基于故障树分析的嵌入式系统AADL模型可靠性分析方法

李东民 李 静 林华锋
(南京航空航天大学计算机科学与技术学院 南京 210016)
(软件新技术与产业化协同创新中心 南京 211100)

摘 要 采用架构分析与设计语言(AADL)建立嵌入式系统的半形式化模型,实现从AADL模型到静态故障树(Static Fault Tree,SFT)模型的转换,并根据故障树定量分析法对系统可靠性进行分析。首先结合AADL错误模型附件建立可靠性模型;然后设计了从AADL模型到SFT模型的语义映射规则,并实现了将AADL模型中的基本元素转换为静态故障树中相对应的元素;最后结合飞机车轮刹车系统实例,使用文献中提出的方法对其进行可靠性分析,从而验证所提方法的可行性和有效性。

关键词 AADL,故障树分析,可靠性分析,模型转换
中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.06.031

Reliability Analysis Method of Embedded System AADL Model Based on Fault Tree Analysis

LI Dong-min LI Jing LIN Hua-feng
(College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China)
(Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing 211100, China)

Abstract We used architecture analysis and designed language(AADL) to build embedded system semi-formalization model, transformed AADL model to static fault tree (SFT) model, and analyzed the reliability of the system according to the fault tree analysis method. Firstly, the reliability model is built with the attachment of the AADL error model. Then, the semantic mapping rules from AADL model to SFT model are designed and used to transform from AADL model to SFT model. Finally, based on the example of aircraft wheel brake system, the reliability analysis is carried out according to the method proposed in this paper to prove the feasibility and effectiveness of the proposed method.

Keywords AADL, Fault tree analysis, Reliability analysis, Model transformation

1 引言

嵌入式系统开发技术的不断发展,促使嵌入式被广泛应用于航空航天、工业制造等领域。这些领域的嵌入式系统具有规模庞大、产品复杂、开发周期长等特点,对诸如可靠性、安全性及可调度性等非功能属性要求较高,导致传统的嵌入式系统开发方法不能满足复杂关键系统的开发要求,并导致软件安全事故频出,因此,系统的可靠性便成为嵌入式系统开发的关注热点。可靠性分析是指通过概率的方式对嵌入式系统可能发生的故障和系统的稳定性进行定量的分析。传统的系统可靠性分析主要依靠工程师根据经验对系统的设计文档和设计模型的研究得出,这种分析方法面对大型的系统时无法快速、准确地得到分析结果。为此必须采用一种规范的系统可靠性分析方法,这种方法要充分考虑到系统各构件、构件之间的依赖关系以及硬件在系统运行过程中存在的安全问题,从而实现对整体系统可靠性的分析。

美国自动化工程师协会(Society of Automotive Engineers,SAE)于2004年发布了航空标准AS5506——架构分析与设计语言(Architecture Analysis and Design Language,AADL)[1-2]。AADL不关心系统具体的实现方式,而是通过构件之间的相互连接实现软件与硬件的绑定,通过该语言可以高效构建嵌入式系统模型,其主要用于描述嵌入式系统的结构。SAE在2006年发布的AS5506/1标准中首次提出错误模型附件EMA(Error Model Annex)[3],附件提供了对错误信息进行定义的一些属性,结合这些属性可以对系统的可靠性进行定量的分析,实现对软件设计研发的指导。

AADL是一种半形式化建模语言,有确定的语义和严格语法的语言表达规范,但其只能对构件的属性进行描述,例如构件错误模型、构件安全等级等,即通过AADL语言构建的嵌入式系统可靠性模型只是一个静态的模型。为了对系统可靠性进行分析,还需要采用仿真和形式化方法对AADL模型中各构件的交互行为的动态性进行描述[4]。系统可靠性分析

到稿日期:2016-05-16 返修日期:2016-08-18 本文受中央高校基本科研业务费专项资金(NS2015092)资助。
李东民(1992-),男,硕士生,主要研究方向为软件可靠性研究,E-mail:495261077@qq.com;李 静(1976-),女,博士,副教授,主要研究方向为数据挖掘、软件形式化验证;林华锋(1992-),男,硕士生,主要研究方向为图像显著性检测。

的常规方法包括马尔科夫分析法、故障树分析法等。其中马尔科夫分析法的计算量很容易随着系统规模的增大而发生状态爆炸，且马尔科夫链无法对系统构件间的交互行为进行动态描述，只能描述单个构件不同状态的迁移，需对其进行转化后才能应用于系统可靠性分析。故障树分析方法对构件故障的刻画比较简单，没有将构件划分为不同状态，避免了状态爆炸的情形，相比于马尔科夫分析法分析过程，时间复杂度更低；同时，从分析结果中不但能够得到系统总体故障率，还能得到各构件的错误事件对系统整体可靠性的影响程度。

本文第 2 节阐述了结合 AADL 模型进行软件可靠性分析的相关问题和研究；第 3 节介绍了基于 EMA 进行 AADL 可靠性模型的创建，给出了故障树模型的基本元素，并设计了 AADL 模型到故障树模型的映射规则以及一种实现二者间快速转换的高效、准确的转换流程；第 4 节建立飞机车轮控制系统的 AADL 模型，并通过使用第 3 节所提出的模型转换规则得到相应的故障树模型，实现对该系统的定量分析；最后总结全文。

2　相关工作

可靠性最早是由硬件研究人员提出，指一款产品或者工具在规定的条件及规定时间内完成规定功能的能力。与之相对应的软件可靠性是指软件内部不存在错误、在一定时间条件下正确执行的能力。嵌入式系统的可靠性分析需要将两者结合。AADL 在构建嵌入式系统软硬件结构的同时，可以通过错误附件对系统中各构件的安全信息进行刻画，这使得其在工业界和学术界受到关注和重视。很多公司基于 AADL 模型进行可靠性建模与分析研究，并开发出一些相关分析工具。如 AADL 模型仿真测试工具 AADS，其实现了从 AADL 到 SystemC 的转换，基于 SystemC 代码对 AADL 模型进行仿真和性能分析；Isograph 软件是由英国 Isograph 公司开发的可靠性、可用性、安全性、维修性和保障性工程设计分析软件，采用故障树分析法、马尔可夫分析和可靠性框图分析等方法对系统可靠性进行分析。

基于 AADL 的可靠性分析主要采用 AADL 语言进行模型构建，运用仿真和形式化方法进行模型的分析与验证。国内外很多学者研究了 AADL 模型与各种形式化语言模型的转换规则。如文献[5]设计了 AADL 模型转化为 IMA 模型的转化规则，并基于 IMA 模型进行了可靠性分析；文献[6]针对医疗系统软件在使用中存在的安全问题，通过将 AADL 错误模型与 STAMP 模型相结合，提出了 STPA 方法来对系统进行可靠性分析；文献[7]研究了基于 AADL 模型生成故障树模型，其在转化过程中将错误模型存储为有向图的故障树模型，通过消除环后得到最终模型，转化流程较为复杂；文献[8]提出将故障树与诊断专家系统相结合的方法，实现对航天器控制系统的故障诊断，最后结合 VC＋＋编程语言和 SQL2000 数据库开发了航天器姿态控制发动机故障诊断专家系统。

国内也有许多研究者对 AADL 模型的可靠性分析进行研究。文献[11]提出了从 AADL 到 GSPN 模型的转换规则，并提出了对 GSPN 模型的改进；文献[12]基于 AADL 和 GSPN 模型进行可靠性分析，并设计了可靠性分析软件 ARAM；文献[13]设计并实现了 AADL 可靠性析工具 RAT，并结合实际系统验证了工具的可靠性；文献[14]提出了一种将 AADL 模型转化为交互式马尔科夫链模型的方法；文献[15]基于 AADL 模型与 GSPN 模型的映射规则，设计了一种模型自动转换插件。

故障树分析法的缺点是构造故障树的过程较为复杂且有大量重复工作，面对大型系统中各构件错综复杂的交互结构时，人工构建故障树会发生故障遗漏或误建。通过构建 AADL 模型可以弥补这一不足，其模型的构建过程简单且构件之间通过错误传播的方式建立各种故障之间的联系，能清晰表达系统内故障之间复杂的逻辑联系。本文提出一套完整的 AADL 模型元素到故障树模型元素的映射规则，并通过更简洁的转化流程实现二者之间的转化，通过这种方式提高可靠性分析方法的分析效率，使之能实现对大规模复杂系统的可靠性分析。

3　AADL 模型与 SFT 模型的转换

使用 OSATE 创建嵌入式系统的 AADL 模型并实例化生成对应的 AAXL 格式文件，通过转化插件将文件语义进行解析，转换为对应的 SFT 语义文件，然后利用 FAT 分析工具对生成的 SFT 模型文件进行模拟仿真和结果分析。该方法的核心工作是 AADL 错误模型的创建和对应故障树模型的生成。主要流程如图 1 所示。

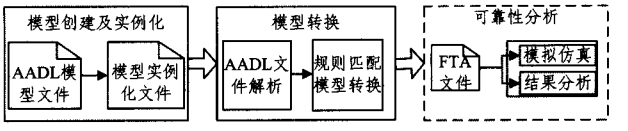

图 1　AADL 模型可靠性分析流程

3.1　AADL 模型的创建

AADL 通过构件(component)和连接(connection)等概念描述系统的体系结构[9]；通过特征(feature)和属性(property)描述系统的功能和非功能性质[17]；通过定义用户属性和附件(annex)来扩充构件的功能和非功能性质，实现针对不同使用环境的模型设计[19]。创建 AADL 可靠性模型的主要流程下。

1)体系结构建模，包括硬件构件设计和软件构件设计。硬件构件建模是通过 AADL 执行平台组件为嵌入式系统硬件进行建模；软件构件建模是通过 AADL 应用软件组件为嵌入式系统数据传输、处理和中断服务程序进行建模。

2)通过属性绑定的方式描述硬件构件、软件构件之间的相互交互行为，包括实时调度、中断处理、资源管理和系统服务等行为。

3)为各个构件添加相应的错误属性(安全等级、发生概率、错误类型等)和错误传播信息。

错误附件是一种能够关联到 AADL 构件或连接的状态机，

用于描述系统错误、错误行为以及错误传播，采用错误模型注释的 AADL 模型可以实现基于模型的可靠性分析。错误模型的描述主要分为：错误类型模型和错误实现模型。2013 年提出 EMA2.0 之后，规定模型的声明和实现可以放在一起统一描述[20]。

错误声明主要包括错误状态(Error State)、错误事件(Error Event)以及发生(Occurrence)属性。其中错误状态定义了构件可转化的状态，当发生错误事件之后，构件的状态会发生转变；发生属性规定了错误发生的概率[21]。

如图 2 所示，代码中包含 3 种错误事件，分别是 Operational，NoValueEvent，LateValueEvent；包含 3 种状态，其中 Operational 是初始状态，另外两种错误状态分别为 Alternate 和 Failed，这两种信息是错误声明的基本信息。

```
annex EMV2 { **
error types //错误模型类型
NoPower : type;
NoService : type;
ValueError : type;
NoValue : type extends ValueError;
PlatformFailure : type;
HardwareFailure : type extends PlatformFailure;
SoftwareFailure : type extends PlatformFailure;
end types;
error behavior bscu
events
BadValueEvent: error event; //错误事件
NoValueEvent: error event; //错误事件
LateValueEvent: error event; //错误事件
states
Operational : initial state ; //初始状态
Alternate : state; //错误状态
Failed : state; //错误状态
end behavior;
** };
```

图 2 独立构件的错误模型

在 AADL 模型中，若一个构件与另一个构件之间存在信息交互，则认为两个构件为依赖构件，构件之间通过端口进行信息传播，传播的内容包括错误信息和非错误信息。有依赖构件的 AADL 模型如图 3 所示。本文主要介绍构件之间错误信息的传播。错误模型中将错误传播分为 In-propagation 和 Out-propagation 两种。Out-propagation 负责构件错误的传出，通过端口将错误传播给与这个构件有依赖关系的其他构件；In-propagation 负责接受其他构件传入的错误。一个 Out-propagation 可以对应多个 In-propagation。

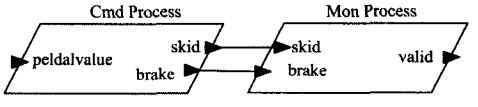

图 3 有依赖构件的 AADL 模型

图 4 是有依赖构件的错误模型代码，在代码中可以定义 error propagations 和 error behavior。s1－s4 定义了 In-propagation 和 Out-propagation；s5 和 s8 表示该构件作为 NoService，SoftWareFailure 和 HardwareFailure 错误传播的终点，构件对该错误做出处理并终止该错误的传播；s6－s7 表示构件是 NoValue 错误传播的源头，这个构件产生 NoValue 错误并向外传播；s9－s11 定义构件对错误的处理行为，即构件状态由 Operational 转变为 Failed。

```
annex EMV2 { **
use types error_library;
use behavior error_library::simple;
error propagations
s1 pedalvalue :inpropagation{NoService};
s2 brake:outpropagation{NoValue};
s3 skid:outpropagation{NoValue};
s4 processor : in propagation {SoftwareFailure,HardwareFailure};
flows
s5 nopedal :error sink pedalvalue{NoService};
s6 noskid:error source skid{NoValue};
s7 nobrake:error source brake{NoValue};
s8 platformerr:error sink processor{SoftwareFailure,HardwareFai-
   lure};
end propagations;
componen terro rbehavior
transitions
s9 terrfrompedal: Operational －[pedalvalue{NoService}]－>
   Failed;
s10 terrfromplatformsoft : Operational －[processor{SoftwareFai-
   lure}]－>Failed;
s11 terrfromplatformhard : Operational －[processor{HardwareFai-
   lure}]－>Failed;
propagations
p1 : Failed －[ ]－> brake{NoValue};
p2 : Failed －[ ]－> skid{NoValue};
endcomponent;
** };
```

图 4 有依赖构件的错误模型

3.2 SFT 模型以及映射规则

静态故障树 SFT 通过有向图的方式体现故障之间的传播关系，将最不希望发生的故障作为顶事件，按照对象的结构和功能逐层展开，直到不可分事件(底事件)为止；层与层之间通过逻辑符连接，以分析各零部件故障对总体系统可靠性的影响。其优点是能够实现快速诊断；可靠性诊断只需根据系统模型构造出相应故障树即可，无需相关领域知识。缺点是由于故障树是由相应系统模型得到，因此不能分析不可预知的故障[5]。

故障树主要包括以下几个元素。

1)顶事件：分析系统最不希望发生的事件，位于故障树顶端。它总是逻辑门的输出事件而不可能是任何逻辑门的输入事件，在故障树中通常用矩形符号表示。

2)中间事件：除顶事件以外的其他结果事件均属于中间事件，位于顶事件和底事件之间。它既可能是某个逻辑门的

输入，也可能是逻辑门的输出，在故障树中也用矩形符号表示。

3)底事件：位于故障树底部的事件。它只能是逻辑门的输入事件，在故障树中通常用圆形符号表示。

4)或门：若所有输入事件中至少有一个事件发生，则输出事件发生。

5)与门：表示仅当所有输入事件都发生时，输出事件才会发生。

通过分析 AADL 模型中的基本元素以及故障树模型中的基本元素，可以得出这两种模型语言中元素的对应关系。下面给出 AADL 模型到故障树模型的基本转换规则[18]。

规则 1　错误模型中的错误状态转换为故障树中的顶事件或中间事件。

规则 2　错误模型中的错误事件转换为故障树中的底事件或基本事件。

规则 3　错误模型的连接弧转换为故障树中的逻辑门，若构件之间通过 and 连接则转换为与门，若构件之间通过 or 连接则转换为或门。表 1 列出了 AADL 错误模型转换为故障树模型的基本转换规则。

表 1　基本的 AADL 到 SFT 的映射规则

AADL 错误模型元素	SFT 模型	
	元素	符号
错误状态	顶事件或中间事件	□
错误事件	底事件或基本事件	○
源状态-[事件]-目标状态	与门 或门	

规则 4　若多个错误事件发生均导致一个构件转变为相同的错误状态，则将错误状态转换为中间事件，各错误事件转换为这个中间事件的叶子节点。依据事件之间的关系，添加相应的逻辑门来连接各元素。如果这几个错误事件均发生才导致状态发生转变，则用与门连接；若任意一个事件发生就导致转换则用或门连接。简单构件所对应的故障树模型如图 5 所示。

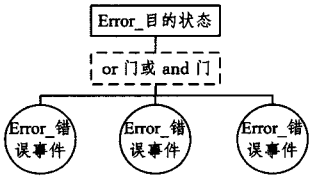

图 5　简单构件对应的故障树模型

规则 5　不同构件可以通过端口构成相互依赖关系，端口既可以传递数据和事件，也可以是错误的传播路径。只有单个错误传播的故障树如图 6(a)所示；一个构件与多个构件相关联的故障树模型如图 6(b)所示。

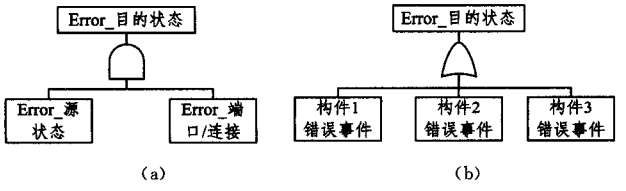

图 6　有依赖的构件对应的故障树模型

规则 6　构件之间除了错误传播外，还有错误过滤/屏蔽机制，它们可以保证构件不受一些类型的错误的影响。在转换过程中将错误过滤/屏蔽之后的构件状态转化为中间事件，导致错误传播的错误事件转化为这个中间事件的叶子节点，节点之间根据屏蔽/过滤的逻辑关系，用相应的门连接。

3.3　静态故障树的生成

故障树分析的核心是将 AADL 模型转化为对应的故障树模型，转化过程主要分为两个步骤：提取构件错误模型和生成故障树。

(1)提取构件错误模型：系统错误模型实例包括构件错误模型实例、通信错误模型实例以及系统实例。若要生成整个系统的 SFT 模型，首先需要提取出系统中的错误模型实例，并基于模型转换规则将这些错误模型实例进行转化。构件声明和连接声明能够直接用于实例获取。

(2)生成故障树：提取得到各构件错误模型后，需要通过转化规则将其转化为对应的故障树模型。在错误模型中，如果构件之间存在依赖关系，则按照有依赖构件对应的故障树模型转换规则进行转换。故障传播的转换就是在源构件和目标构件之间添加相应的逻辑门，以反映原 AADL 模型中的相互依赖关系。

通过对 AADL 模型对应的 XML 模型进行解析，提取关于模型的构件信息、错误模型信息以及错误模型架构信息，基于之前提出的转换规则将模型元素对应转换成故障树模型。转换的整体过程如图 7 所示，遍历 AADL 模型中的所有构件信息，检查其中是否存在错误模型。如果存在错误模型，再判断是否存在错误传播事件。若不存在错误传播事件，则按照简单构件模型转换规则进行转换；如果存在错误传播事件，判断其是否存在错误过滤和错误屏蔽，如果不存在则记录传播事件的状态迁移，若存在则记录错误过滤和错误屏蔽信息。遍历完所有的构件后，基于错误传播和错误过滤、屏蔽转换规则对模型进行转换，生成最终的故障树模型。

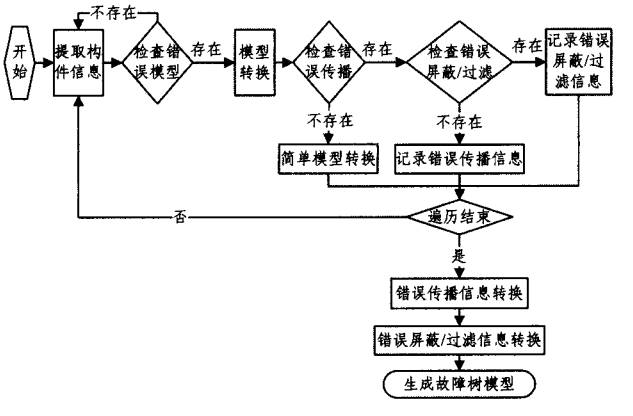

图 7 故障树模型生成流程图

图 8 是图 4 中错误模型所对应的故障树模型。根据模型定义的错误类型，以下 3 种错误事件的发生会导致 Command 构件产生故障。

(1)构件在 Operational 状态，In-propagation 传入 NoService 错误。

(2)构件在 Operational 状态，In-propagation 传入 SoftwareFailure 错误。

(3)构件在 Operational 状态,In-propagation 传入 HardwareFailure 错误。

以上 3 种情况中任意一种情况发生均会导致构件发生错误,所以可将其转化为带有 3 个输入事件的或门连接的故障树模型。

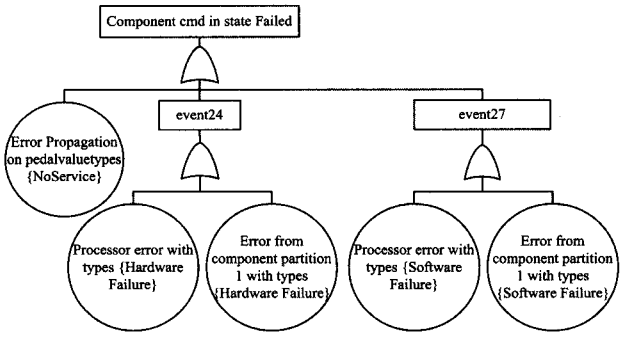

图 8 Command 构件故障树模型

3.4 故障树定量分析

故障树的定量分析是指计算出系统的顶事件发生的概率以及系统的一些可靠性指标。计算过程需要给出底事件发生的概率,利用底事件的发生概率以及故障树模型的结构,求出顶事件发生的概率以及每个底事件对顶事件发生的重要程度,即重要度分析。

割集是故障树中一些底事件的集合,当这些底事件发生时顶事件必然发生。最小割集是指在这些底事件的集合中去掉任意一个则不再是割集。在分析顶事件是否发生时需要基于最小割集。因为最小割集中各底事件是逻辑与的关系,所以最小割集的故障概率为它包含的各事件的概率乘积。

对于一个包含 n 个最小割集$\{c_1,c_2,\cdots,c_n\}$的顶事件 T,顶事件发生的概率可由式(1)表示:

$$P_T=\sum_{i=1}^{n}P(c_i)-\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}P(c_ic_j)+\sum_{i=1}^{n-2}\sum_{j=i+1}^{n-1}\sum_{k=j+1}^{n}[P(c_ic_j\,c_k+\cdots+(-1)^nP(c_1c_2c_n))] \tag{1}$$

当最小割集数量足够大时,在计算顶事件概率时可能会出现状态爆炸问题,因为其计算项为 2^{n-1} 项,每一项又是许多项的连乘。在实际工程计算中可将顶事件概率进行近似计算,如式(2)所示:

$$P_T=\sum_{i=1}^{n}P(c_i) \tag{2}$$

重要度是指一个部件或系统发生故障时对顶事件发生的影响程度。它是关于部件的可靠性参数以及系统结构的函数,是系统中各构件重要程度的一种度量,若重要度越大,则它所处的环节越薄弱,在系统中的地位越重要。

底事件 x_i 的重要度可通过式(3)计算,其中 P_{x_i} 表示底事件 x_i 发生的概率,P_T 表示顶事件 T 发生的概率。

$$P_{x_i|T}=\frac{P_{x_i}}{P_T} \tag{3}$$

将全部基本事件的重要度按从大到小排列,即可得出基本事件的重要度顺序。重要度顺序有利于更加清楚地认识系统的薄弱环节,在改善系统设计、系统安全性分析以及系统故障维修环节提供有用的依据。

4 系统测试

本文以飞机车轮刹车控制系统为例,建立了系统的 AADL 模型。该模型由两个子系统构件 sul 和 sub2、命令构件(Command Component)、监控构件(Monitor Component)、选择构件(Select_Alternate System)、处理器构件(Platform)和若干端口连接组成,模型结构如图 9 所示。该系统设计了两个子系统 sub1 和 sub2,每个子系统中各包含一个 Command 和 Monitor 构件来接收刹车系统和供电系统传来的数据并判断数据是否有效;再通过 Select_Alternate 构件选择使用其中一个子系统产生的数据并输出给其他系统,以在飞机着陆时进行减速控制。

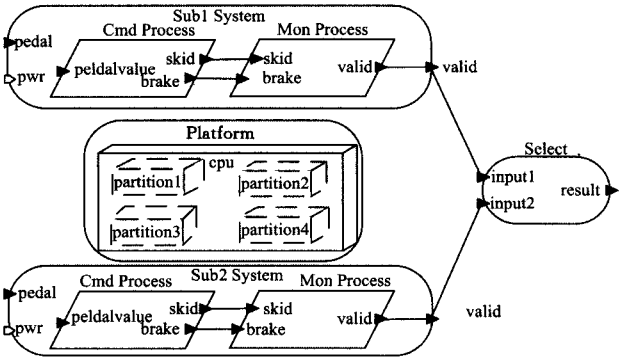

图 9 刹车控制系统 AADL 模型

在 OSTAE 中完成模型创建,语法验证通过之后对系统进行实例化,实例化得到模型对应的 AAXL 模型文件;依据第 3 节所提出的转换规则设计一款转化插件,读入 AAXL 类型文件并生成包含 SFT 模型的 FTA 类型文件。

为测试本文设计的转化方法的转化效率,设计了几个不同规模的测试模型进行转化测试。转化测试在一台内存为 4GB、CPU 频率为 3.3GHz 的计算机上完成,测试结果如图 10 所示。转化所用时间会受模型复杂度(包括模型中错误数量以及构件数量)的影响,本文选择错误数量作为自变量,转化时间(以秒为单位)作为因变量。从图 10 中可以看出,本文设计的转化方法在进行模型转化时消耗的时间较少,证明其是一种高效的 AADL 模型到 SFT 模型的转化方法。

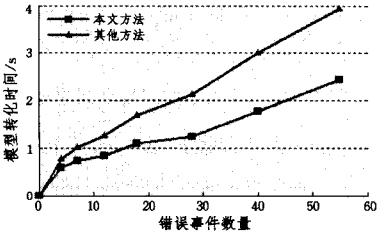

图 10 模型转化时间随错误数量的变化情况

采用蒙特卡洛模拟法(Monte Carlo Simulation)对 SFT 模型进行可靠性模拟。其原理是在每一次实验过程中对每一个故障生成一个 0 到 1 之间的随机数,如果这个数小于或等于时间发生的概率,则认为故障发生,否则认为故障不发生;再通过故障树的与门和或门的结合方式判断顶层时间是否发生故障。

经过转化之后的故障树模型中共包括 28 个错误事件。其中硬件故障(Hardware Failure)是由于处理器运行发生故障而产生，依据硬件厂商提供的可靠性数据设置概率；软件故障(如 NoPower，Noservice 等)依据类似系统故障发生的平均概率设定概率，各构件的错误发生概率如表 2 所列。

表 2　各类型错误发生的概率

错误类型	错误概率 λ/年
NoPower	0.0015
NoValue	0.00211
NoService	0.0032
HardwareFailure	0.00204
SoftwareFailure	0.0019
SelectError	0.006

在进行模拟分析时可以设定模拟实验的次数，测试结果如图 11 所示。可以发现，在实验次数设置得比较少的情况即在 5000 次以下时，故障概率变化较大，这是因为测试次数较少时故障发生的随机性比较明显，从图 11 可以看出故障概率在 0.014 之上浮动很大。当实验次数增大之后，概率就稳定在 0.014 上下，浮动较小。当选择一个较大的实验次数时能得到更加可靠的结果。

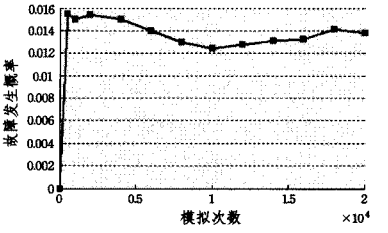

图 11　故障概率随模拟次数的变化情况

测试时同时记录每次顶事件发生故障是由哪些底事件引起的，根据这些信息报告结果可以给出每个底事件对发生总体故障的影响情况，这些分析结果是无法通过马尔科夫分析法获得的。表 3 给出了事件重要度前 15 的重要度排序。从中可以看出，Monitor 构件失效造成的影响最大；接着是 HardwareFailuer 和 SoftwareFailure 类型错误，这两个类型错误是在处理器构件 Platform 中产生的，因此工程设计人员在设计时应特别关注这些设备，尽可能降低其出错概率，从而使整体系统可靠性得到显著提升。

表 3　事件重要度排序

Failure Description	Failure Contribution	Importance/%
Monitor that causes an invalid	1.620804E−004	18.84
Monitor that causes an invalid	1.527296E−004	17.75
Processor error with types {HardwareFailure}	8.415714E−005	9.78
component cmd on skid with types {NoValue}	8.104021E−005	9.42
component cmd on skid with types {NoValue}	7.168941E−005	8.33
Error Propagation on unknown propagation pointtypes {NoPower}	6.857247E−005	7.97
Processor error with types {HardwareFailure}	6.545555E−005	7.61
Error from component partition2 with types {SoftwareFailure}	6.233861E−005	7.25
Processor error with types {SoftwareFailure}	5.922169E−005	6.88
Processor error with types {SoftwareFailure}	5.922169E−005	6.88
Error from component partition4 with types {SoftwareFailure}	5.922169E−005	6.88
Error from component partition1 with types {HardwareFailure}	5.610475E−005	6.52
Processor error with types {SoftwareFailure}	5.610475E−005	6.52
Error from component partition2 with types {HardwareFailure}	5.610475E−005	6.52
Error from component cmd on skid with types {NoValue}	5.610475E−005	6.52

结束语　本文提出了基于故障树分析的方法来对 AADL 模型进行可靠性及定量分析，同时提出了两种模型元素之间的映射规则并实现了转换，利用飞机车轮控制系统证明了所提方法的可靠性。下一步的研究主要集中于扩展所提出的转换规则，使之能适应错误模型附件中不断添加的新元素，使转换得到的 SFT 模型对可靠性有更高的分析精度。

参 考 文 献

[1] WANG G, ZHOU X S, ZHANG F, et al. Research on Model-based Testing on AADL[J]. Computer Science, 2009, 36(11): 127-130. (in Chinese)
王庚，周兴社，张凡，等. AADL 模型的测试方法研究[J]. 计算机科学，2009，36(11)：127-130.

[2] YANG Z B, LEI P I, KAI H U, et al. AADL: An Architecture Design and Analysis Language for Complex Embedded Real-Time Systems[J]. Journal of Software, 2010, 21(5): 899-915.

[3] DELANGE J, FEILER P. Architecture Fault Modeling with the AADL Error-Model Annex[C]// 2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA). IEEE, 2014: 361-368.

[4] GU B, DONG, WEI X. A Qualitative Safety Analysis Method for AADL Model[C]// IEEE International Conference on Software Security and Reliability-Companion. 2014: 213-217.

[5] ZHANG Q, WANG S, LIU B. Approach for integrated modular avionics reconfiguration modelling and reliability analysis based on AADL[J]. Iet Software, 2016, 10(1): 18-25.

[6] PROCTER S, HATCLIFF J. An architecturally-integrated, systems-based hazard analysis for medical applications [C] // Twelfth ACM/IEEE International Conference on Formal Methods and MODELS for Codesign. IEEE, 2014: 124-133.

[7] ZHANG Q, WANG S, LIU B. Some Improvements on the Rules for Exchanging Between Error Model Annex and AADL to Fault Tree[C]// 2013 International Conference on Information Technology and Applications (ITA). IEEE, 2013: 338-342.

[8] SUN H, HAUPTMAN M, LUTZ R. Integrating Product-Line

Fault Tree Analysis into AADL Models[C]// IEEE High Assurance Systems Engineering Symposium. IEEE Computer Society, 2007: 15-22.

[9] SINGHOFF F, LEGRAND J, NANA L, et al. Cheddar: a flexible real time scheduling framework[J]. Abstr, 2004, xxiv(4): 1-8.

[10] JEAN-PAUL B, RAPHAËL C, DAVID C, et al. A mapping from AADL to Java-RTSJ[C]// International Workshop on Java Technologies for Real-Time and Embedded Systems, Jtres 2007. Institute of Computer Engineering, Vienna University of Technology, September 2007, Vienna, Austria. 2007: 165-174.

[11] GAO J L, ZHANG G, JING X C, et al. Reliability modeling and evaluation method of software system based on AADL [J]. Frontiers of Computer Science and Technology, 2011, 5(10): 942-952. (in Chinese)
高金梁, 张刚, 经小川, 等. 采用 AADL 的软件系统可靠性建模与评估方法[J]. 计算机科学与探索, 2011, 5(10): 942-952.

[12] DONG Y W, WANG G R, ZHANG F, et al. AADL model reliability analysis and evaluation tool[J]. Journal of Software, 2011, 22(6): 1252-1266. (in Chinese)
董云卫, 王广仁, 张凡, 等. AADL 模型可靠性分析评估工具[J]. 软件学报, 2011, 22(6): 1252-1266.

[13] LIU J J, MENG H N. Reliability analysis tool of avionics system based on AADL[J]. Modern Electronic Technology, 2014, 37(8): 65-68. (in Chinese)
刘建军, 孟海宁. 基于 AADL 的航电系统可靠性分析工具[J]. 现代电子技术, 2014, 37(8): 65-68.

[14] CHENG Y H, HUANG Z Q, KAN S L. System reliability modeling method combining AADL and IMC[J]. Computer Engineering and Science, 2015, 37(8): 1517-1524. (in Chinese)
程亦涵, 黄志球, 阚双龙. 一种结合 AADL 和 IMC 的系统可靠性建模方法[J]. 计算机工程与科学, 2015, 37(8): 1517-1524.

[15] TANG Y, SU W, LI S Y. AADL model to the generalized stochastic Petri net conversion tool[J]. Modern Electronic Technology, 2015, 38(12): 62-65. (in Chinese)
汤玥, 苏威, 李蜀瑜. AADL 模型到广义随机 Petri 网的转换工具[J]. 现代电子技术, 2015, 38(12): 62-65.

[16] SHEN N M, LI J, BAI H Y. Transformation and verification of AADL data flow model of real time system based on Uppaal[J]. Computer Science, 2016(1): 211-217.

[17] SHARVIA S, PAPADOPOOULOS Y. Integrating Model Checking with HiP-HOPS in Model-Based Safety Analysis[J]. Reliability Engineering System Safety, 2015, 135: 64-80.

[18] SU W. Research on verification technology of embedded software system based on AADL[D]. Xi'an: Shanxi Normal University, 2015. (in Chinese)
苏威. 基于 AADL 的嵌入式软件系统验证技术研究[D]. 西安: 陕西师范大学, 2015.

[19] GAO L, DONG Y W, ZHANG F. AADL system reliability model conversion method [J]. Computer Engineering, 2011, 37(14): 21-26.

[20] BOZZANO M, CIMATI A, KATOEN J P, et al. Safety, Dependability and Performance Analysis of Extended AADL Models [J]. Computer Journal, 2011, 54(5): 754-775.

[21] YANG Z B, PI L, HU K, et al. Architecture design and analysis of complex embedded real time systems: AADL[J]. Journal of Software, 2010, 21(5): 899-915. (in Chinese)
杨志斌, 皮磊, 胡凯, 等. 复杂嵌入式实时系统体系结构设计与分析语言: AADL[J]. 软件学报, 2010, 21(5): 899-915.

(上接第 176 页)

[10] JANG J W, KIM S H. Quaternary sequences with good autocorrelation constructed by Gray mapping[J]. IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences, 2009, E92-A(8): 2139-2140.

[11] YANG Z, KE P H. Construction of quaternary sequences of length pq with low correlation[J]. Cryptography and Communications, 2011, 3(2): 55-64.

[12] DU X N, CHEN Z X. Linear complexity of quaternary sequence generated using generalized cyclotomic classes modulo 2p[J]. IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences, 2011, 94(5): 1214-1217.

[13] KE P H. New classes of quaternary cyclotomic sequence of length $2p^m$ with high linear complexity[J]. Inf. Process. Lett., 2012, 112(16): 646-650.

[14] CHANG Z L, LI D D. On the linear complexity of quaternary cyclotomic sequences with the period $2pq$[J]. IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences, 2014, 97(2): 679-684.

[15] LI D D, WEN Q Y, ZHANG J, et al. Linear complexity of generalized cyclotomic quaternary sequences with period pq[J]. IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences, 2014, E97-A(5): 1153-1158.

[16] WANG G H, DU X N, WAN Y Q, et al. Linear complexity of balanced quaternary generalized cyclotomic sequences with period pq[J]. Journal of Shandong University (Natural Science), 2016, 51(9): 145-150. (in Chinese)
王国辉, 杜小妮, 万韫琦, 等. 周期为 pq 的平衡四元广义分圆序列的线性复杂度[J]. 山东大学学报(理学版), 2016, 51(9): 145-150.

[17] CHANG Z L, LI D D. On the linear complexity of generalized cyclotomic binary sequence of length $2pq$[J]. Applicable Algebra in Engineering, Communication and Computing, 2010: 21(2): 93-108.