表示模式接通逻辑的输入事件集合.

(4) Σ_{exit} 表示切出逻辑的输入变量集合.

(5) E_{exit} 表示模式切出逻辑的输入事件集合

(6) Σ_{mode} 表示模式的状态变量集合.

(7) Σ_{func} 表示模式的功能变量集合.

(8) E_{em} 表示模式的内部事件的集合

(9) Σ_{out} 表示模式的输出变量的集合.

(10) E_{out} 表示模式的输出事件的集合.

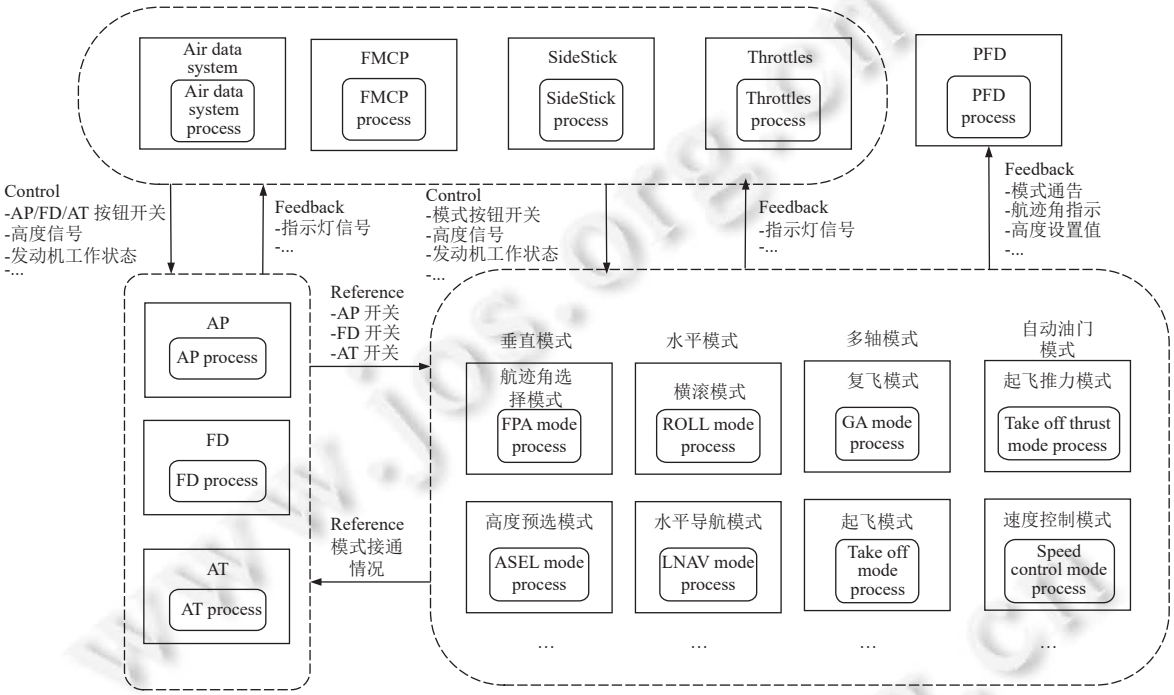

图 3 系统结构示意

mode 表达了飞行模式的结构, 在 modular 中对变量和事件的分类基础上, mode 的输入被进一步分为接通逻辑和切出逻辑, 内部封装变量被分为模式状态变量和模式功能变量. 以高度预选 ASEL 模式为例, 需求定义了何时应自动预位 ASEL 模式, 何时判定满足高度捕获条件, 因此用事件 ARM_ASEL 和变量 Is_ASEL_Capture_Cond_Met 表示这两个接通条件, 它们的取值影响了模式状态变量 ASEL 的取值. 模式状态变量 ASEL 有 cleard, armed, capture, track 这 4 个取值, 表示了高度预选模式未激活、预位、目标捕获、目标跟踪这 4 种状态. 此外, 由于不同模式下系统需要执行的功能不同, 因此也要定义功能变量来描述两类功能行为, 一类是对该模式相关参数的调整, 另一类是激活该模式下的飞行控制律. 以航迹角选择 FPA 模式为例, 在 FPA 模式下就需要根据情况设定不同的飞行航迹角, 以及激活 FPA 模式下的控制律, 因此 FPA 模式的功能变量需要定义整数型变量 Target_FPA 和布尔型变量 Is_FPA_Law_Active.

定义 4. 一个关系是一个元组 relationship = $(type_r, rel)$, 其中,

(1) $type_r \in Type_{dr} \cup Type_{ur}$ 表示关系的类型, 其中 $Type_{dr}$ 表示模式间关系的类型集合, $Type_{ur}$ 表示模式和模块或模块间关系的类型集合.

(2) $rel \in (N_u \cup N_d) \times (N_u \cup N_d)$ 表示模式或模块间的关系.

关系表示了模式间关系和模块间关系, 模式间关系在后续工作中分为隶属, 互斥, 兼容和可转换这 4 类关系, 这

些关系用于体现在整个模式转换过程中的模式总体关系. 模式和模块或模块间关系则包含了常见的组合、泛化等关系.

定义 5. 一个过程是一个元组 process = $\left(n_p, g_p, type_p, \Sigma_{\text{in}}, E_{\text{in}}, \sigma_p, e_p, L, G, tr, Ef, T\right)$, 其中,

(1) n_p 表示过程的名称, 我们用 N_p 表示所有过程名称的集合.

(2) $g_p \in N_u \cup N_d$ 表示过程从属的模块或模式, 如果 $g_p \in N_u$ 称过程 n_p 从属于模块 g_p, 如果 $g_p \in N_d$ 则称过程 n_p 从属于模式 g_p.

(3) $type_p \in$ {Imported, Encapsulated, Exported} 表示过程的类型.

(4) Σ_{in} 表示过程的输入变量集合.

(5) E_{in} 示过程的输入事件集合.

(6) σ_p 表示过程的过程变量, 我们用 Σ_p 表示所有过程变量的集合.

(7) e_p 表示过程的过程事件, 我们用 E_p 表示所有过程事件的集合.

(8) L 表示过程的状态.

(9) $G \subseteq 2^{\Sigma_{\text{in}}} \times \phi(\Sigma_{\text{in}})$ 表示过程发生的卫士条件, 其中 $\phi(\Sigma_{\text{in}})$ 是 Σ_{in} 上的约束条件.

(10) $tr \subseteq 2^{E_{\text{in}}} \times \phi(E_{\text{in}})$ 表示过程发生的触发条件.

(11) $T \subseteq L \times G \times tr \times f(\sigma_p, e_p) \times L$ 表示状态的迁移过程, 其中 $f(\sigma_p, e_p)$ 是赋值函数.

process 定义了模块或模式的状态迁移过程, 一个模块或模块具有多个过程. 具体来说, 过程定义了模块或模式中变量和事件的取值情况, 过程类型则表示了这个过程描述的过程变量或事件类型. 如过程变量为 AP_Engaged_State 的过程, 由于 AP_Engaged_State 是内部封装变量, 则该过程类型是 Encapsulated, 决定 AP_Engaged_State 取值的两个变量 AP_Available_State 和 FMCP_Engage_AP_Signal 则是该过程的输入变量. 该过程具有 AP_Engaged 和 AP_Disengaged 两个状态, 从 AP_Disengaged 状态迁移到 AP_Engaged 状态的过程中, 需要满足相应的卫士条件, 并赋值 AP_Engaged_State 为 TRUE.

定义 6. 一个通信是一个元组 comm = $(n_c, g_s, g_r, type_c, \Sigma_c, E_c)$, 其中,

(1) n_c 表示通信的名称, 我们用 N_c 表示所有过程名称的集合.

(2) $g_s \in N_u \cup N_d$ 表示通信的发起方.

(3) $g_r \in N_u \cup N_d$ 表示通信的接受方.

(4) $type_c \in Type_c$ 表示通信的类型, 其中 $Type_c$ 表示所有模式名称的集合.

(5) Σ_c 表示通信变量的集合.

(6) E_c 表示通信事件的集合.

comm 定义了模块或模式间的通信, 包括通信双方、通信方向、通信类型和通信变量. 在本文工作中, 通信变量即是通信来源方的输出变量或事件, 通过 comm 我们建立起系统中模块或模式间的数据交互. 值得一提的是, 虽然通信过程的干扰和延迟在系统安全分析中也极为重要, 但由于条目化需求中未描述这些因素, 且加入这些因素会极大增加系统的复杂度, 因此这些因素未作描述.

MTRDL 作为系统形式化建模的基础, 以它为元模型的 SysML profile 和传统的 SysML 相比, 严格定义了语法语义, 扩展了领域中模式转换相关的概念, 并建立模型和数据字典间的关系, 如图 4 所示. 一方面, 在 SysML profile 中按照 MTRDL 范式定义了模型元素, 另一方面通过数据字典和模型元素的对应关系构建了在自动飞行模式转换领域使用的领域概念库, 从而实现了支持形式化表示的领域化 SysML 建模方法.

在以 MTRDL 为基础的模型结构中, 我们采用数据字典对需求中的变量、事件等基本元素进行定义, 数据字典被定义为:

$$DataDict = \{N_d, N_u, N_p, N_c, Type_{dr}, Type_{ur}, Type_c, \Sigma, E, \Gamma, V, M\},$$

其中, $\Sigma = \Sigma_{\text{im}} \cup \Sigma_{\text{ex}} \cup \Sigma_{\text{en}} \cup \Sigma_{\text{entry}} \cup \Sigma_{\text{exit}} \cup \Sigma_{\text{mode}} \cup \Sigma_{\text{func}} \cup \Sigma_{\text{out}} \cup \Sigma_{\text{in}} \cup \Sigma_p \cup \Sigma_c$ 表示 MTRDL 所有使用到的变量的集合, $E = E_{\text{im}} \cup E_{\text{ex}} \cup E_{\text{en}} \cup E_{\text{entry}} \cup E_{\text{exit}} \cup E_{\text{em}} \cup E_{\text{out}} \cup E_{\text{in}} \cup E_p \cup E_c$ 表示 MTRDL 所有使用到的事件的集合, Γ 表示变量和事件的类型集合如 Boolean, interger 等, V 表示变量和事件的取值范围, $M: \Sigma \cup E \to \Gamma \times V$ 表示变量和事件到其类型与取值的映射.

图 4 以 MTRDL 为基础的模型结构

在数据字典中, 不仅描述了自动飞行领域中需要描述的概念术语, 包括各种模式、模块、过程、通信的名称, 还描述了 MTRDL 模型中需要使用的具体变量和事件, 包括它们的取值类型和取值范围. 因此我们将数据字典分为常量字典、变量字典和类型字典和事件字典. 在常量字典中, 定义的每一个常量必须有效, 即它存在值, 以及值的类型符合该常量的类型; 在变量字典中, 需要定义变量的类型, 以及变量的数据类型、初始值、取值范围. 类型字典定义了所有用到的类型, 包括基础数据类型和自定义类型. 事件字典中和变量字典类似, 但由于事件表示的是瞬态发生的行为, 因此数据类型固定为布尔类型. 字典中所有的项有对应的自然语言注释说明其含义.

2.2 基于 MTRDL 的 SysML profile

SysML 由统一建模语言 (unified modeling language, UML) 子集的基础上扩展而来, 是一种用于系统工程应用的通用系统架构建模语言, 支持各种系统包括硬件、软件、信息、流程、人员和设施的规范、分析、设计、验证和确认. 和 UML 不同, 由于 SysML 用于系统建模, 系统需求包含系统的功能、接口、物理和性能特性, 和其他质量特性, 因此 SysML 所有视图都可以用于需求阶段的需求建模. 由于 SysML 在需求建模上的适用性和其本身的可扩展性, 本文构建了基于 MTRDL 的 FMSysML profile, 主要对 SysML 现有的 Attribute、Block、Statemachine 等进行了扩展, 使得更符合飞行模式转换的领域特征. Profile 配置文件构造如图 5 所示.

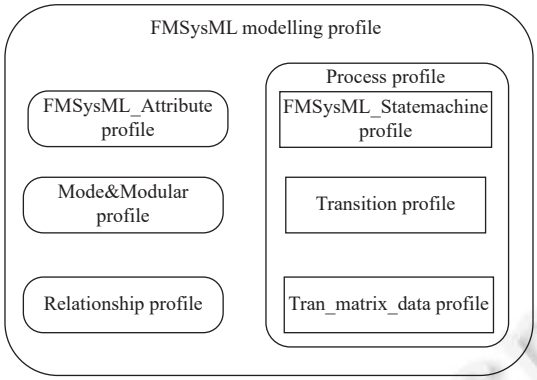

图 5 FMSysML profile 构造

如图 6 所示, 在 Relationship profile 中通过对《Dependency》进行扩展定义了两类关系: 模式间关系和通信关系. 模式间关系又分为隶属 (Submode of), 互斥 (Exclude), 兼容 (Compatible) 和可转换 (Transitable). 隶属子模式表示一个模式是另一个模式的子模式, 互斥表示当一个模式处于接通状态时另一个模式不能接通, 兼容表示在一个模式处于接通状态时另一个模式可以接通, 可转换表示一个模式可以过渡切换到另一个模式. 通信关系分为控制 (Control), 反馈到 (Feedback to), 参照 (Refer to) 和感知 (Sense), 表示了模块间以及模式和模块间的通信关系, 其中控制表示一个模块/模式可以控制另一个模块/模式的行为, 反馈到表示一个模块/模式能够给予另一个模块/模式它某些行为的反馈信息, 参照表示一个模块/模式可以参考另一个模块/模式的某些信息从而执行自身的一些行为, 感知表示一个模块/模式 (通常是传感器) 可以感知另一个模块/模式的某些信息.

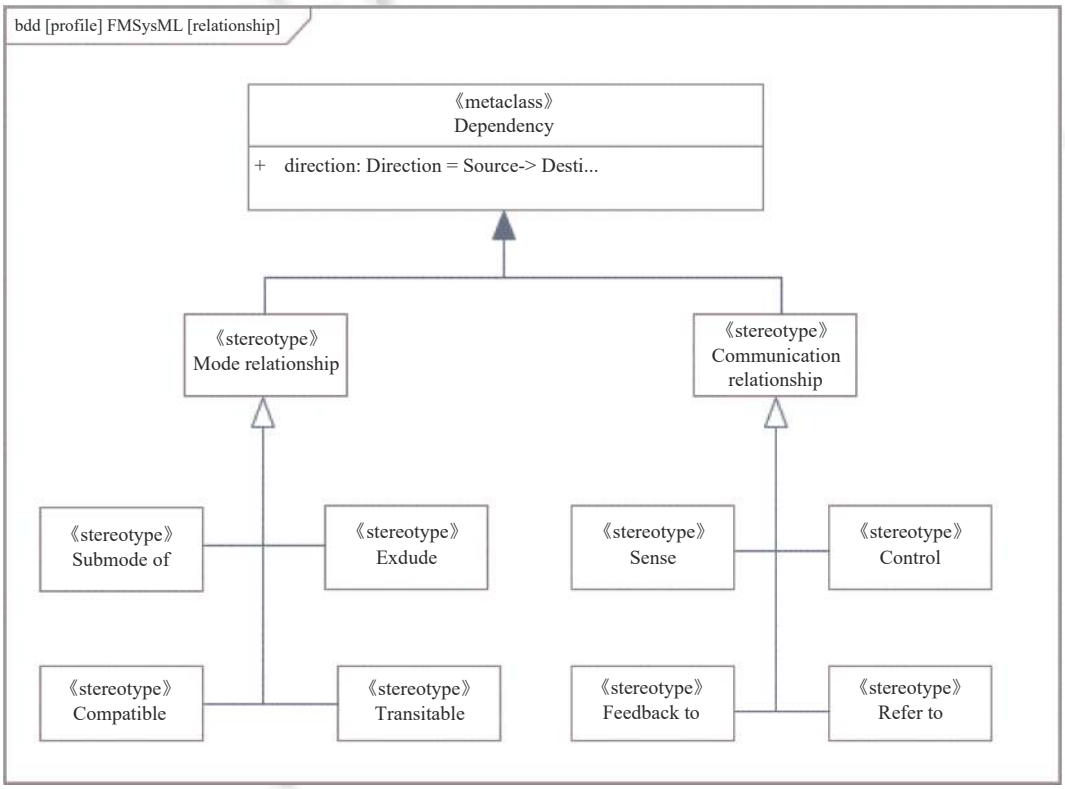

图 6 Relationship profile 的定义

如图 7 所示, 在 Mode&Modular profile 中构造型《Mode》和构造型《Modular》由《Block》扩展而来, Mode 中包含了模式维度, Modular 又分为功能模块和传感器模块. 在 Attribute profile 中由《Attribute》扩展了《Imported attribute》《Encapsulated attribute》和《Exported attribute》, 并在这 3 个构造型的基础上又扩展了模式相关的属性构造型.

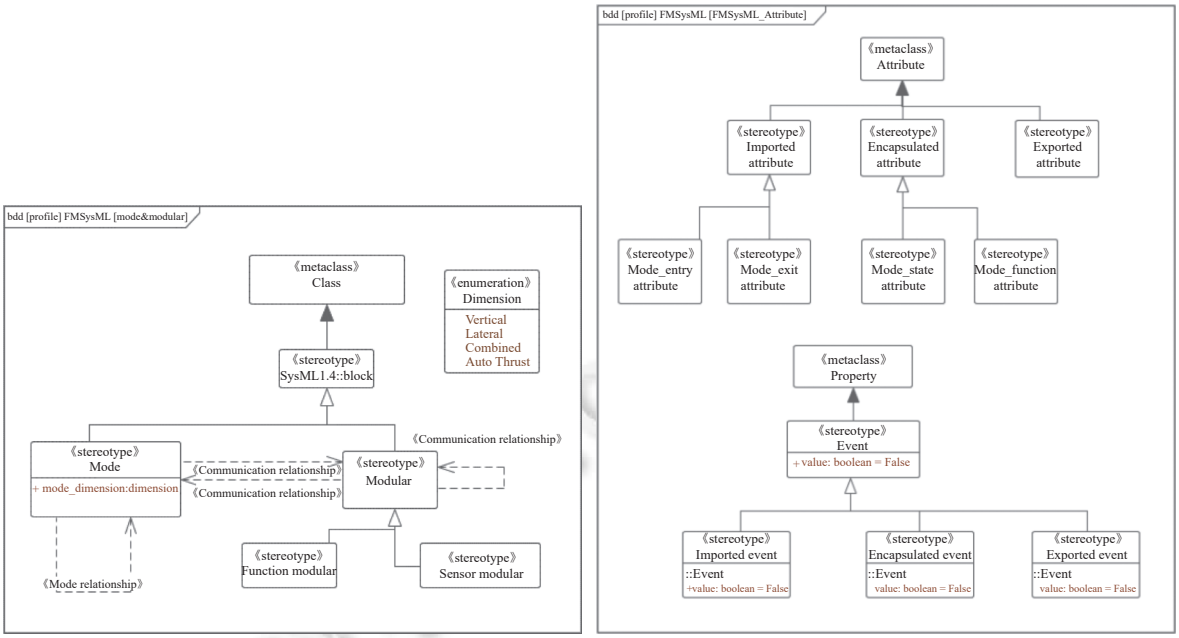

图 7 模式、模块和属性的 profile 定义

接下来, 在 StateMachine profile 中, 构造型《Process》由《StateMachine》扩展而来, 并根据状态机中关联变量或事件的不同发展出《Imported process》《Encapsulated process》和《Exported process》这 3 种构造型, 如图 8 所示. 《Imported process》和《Exported process》中具有 input variable、input event 两种关联列表类型的 tagged value, 与 process variable、process event 两种关联类型的 tagged value. 《Encapsulated process》具有 input im variable、input en variable、input im event、input en event 这 4 种关联列表类型的 tagged value, 与 process variable、process event 两种关联类型的 tagged value. 通过定义这些 tagged value, 可以使 process 与 mode 或 modular 中定义的变量和事件产生关联, 同时保证了层次化特征, 即: 输入变量或事件只由其他部分的输出变量或事件决定; 内部封装变量或事件只由输入变量或事件以及其他内部封装变量或事件决定; 输出变量或事件只由内部封装变量或事件决定.

为了解决 SysML 状态机中 Transition 没有严格的语法语义的问题, 结合工程人员通常用矩阵表达状态转换的工程习惯, 如图 9 所示, 我们首先通过对《Block》扩展得到构造型《Trans_matrix_data》, 然后对《Transition》扩展得到《Trans》, 在《Trans》中定义了 tagged value 对《Trans_matrix_data》进行了关联.《Trans_matrix_data》中定义了最多 15 个数组类型的属性 row, 从而能够表示转换矩阵. 以前文提到从 AP_Disengaged 状态到 AP_Engaged 状态的转换为例, 该转换需要满足 AP Available 状态为 TRUE 且 FMCP 的 AP 接通请求信号为 TRUE, 转换矩阵如表 1 所示.《Trans_matrix_data》中属性 row1 的取值对应了表 1 中条件 1 该行的变量取值, 为 {1, 1}. 通过对各种数据类型的变量取值编码,《Trans_matrix_data》可以表示任意的状态转移条件. 在《Trans》中不仅关联了转换矩阵, 还通过 condition、guard、trigger 这 3 个属性描述了转换的条件数、转换从属的《Process》中的输入变量数和输入事件数, 便于检查是否关联了错误的 Trans_matrix_data.

最后, 为了便于表示模型中《Mode》《Modular》间的交互, FlowProperty Profile 在《FlowProperty》上扩展

了与《Exported attibute》《Exported event》关联的 tagged vaule, 如图 10 所示. 此外, 由于自动飞行模式转换中某些事件不能同时发生, 如同时在 FMCP 上按下两个不同的按钮可能会导致危险, 因此模式转换需求需要考虑事件的优先级顺序, 虽然在 Process 中可以通过状态转换条件表示优先级, 但是为了便于表示, 我们仍设计了 Priority profile 支持对优先级的表达, 如图 11 所示. 在 Priority profile 中, 通过定义了一个向量类型的变量 priority_vector 来表示优先级, 优先级大小通过数字表示, 0 表示最大的优先级, 数字越大优先级越低.

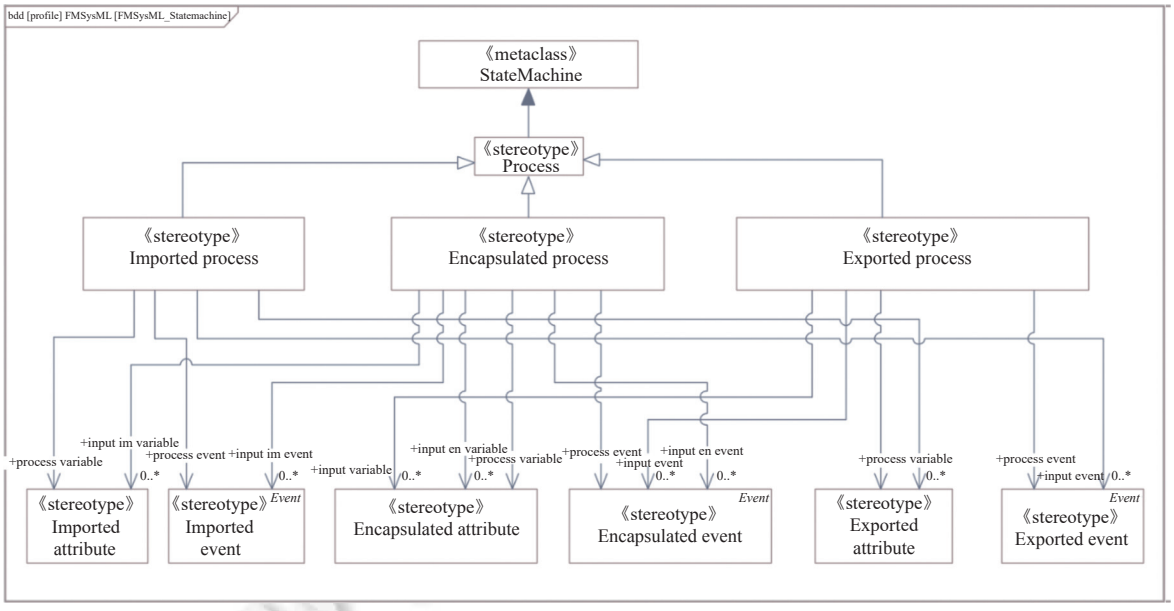

图 8 FMSysML_Statemachine profile 的定义

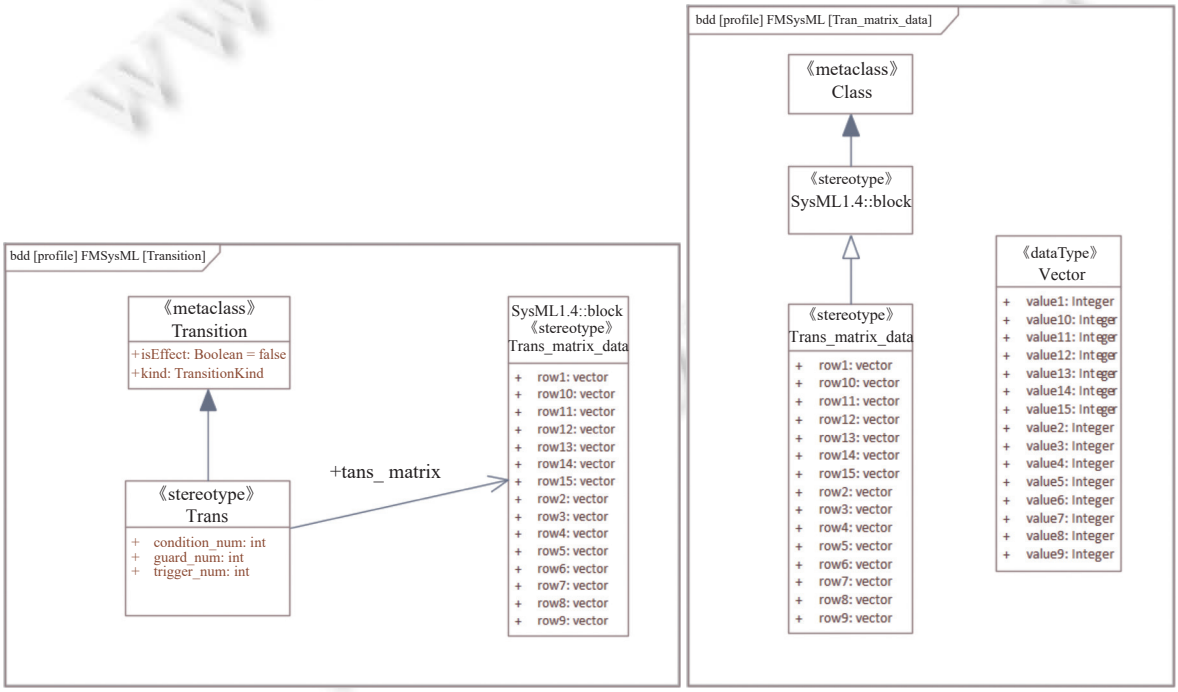

图 9 Transition 和 Tran_matrix_data profile 的定义

表 1 Trans_matrix_data 表示的矩阵示例

变量	条件1
AP_Available_State	TRUE
FMCP_Engage_AP_Signal	TRUE

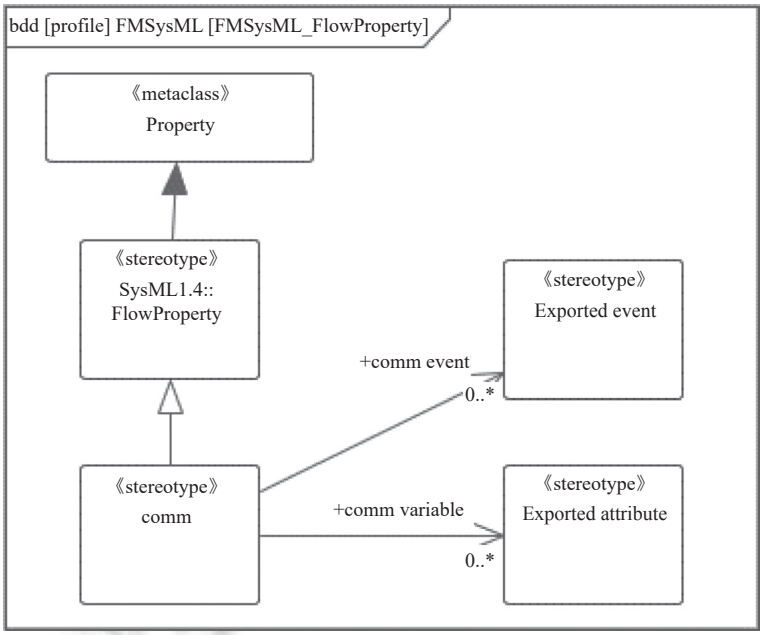

图 10 FlowProperty profile 的定义

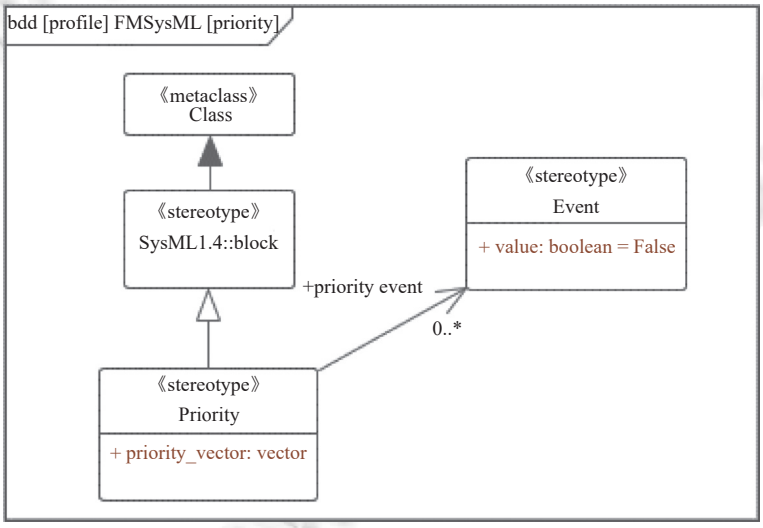

图 11 Priority profile 的定义

3 安全性质的辅助规约

对系统需求进行形式化分析验证时, 通常将功能需求和安全需求分开阐述, 功能需求阐述系统需要实现的功能, 而安全需求阐述了系统在某些功能实现时或状态下需要满足的性质和条件. 在实际项目中, 安全需求通常没有

固定的规范, 往往由自然语言撰写, 并且获取安全需求前需要的安全分析过程极耗费时间和人力. 除此之外, 在规约形式化描述的安全性质时, 通常先由领域专家使用自然语言描述安全需求, 再由形式化专家来手动转化, 这种方式存在一些问题, 如形式化专家需要对领域知识有一定的理解否则容易转化错误且转化错误的情况下领域专家也难以分辨, 更有转化效率低下、双方交流过程中容易出现差错等情况. 因此本文提出一种结合 FMSysML 特征来降低安全性质规约难度的方法.

自动飞行系统模式转换的安全需求本质上描述了飞行模式和相关控制模块是否兼容, 即在上下文变量和上下文事件制约的前置条件下, 一方面在静态角度飞行模式和相关控制模块的当前状态必须处于允许的范围内, 另一方面在动态角度飞行模式和相关控制模块的状态变迁必须处于允许的范围内. 以真实工程中的安全需求 R1“If the autopilot is engaged, disengaging the autopilot shall not turn off the FD”和 R2“Only one lateral mode shall ever be active at any time”为例, 前者描述了在前置条件为“If the autopilot is engaged, disengaging the autopilot”的情况下 FD 必须保持 FD_On 的状态, 即对 FD 的状态变迁做出限制, 后者描述了在前置条件为“at any time”的情况下所有水平模式的子模式共同满足有且只有一个子模式激活, 即对当前所有水平模式的子模式状态组合做出限制.

基于自动飞行系统模式转换需求的特性和安全需求的描述规范, 我们总结出易于结合 FMSysML 生成的几类安全需求.

(1) 全局性的模式约束: 指任意时刻系统没有违反模式间的互斥、兼容、可转换关系. 如“任意时刻只能有一个水平模式处于接通状态”.

(2) 局部的模式约束: 指在某些条件下, 应满足某些模式间的互斥、兼容、可转换关系. 如“当滚转角大于 6 度时, 不应该出现 GA 模式和航迹角保持 Track hold 同时接通的情况”.

(3) 两个模块或模式间控制和反馈的约束: 这类安全需求要求在某些条件, 发出控制的模块能接受到相对应的反馈, 通常是指示灯信号的反馈. 如按下 FMCP 上的 AP 按钮后 AP 指示灯会亮.

(4) 控制和到 PFD 反馈的约束: 这类安全需求关注 PFD 的显示情况, 要求模块发出的控制能正确影响 PFD 的显示, 如在 FPA 模式接通时转动 HDG/TRK 旋钮时应在 PFD 上显示选择的航向/航迹角.

(5) 两个模块或模式间参照的约束: 这类安全需求定义了在某些条件下, 两个互相参照的模式或模块间的参照具有的约束. 如在拉平模式 FLARE 接通时, 如果 AP 断开则拉平模式 FLARE 应转换到待机模式 STDBY.

值得注意的是, 上述的几类安全需求并不能全部涵盖完整的安全需求范围, 该总结的用处在于可以结合需求模板和 FMSysML 模型自动化生成安全性质. 需求模板是常用的一种缩小自然语言和形式化方法间鸿沟的方法, 需求模板要求用户按照规定好的语法模板和关键字进行需求描述, 通常包含定义不同的条件语法结构, 条件中的条件规则结构, 满足条件后进行的操作和回应动作的结构等. 本文采用需求模板来描述安全需求的优点在于自动飞行系统模式转换的安全需求描述比较简单, 没有事件时间属性的描述, 也没有复杂的系统回应动作. 安全需求模板如下所示.

SafeReq::= <Actor> 'shall/shall not satisfy' <Specification>+'always'|'if'<Con>|Null

Actor::= < Mode>|<Modular>|<CardinalNum >< Mode >|<CardinalNum><Modular >|<Actor>< Logic operator > <Actor>

Specification::= < Specification 1>|< Specification 2>|< Specification3>|< Specification 4>|< Specification5>|< Specification >< Logic operator >< Specification >

Specification 1::=< Mode Relationship>

Specification 2::=< Comm_Control_Con>imply< Comm_Feedback_Con >

Specification 3::=< Comm_Reference_Con>imply< Comm_ Reference _Con >

Con::=<Variable Con>|<Event Con>|<Con>< Logic operator ><Con>

Variable Con::= <Mode State Variable>'is'<Value>|<Mode Func Variable>'is'<Value>|< Modular Encapsulated Variable>'is'<Value>|< Variable Con >< Logic operator >< Variable Con >

Event Con::= <Mode Encapsulated Event >'is'<Value>|< Modular Encapsulated Event>'is'<Value>
Comm_Control_Con::=< Comm_Control_Variable>'is'<Value>|< Comm_Control_Event>'is'<Value>| < Comm_Control_Con >< Logic operator >< Comm_Control_Con >
Comm_Feedback_Con::=< Comm_ Feedback _Variable>'is'<Value>|< Comm_ Feedback _Event>'is'<Value>| < Comm_ Feedback_Con >< Logic operator >< Comm__Feedback _Con >
Comm_ Reference _Con::=< Comm_ Reference _Variable>'is'<Value>|< Comm_ Reference _Event>'is'<Value>| < Comm_ Reference _Con >< Logic operator >< Comm_ Reference _Con >
CardinalNum::= 'at least'<number>| 'only' <number>|…
Logic operator::= 'And'|'Or'

在该模板中, 对安全需求描述需要遵守的语义结构进行了迭代式的定义, 根据该需求模板, 需求的参与者 (Actor) 是飞行模式或者控制模块, 需求的规约 (Specification) 描述了前文中几类安全需求的不同约束, 需求的条件 (Con) 描述了安全需求的前提条件, 在此处只和 FMSysML 模型中的内部封装变量和事件关联.

通过安全需求模板, 安全需求的形式已经接近于形式化的描述, 随后对安全需求进行相关信息的提取 (如: 变量定义、条件、事件、模式等), 这些相关信息组合成为一张安全需求信息表. 这么做的好处首先是符合机载软件工程领域中经常使用各种表格形式的习惯, 也便于领域专家对安全需求的进一步审查, 除此之外, 上述信息提取后可以通过数据字典中的注释还原为自然语言需求, 从而便于理解需求含义和查找出安全需求模板可能出现的错误. 同时, 也需要定义多条规则支持对安全需求描述一致性的自动审查, 包括:

(1) 当且仅当条件类型是 always 时, 条件语句为空.

(2) 参与者包含基数词时, 基数词后的飞行模式或模块不允许是原子级.

(3) 参与者包含基数词时, 不允许基数词后的飞行模式或模块重复出现, 如“only one Lateralmode and only one Verticalmode”是允许的, 但“only one Lateralmode and at least one Lateralmode”是不被允许的.

(4) 参与者包含基数词时, 规约语句中的使用 This 进行指代, 如“only one Lateralmode”对应到规约语句用“This Lateralmode”来进行指代. 不包含时则不允许这样使用.

由于安全需求模板与 FMSysML 存在映射关系, 且本文的安全需求模板本质上是一种将形式化的安全性质语义模板化的产物, 因此可通过算法自动将安全需求转换为相应的安全性质, 具体的生成算法由于篇幅原因在此不作描述.

4 实例研究

自动飞行系统模式与航空电子设备其他部件之间接口的简化视图如后文图 12 所示, 显示了在本文研究中使用的子系统, 由于系统复杂度问题我们暂时忽略了自动飞行系统模式与飞行管理系统的交互. 在自动飞行模式激活时, 模式转换控制模块会对飞行指引仪和自动驾驶仪提供指引, 同时提供模式通告给主飞行显示器.

本节给出一个自动飞行系统中飞行模式转换需求实例, 包括了从自然语言的条目化需求建模到安全性质生成以及后续形式化验证的过程. 内容主要有: 条目化需求预处理, 模型的形式化分析及结果等.

4.1 飞行模式转换的条目化需求处理

预处理的目的是建立一个领域概念库, 包括自动飞行系统模式转换中常见的各种模式、模块、输入/输出变量、数据类型、中间变量等的定义, 本文中 MTRDL 模型的数组字典即可视为一种领域概念库. 领域概念库可以支持在多种机型的需求开发与验证的复用, 是需求建模和安全性质获取的基础. 表 2 中给出了一部分自动飞行系统模式转换的条目化需求.

由表 2 中的需求, 需求条目中的信息可以被提取并构造领域概念库, 这个过程需要建模人员需要与工程人员共同讨论并确定各条需求具体要表达的语义: 首先, 需求中的飞行模式被提取出来, 如表 2 中的水平、垂直、风切

变导引模式等概念, 这些概念间的层次关系也由需求所描述; 其次, 需求中的模块概念被提取, 如 AP、FD 等, 这些模块对应了实际系统中的软硬件实体, 构成整个系统; 最后, 提取需求中的常量、变量和事件, 包括这些它们的名称、类型、类别、初始值和语义注释. 通过处理, 我们能得到常量字典、变量字典、数据类型字典和事件字典, 如后文表 3–表 5 所示.

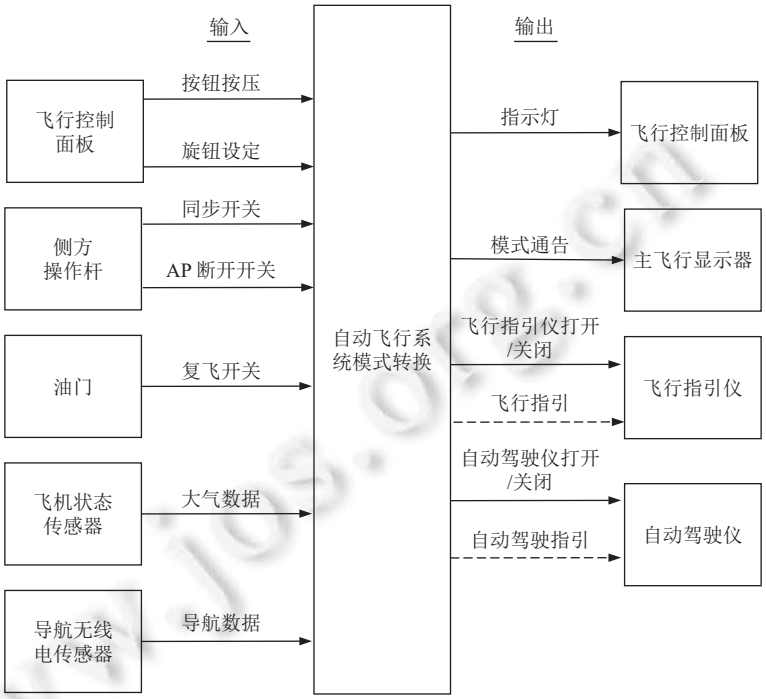

图 12 自动飞行系统模式转换接口简图

表 2 条目化需求示例

需求编号	需求内容
1	当FMCP数据无效时, 主飞控应退出正常模式
2	FD应自动接通, 当 (1) 主飞控在正常模式, 且 (2) 任意水平和垂直模式接通
3	AP应接通, 当 (1) FMCP上的AP按钮被按压, (2) AP接通条件被满足
4	AP按钮应点亮, 当AP处于接通时
5	AP接通条件被满足, 当 (1) 满足AP接通最小构型, 且主飞控处于正常状态, (2) 飞机起飞后在空中400英尺以上, 或飞机在地面上且两个发动机均关闭, (3) FD处于接通, (4) FMCP数据有效
6	AP应正常断开, 当侧杆上的AP断开开关被按压时
7	AP应非正常断开, 当 (1) AP已处于接通状态, (2) 主轮触地5 s后, 除了复飞模式激活时
8	AP应非正常断开, 当 (1) AP已处于接通状态, (2) 按下FMCP上的AP按钮, 除了高度在200 ft以下, 且自动着陆模式已接通
9	AP应非正常断开, 当 (1) AP已处于接通状态, (2) 飞行员通过在侧杆施加足够的力超控AP, 使脚蹬从中立位置移开
10	AP应非正常断开, 当 (1) AP已处于接通状态, (2) 飞行员通过在侧杆施加足够的力超控AP, 使侧杆从中立位置移开
11	AP应非正常断开, 当 (1) AP已处于接通状态, (2) 风切变导引(WS)模式接通

4.2 基于 MTRDL 的 SysML 建模

由表 2 中的需求构造数据字典后, 根据 MTRDL 的模型语法和语义首先将领域概念对应到 FMSysML 模型, 如变量字典中存在 Is_WSMode_ Selected 变量, 类型为输出变量, 表示 WS 模式是否接通, 因此存在 WS 模式属于

飞行模式的一种, 需要在 FMSysML 中作为 Mode 元素进行建模. 在 FMSysML 模型中将 MTRDL 模型的静态机构元素定义完整后, 再添加飞行模式间的转换关系和模块功能状态间的转换关系, 与模式模块间的交互关系. 以表 2 中需求为例, 使用 FMSysML 建模得到的模型如图 13 所示.

表 3 MTRDL 模型的变量字典示例

变量名	类型	类别	初始值
Is_FMCPData_Valid	Boolean	Exported	True
Is_MainFC_Abnormal_Cond_Met	Boolean	Imported	False
MainFC_Status	Modularstate_Type2	Encapsulated	Normal
Is_MainFC_Normal	Boolean	Exported	True
Is_FD_Engage_Cond_Met	Boolean	Imported	False
FD_State	Modularstate_Type1	Encapsulated	Disengaged
Is_FD_Engaged	Boolean	Exported	False
Is_LateralMode_Selected	Boolean	Imported	False
Is_VerticalMode_ Selected	Boolean	Imported	False
Is_WSMode_ Selected	Boolean	Exported	False
Is_AutolandMode_ Selected	Boolean	Exported	False
Is_GAMode__Selected	Boolean	Exported	False
Is_AP_Engaged_Cond_Met	Boolean	Imported	False
Is_AP_Minconfig_Cond_Met	Boolean	Imported	False
AP_State	Modularstate_Type1	Encapsulated	Disengaged
Is_AP_Engaged	Boolean	Exported	False
Is_APlamp_On	Boolean	Encapsulated	False
MainFC_Status	Modularstate_Type2	Encapsulated	Normal
Is_MainFC_Normal	Boolean	Exported	True
Altitude	Altitude	Encapsulated	0
Is_Altitude_ exceed_400	Boolean	Exported	False
MainwheelTouchdownTime	Time	Encapsulated	0

表 4 MTRDL 模型的事件字典示例

事件名	类型	类别	初始值
FMCP_APSwitch_Pressed	Boolean	Exported	False
When_Illuminate_APlamp	Boolean	Imported	False
Sidelever_APSwitch_Pressed	Boolean	Exported	False
Sidelever_Override_AP	Boolean	Exported	False
When_Engage_AP	Boolean	Imported	False
When_Normal_Disengage_AP	Boolean	Imported	False
When_Abnormal_Disengage_AP	Boolean	Imported	False
MainwheelTouchdownTime_ exceed_5	Boolean	Exported	False

表 5 MTRDL 模型的数据类型字典示例

类型名称	父类型	值域
Boolean	Boolean	False, True
Modularstate_Type1	enumerated	Engaged, Disengaged
Modularstate_Type2	enumerated	Normal, Abnormal
Time	int	0,…, 1 000
Altitude	int	−1 000,…, 1,…, 20 000

如图 13 所示, 构造型《Mode》《function modular》《sensor modular》对需求中出现的模式、模块进行了定义, 包括名称、输入变量和事件、内部封装变量和事件、输出变量和事件. 以 AP 为例, 它的内部封装变量是

AP_State, 表示了 AP 当前状态是接通或断开, 输出变量是 Is_AP_Engaged, 输出给其他模块表示当前 AP 是否接通, 其他模块的内部封装变量取值可能和 Is_AP_Engaged 有关, 如 FMCP 中的内部封装变量 Is_APlamp_On 表示 AP 指示灯是否点亮, 其取值受到 Is_AP_Engaged 的影响. 为了表示模式、模块间的交互, 一方面采用如 control、refer to 等从整体结构上的关系, 另一方面采用端口和流属性对模式、模块间交互的变量进行定义, 如图中的 com_AP_FMCP 是扩展后的流属性, 其表示从 AP 到 FMCP 的信息流, 我们将其关联到输出变量 Is_AP_Engaged, 从而表示 FMCP 的内部状态迁移需要来自 AP 的输出变量 Is_AP_Engaged.

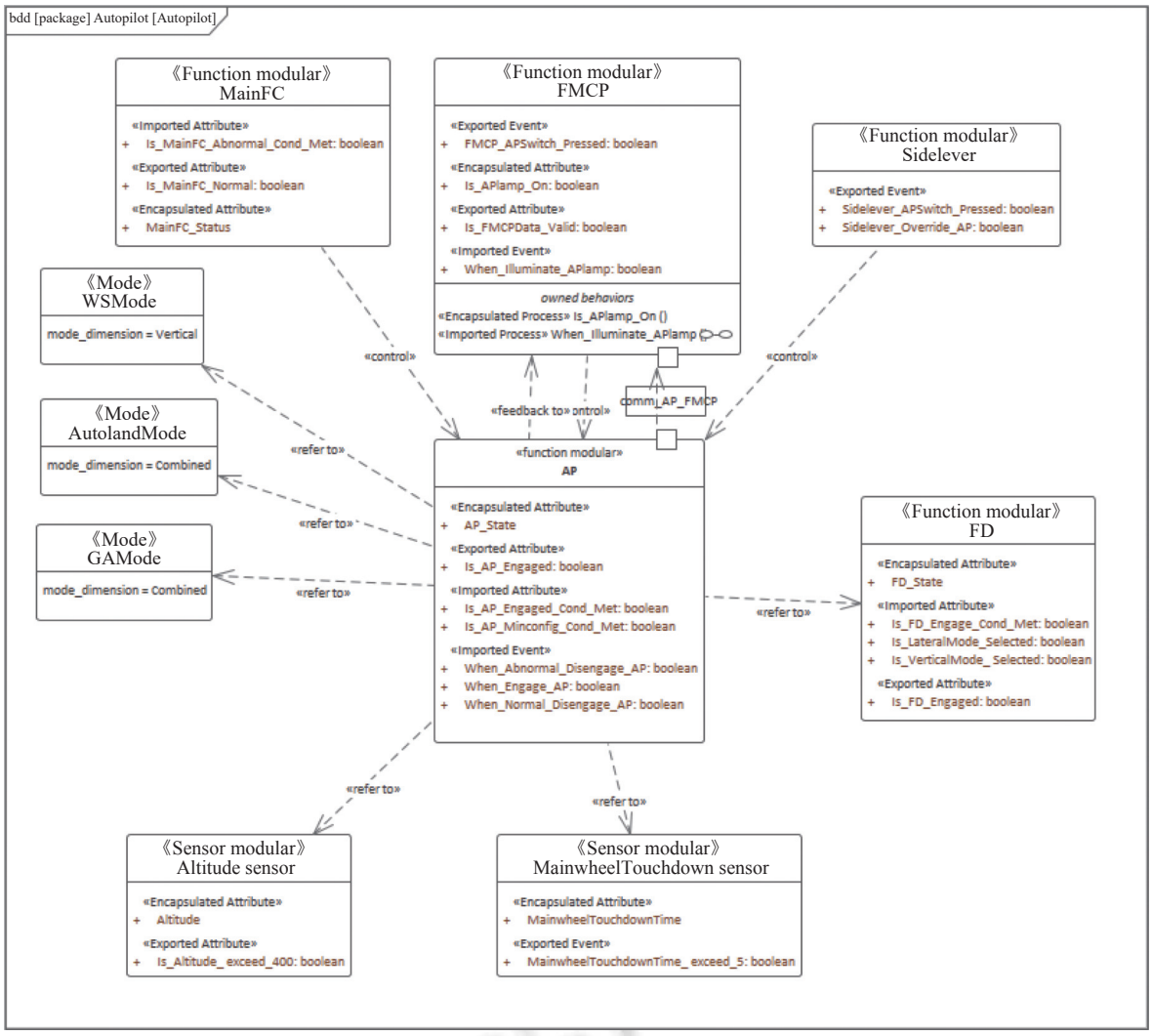

图 13 FMSysML 的 Mode&Modular 建模示例

图 14 所示的内部块图从属于 FMCP 模块, 其内部的 Process 表示了状态迁移, 我们通过端口和扩展后的流属性表示了 Process 间的联系, 以《Imported process》构造的 When_Illuminate_APlamp 为例, 其输入是通过端口接收的 AP 到 FMCP 的信息流, 即其他模式模块的输出变量或事件, 其输出是输入事件 When_Illuminate_APlamp 的取值, 该取值取决于《Trans》和《Trans_matrix_data》所定义的状态迁移. 输入事件 When_Illuminate_APlamp 作为《Encapsulated process》的输入, 参与后续的状态迁移过程.

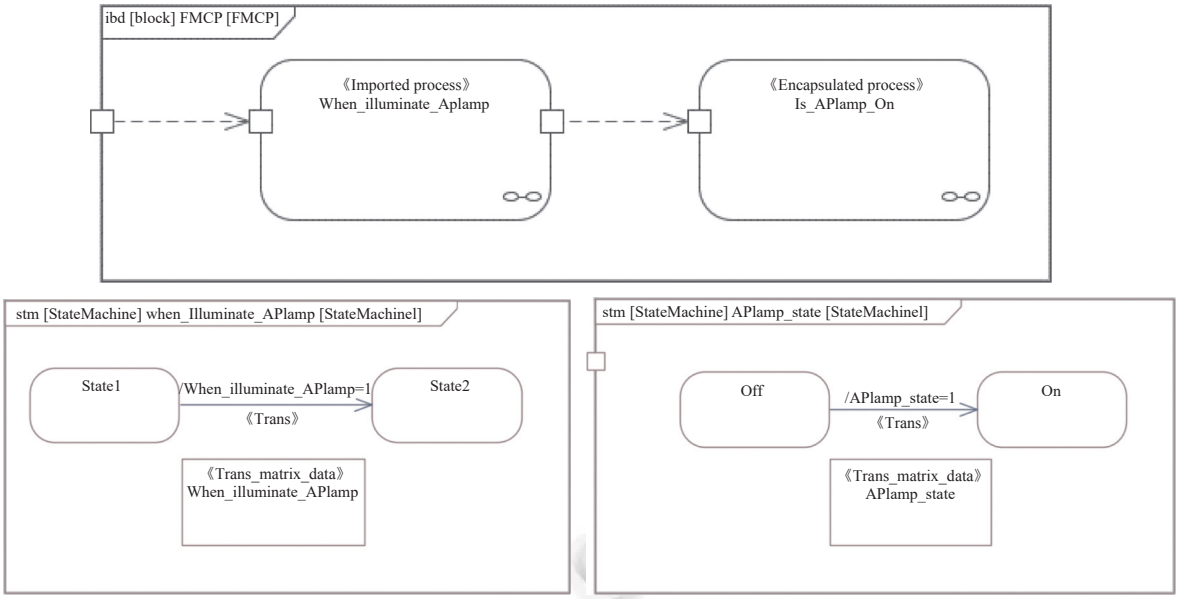

图 14 FMSysML 的 Process 建模示例

4.3 FMSysML 建模与验证结果评估

如表 6 所示, 我们对某机型自动飞行系统模式转换的条目化需求进行分析, 初期工作覆盖了 HDG、ROLL、VS 这 3 个飞行模式和其子模式, 以及 3 个模块 FMCP、AP 和 FD, 包含条目化需求数量有 61 条. 在条目化需求处理过程中, 我们定义了 42 个输入变量、32 个输入事件, 7 个内部封装变量和 15 个内部封装事件、3 个模式状态变量和 8 个模式功能变量, 47 个输出变量和 36 个输出事件. 在建立 FMSysML 模型后, 使用安全需求模板辅助生成了 25 条安全性质. 由于 FMSysML 模型不能直接进行模型检验, 因此我们将其转换到 NuSMV 模型后, 利用 NuSMV 工具进行验证, 现有相对成熟的将 SysML 模型转换到 NuSMV 的工作大多基于模型转换的规范和语言, 依附转换平台如 MOF, QVT, ATL 等建立不同模型之间的转换规则, 实现模型间自动转换[30], 本文在 AMMA 平台下对 FMSysML 和 NuSMV 模型进行元建模, 并基于 ATL 语言建立了转换规则, 实现了自动模型转换, 详细的转换过程在此不展开阐述. 现有模型转换的一致性证明工作[31,32]仍未形成统一的标准或成熟工具, 由于工业方并未对一致性提出严格的要求, 我们的现有工作主要通过目标模型是否实现了预期功能来对转换工作进行评价.

在实际应用中我们发现本文方法的优点为: (1) 早期验证: 在建模过程中可以提前发现需求描述的不一致, 如在错误的需求中 FPA 模式的切出逻辑中描述了当 FPA 模式接通且 FMCP 的 ALT HOLD 按键由 TRUE 转为 FALSE 时应从 FPA 模式转为 ALT 模式, 而在 ALT 模式的需求描述中, 当以下任意条件 (包括 FPA 模式接通) 满足且 FMCP 高度保持请求信号变为 TRUE 时应接通 ALT 模式, 通过对 FMSysML 模型的人工检查和较为前期的验证发现了两者的不一致, “ALT HOLD 按键由 TRUE 转为 FALSE”应更改为“ALT HOLD 按键由 FALSE 转为 TRUE”. (2) 识别缺少的需求: 团队能够识别出验证工作需要但可能缺少的条目化需求, 例如在 ASEL 模式需求中提到了高度捕获的条件, 但高度捕获条件的定义缺少, 这种情况极有可能是缺少了相关的需求. (3) 模型易修改: 由于 FMSysML 模型的层次化建模特征, 使得某一处需求的更改只影响其对应的 Process, 而不会对其他模型部分造成影响. 但通过和工程人员交流, FMSysM 也具有一个明显的缺点, 即工程人员无法很好地区分变量和事件的类型, 尤其是输入、内部封装和输出的分类.

在本文工作中, 时间成本主要包括人工时间成本和机器验证时间成本, 在表 6 的工作范围内花费了 5 个研究人员 3 周的工作时间, 而对单个性质验证的时间成本会随着模型的规模增大而增加, 如表 7 所示, 其中等级 1 的规

模具体为表 6 中表现的规模, 等级 2 在等级 1 的基础上增加了一个垂直模式航迹角选择 FPA 模式和一个水平模式机翼水平 Wings Level 模式, 等级 3 在等级 2 的基础上增加了一个多轴模式复飞 GA 模式.

表 6　对条目化需求的 FMSysML 建模规模

统计项目	详细分类	数量
条目化需求涵盖范围	HDG模式及其子模式相关	11
	ROLL模式及其子模式相关	12
	VS模式及其子模式相关	16
	AP模块相关	11
	FD模块相关	6
	其他模式和模块相关	5
	总计	61
FMTRDL数据字典规模	输入变量	42
	输入事件	32
	内部封装变量	7
	内部封装事件	15
	模式状态变量	3
	模式功能变量	8
	输出变量	47
	输出事件	36
FMSysML模型规模	Mode	15
	Function modular	11
	Sensor modular	9
	Process	61
	Relationship	46
	Comm	116

表 7　基于 NuSMV 的单个性质验证开销

模型规模	等级1	等级2	等级3
内存 (KB)	4.7×10^3	4.6×10^4	8.3×10^4
平均时间 (s)	0.688	1.3	2.43

4.4　分析与讨论

模型驱动工程和形式化方法用于机载系统需求验证是保障机载系统安全性可靠性的重要手段, 已成为工业界和学术界的共识, 然而如何在项目中应用并形成工具链仍然是一个迫在眉睫的难题. 工程师在使用形式化方法时, 仍过多依赖于人工经验, 缺乏自动化的工程方法以高效地对需求进行建模和性质规约. 通过在对自动飞行系统模式需求的建模和验证中我们积累了一些经验, 并形成以下总结.

(1) 现有的航空软件工程中对自然语言需求的条目化整理已相对成熟, 然而需求本身的语法语义并未有严格的约束或者管理, 对需求的层级也没有区分. 要求工程人员给出待验证的安全性质, 即使是自然语言描述的也极为困难. 因此需要根据领域特征提出一些辅助生成待验证的安全需求的方法, 如使用本文中的安全需求模板. 这种辅助规约方法虽然仅能保证在模板的能力范围内辅助生成安全需求和转换为安全性质, 其涵盖范围比不上基于专家知识总结的安全需求, 但在其能力范围内完整性通常比基于专家知识总结的安全需求更充分.

(2) 在使用 FMSysML 建模的过程中, 我们发现会出现模型不一致的情况, 主要有: a) 不同视图间的不一致, 这是由于 SysML 中缺乏视图元素的统一规则, 尤其是行为模型和结构模型中的参数统一, 虽然由于数据字典的存在下以及 FMSysML 本身的形式化语义基础, 我们有效减少了不一致情况, 但仍然需要设计由规则制导的自动识别

不一致的算法; b) 人为错误导致的不一致, 如在 Process 中有时出现将迁移和错误的转移矩阵关联的情况, 虽然前文中提到了利用矩阵元素数量来检查错误的方法, 但该方法只能解决关联到元素数量不一致的矩阵的情况, 仍需要设计更完善的不一致检查方法.

(3) 基于 FMSysML 建模能够使得到的模型和形式化模型更为接近, 从而进行自动化的转换和验证, 但也引入了更严格的语义区分, 如从条目化需求中提取变量和事件时, 工程人员常常不能明确哪些是事件, 将需要考虑当前系统状态和前一个系统状态之间特定变量集合变化的事件错认为只需要考虑当前系统状态的变量条件, 这对于后续的形式化验证是致命的. 因此领域概念库构建时建模人员需要和工程人员进行详细的讨论.

(4) 在建模和验证工作中, 我们发现更多的错误来自于条目化需求处理. 由于数组字典的内容是人工从条目化需求中抽取的, 并用于后续的建模, 因此数组字典存在定义错误的情况, 并会引发后续一系列错误. 解决这个问题, 一方面需要从建模验证过程和结果中逆向完善数组字典的方法和工具, 另一方面可能需要利用人工智能等手段形成自动化的条目化需求处理过程.

5 总结与未来工作

自动飞行系统模式转换是现代飞机系统设计过程中决定飞行安全的重要因素, 对其需求展开形式化建模验证对保障自动飞行系统安全具有重要意义. 本文提出一种自动飞行系统模式转换的领域需求描述语言 MTRDL, 基于该语言设计了在通用建模语言 SysML 上扩展的 FMSysML Profile, 并在 FMSysML 模型的基础上设计了基于安全需求模板辅助安全性质生成的方法, 填补了部分自然语言需求和基于数学语言的形式化验证方法间的鸿沟. 最后以航空领域典型需求实例说明了本文所提出方法的有效性.

本文提出的方法能够将模型驱动工程和形式化方法在自动飞行系统模式转换需求验证问题中有机结合, 提高了需求建模和安全性质规约的自动化程度, 未来我们将对已有工作细化和扩展, 包括需求模型的一致性和完整性分析工作, 通过定义规则制导的自动化算法保证需求模型的语法语义一致, 以及对 FMSysML 模型到形式化模型的自动转换一致性证明工作. 除此之外, 现有工作处理的条目化需求仍属于较高层次的需求, 虽然在现有的工程需求实例中尚未出现状态空间爆炸, 但可以预见在低层需求验证时状态空间爆炸问题仍会出现, 因此未来工作会进一步研究组合验证、增量验证等方法在验证工作中的应用.

References:

[1] Savelev A, Neretin E. Development of safety requirements for tracking active pilot controls by signals from an automatic flight control system. In: Proc. of the 2019 Int'l Conf. on Control, Artificial Intelligence, Robotics & Optimization. Athens: IEEE, 2019, 19–24. [doi: 10.1109/ICCAIRO47923.2019.00012]

[2] Hovgaard F, Mueller C, Biro G. Interaction of man and machine: Lessons learned from aviation. London: CRC Press, 2019. 746–755.

[3] Small A. Human factors analysis and classification system (HFACS): As applied to Asiana airlines flight 214. The Journal of Purdue Undergraduate Research, 2020, 10(1): 18. [doi: 10.7771/2158-4052.1485]

[4] Backes J, Cofer D, Miller S, Whalen MW. Requirements analysis of a quad-redundant flight control system. In: Proc. of the 7th Int'l Symp. on NASA Formal Methods. Pasadena: Springer, 2015. 82–96. [doi: 10.1007/978-3-319-17524-9_7]

[5] Le Sergent T, Dormoy FX, Le Guennec A. Benefits of model based system engineering for avionics systems. In: Proc. of the 8th European Congress on Embedded Real Time Software and Systems (ERTS 2016). Toulouse, 2016. 1–11.

[6] Wiels V, Delmas R, Doose D, Garoche PL, Cazin J, Durrieu G. Formal verification of critical aerospace software. Aerospace Lab, 2012, 4: 1–8.

[7] Stewart D, Liu JJ, Cofer D, Heimdahl M, Whalen MW, Peterson M. AADL-based safety analysis using formal methods applied to aircraft digital systems. Reliability Engineering & System Safety, 2021, 213: 107649. [doi: 10.1016/j.ress.2021.107649]

[8] Wang F, Yang ZB, Huang ZQ, Liu CW, Zhou Y, Bodeveix JP, Filali M. An approach to generate the traceability between restricted natural language requirements and AADL models. IEEE Trans. on Reliability, 2020, 69(1): 154–173. [doi: 10.1109/TR.2019.2936072]

[9] 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 with English abstract). http://www.jos.org.cn/1000-9825/4530.htm [doi: 10.13328/j.cnki.jos.004530]

[10] Zoughbi G, Briand L, Labiche Y. Modeling safety and airworthiness (RTCA DO-178B) information: Conceptual model and UML profile. Software & Systems Modeling, 2011, 10: 337–367. [doi: 10.1007/s10270-010-0164-x]

[11] Sun JH, Chen L, Cang FZ, Li HW, Pi FJ. Civil aircraft airborne software safety and reliability study based on RTCA/DO-178C. In: Proc. of the 5th Int'l Conf. on Computer Science and Software Engineering. New York: Association for Computing Machinery, 2022. 27–33.

[12] Liu J, Zhang X, Zhao Y. Tool qualification requirements comparison and analyses between RTCA/DO-178B and RTCA/DO-178C+ DO-330. Journal of Physics: Conf. Series, 2021, 1827(1): 012191. [doi: 10.1088/1742-6596/1827/1/012191]

[13] Ciccozzi F, Malavolta I, Selic B. Execution of UML models: A systematic review of research and practice. Software & Systems Modeling, 2019, 18(3): 2313–2360. [doi: 10.1007/s10270-018-0675-4]

[14] Wolny S, Mazak A, Carpella C, Geist V, Wimmer M. Thirteen years of SysML: A systematic mapping study. Software & Systems Modeling, 2020, 19(1): 111–169. [doi: 10.1007/s10270-019-00735-y]

[15] Deng PY, Zhou Q, An D, Wang SH, Li K. A modeling method of agents and SOA in advanced avionics system based on AADL. Applied Sciences, 2022, 12(16): 8157. [doi: 10.3390/app12168157]

[16] Insaurralde CC, Seminario MA, Jimenez JF, Giron-Sierra JM. Model-driven system development for distributed fuel management in avionics. Journal of Aerospace Information Systems, 2013, 10(2): 71–86. [doi: 10.2514/1.53714]

[17] Mercer E, Slind K, Amundson I, Cofer D, Babar J, Hardin D. Synthesizing verified components for cyber assured systems engineering. Software and Systems Modeling, 2023, 22(5): 1451–1471. [doi: 10.1007/s10270-023-01096-3]

[18] Hatcliff J, Belt J, Robby, *et al*. Automated property-based testing from AADL component contracts. In: Proc. of the 2023 Int'l Conf. on Formal Methods for Industrial Critical Systems. Cham: Springer, 2023. 131–150.

[19] Zou L, Zhan NJ, Wang SL, Fränzle M. Formal verification of Simulink/Stateflow diagrams. In: Proc. of the 13th Int'l Symp. on Automated Technology for Verification and Analysis. Shanghai: Springer, 2015. 464–481. [doi: 10.1007/978-3-319-24953-7_33]

[20] Colaço JL, Pagano B, Pouzet M. SCADE 6: A formal language for embedded critical software development (invited paper). In: Proc. of the 2017 Int'l Symp. on Theoretical Aspects of Software Engineering (TASE). Sophia Antipolis: IEEE, 2017. 1–11. [doi: 10.1109/TASE.2017.8285623]

[21] Whalen MW, Innis JD, Miller SP, Wagner LG. ADGS-2100 Adaptive Display and Guidance System Window Manager Analysis. 2006. http://shemesh.larc.nasa.gov/fm/fmcollinspubs.html

[22] 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-companion Volume. Vancouver: IEEE, 2009. 18–27. [doi: 10.1109/ICSE-COMPANION.2009.5070960]

[23] Wagner L, Mebsout A, Tinelli C, Cofer D, Slind K. Qualification of a model checker for avionics software verification. In: Proc. of the 9th Int'l Symp. on NASA Formal Methods. Moffett Field: Springer, 2017. 404–419. [doi: 10.1007/978-3-319-57288-8_29]

[24] Zhang M, Yue T, Ali S, Zhang HH, Wu J. A systematic approach to automatically derive test cases from use cases specified in restricted natural languages. In: Proc. of the 8th Int'l Conf. on System Analysis and Modeling: Models and Reusability. Valencia: Springer, 2014. 142–157. [doi: 10.1007/978-3-319-11743-0_10]

[25] Yue T, Briand LC, Labiche Y. aToucan: An automated framework to derive UML analysis models from use case models. ACM Trans. on Software Engineering and Methodology, 2015, 24(3): 13. [doi: 10.1145/2699697]

[26] Wang F, Yang ZB, Huang ZQ, Zhou Y, Liu CW, Zhang WB, Xue L, Xu JM. Approach for generating AADL model based on restricted natural language requirement template. Ruan Jian Xue Bao/Journal of Software, 2018, 29(8): 2350–2370 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5530.htm [doi: 10.13328/j.cnki.jos.005530]

[27] Wang J, Zhan NJ, Feng XY, Liu ZM. Overview of formal methods. Ruan Jian Xue Bao/Journal of Software, 2019, 30(1): 33–61 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5652.htm [doi: 10.13328/j.cnki.jos.005652]

[28] Plaku E. Planning in discrete and continuous spaces: From LTL tasks to robot motions. In: Advances in Autonomous Robotics: Joint Proc. of the 13th Annual TAROS Conf. and the 15th Annual FIRA RoboWorld Congres. Bristol: Springer, 2012. 331–342. [doi: 10.1007/978-3-642-32527-4_30]

[29] Buyukkocak AT, Aksaray D, Yazıcıoğlu Y. Planning of heterogeneous multi-agent systems under signal temporal logic specifications with integral predicates. IEEE Robotics and Automation Letters, 2021, 6(2): 1375–1382. [doi: 10.1109/LRA.2021.3057049]

[30] Caltais G, Leitner-Fischer F, Leue S, Weiser J. SysML to NuSMV model transformation via object-orientation. In: Proc. of the 6th Int'l Workshop on Cyber Physical Systems. Design, Modeling, and Evaluation. Pittsburgh: Springer, 2017. 31–45. [doi: 10.1007/978-3-319-51738-4_3]

[31] Liu H. Description and proof of property preservation of model transformations. Ruan Jian Xue Bao/Journal of Software, 2007, 18(10): 2369–2379 (in Chinese with English abstract). https://www.jos.org.cn/jos/article/abstract/20071001

[32] Lin QQ, Wang SL, Zhan BH, Gu B. Modelling and verification of real-time publish and subscribe protocol using UPPAAL and

Simulink/Stateflow. Journal of Computer Science and Technology, 2020, 35(6): 1324–1342. [doi: 10.1007/s11390-020-0537-8]

附中文参考文献:

[9] 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 [doi: 10.13328/j.cnki.jos.004530]

[26] 王飞, 杨志斌, 黄志球, 周勇, 刘承威, 章文炳, 薛垒, 许金淼. 基于限定自然语言需求模板的 AADL 模型生成方法. 软件学报, 2018, 29(8): 2350–2370. http://www.jos.org.cn/1000-9825/5530.htm [doi: 10.13328/j.cnki.jos.005530]

[27] 王戟, 詹乃军, 冯新宇, 刘志明. 形式化方法概貌. 软件学报, 2019, 30(1): 33–61. http://www.jos.org.cn/1000-9825/5652.htm [doi: 10.13328/j.cnki.jos.005652]

[31] 刘辉, 麻志毅, 邵维忠. 模型转换中特性保持的描述与验证. 软件学报, 2007, 18(10): 2369–2379. https://www.jos.org.cn/jos/article/abstract/20071001

徐恒(1994－), 男, 博士生, CCF 学生会员, 主要研究领域为系统安全性分析, 形式化方法.

陶传奇(1984－), 男, 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, 软件安全性, 形式化方法.

黄志球(1965－), 男, 博士, 教授, 博士生导师, CCF 杰出会员, 主要研究领域为软件工程, 软件安全性, 形式化方法.

王金永(1983－), 男, 博士, 讲师, CCF 专业会员, 主要研究领域为形式化方法与应用, 安全人工智能.

胡军(1973－), 男, 博士, 副教授, CCF 专业会员, 主要研究领域为模型驱动的系统安全性分析, 软件验证.

石帆(1995－), 男, 博士生, 主要研究领域为形式化方法, 强化学习.