基于模态的嵌入式软件动态重构技术研究

覃杨森 董云卫
(西北工业大学计算机学院 西安 710072)

摘 要 机载航空电子系统设计采用综合化系统体系结构,可实现计算系统及其计算资源和计算设施的"物理集成";以及机载嵌入式软件系统的"功能集成";提供对系统计算功能的动态配置管理和实时动态冗余,以期得到较高的计算性能和保障系统的高可靠性。基于软件系统架构的层次关系研究了复杂嵌入式计算任务的运行模态表示方法,分析了嵌入式软件系统任务模态的迁移关系,提出了基于AADL软件体系结构的嵌入式软件模态划分方法,制定了系统动态重构蓝图,并设计了基于模态的嵌入式软件动态重构实施方法。基于软件架构的模态分析及其动态重构,有助于提高复杂嵌入式软件系统的可靠性、安全性和重用性。

关键词 软件体系结构,动态重构,模态
中图法分类号 TP391 **文献标识码** A

Research on Embedded Software Dynamic Reconfigurable Technology Based on Mode

QIN Yang-sen DONG Yun-wei
(School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China)

Abstract The airborne avionics electronic system design adopts the Integrated Modular Avionics(IMA), enabling the physical integration of the computational system, its computational resources as well as its computational infrastructure, and the functional integration of the airborne embedded software system. It can provide the system computational function with dynamic configuration management and real time dynamic redundancy. This paper, based on hierarchical relationship of the software system construction, studied the running mode representation of complicated embedded computational tasks, analysed the transfer relationship of the embedded software task modes, proposed a AADL software architecture-based embedded software mode partition method, made a system dynamic reconfigurable blueprint, and worked out a mode-based implement method for the software dynamic reconfiguration. The mode analyses based on software architecture mode and its dynamic reconfiguration will facilitate promotion of reliability, safety and reusability of complicated embedded software systems.

Keywords Software architecture, Dynamic reconfiguration, Mode

新一代航空电子系统采用综合化系统体系结构(Integrated Modular Avionics, IMA),使得系统具有开放性、可预测性、可维护性、可重用性、高可靠性和高安全性。ARINC653就是为了满足航电系统综合化设计的需求而提出的[1]。ARINC653应用程序接口规范定义了标准的API和系统服务,即APEX层,将应用程序和操作系统划分开来,并通过分区来管理运行的任务及其所需资源,以保证不同应用QoS服务需求。遵循ARINC653标准开发的操作系统可实现运行平台以及操作系统的资源重构,满足了IMA系统要求,如VxWorks653[2],其可移植性、可重用性、高可靠性以及高安全性得到了充分保证。里斯本大学的J. Craveiro则在Linux系统环境下研究了基于模型的分区调度问题,深入阐述了适配层在分区操作系统的重要意义[3]。然而,目前这样的软件系统不能充分解决应用层上的资源重构,以致难以解决系统资源利用率低以及保持系统高可靠性和高安全性的问题。因此,在应用层上实现软件系统的动态重构,是提高软件系统资源利用效率、保障系统可靠性的重要手段。同时,在应用层上对软件系统进行动态重构,可以实时监测到系统的运行状态。通过对系统模态的划分和系统模态的重配置,可使软件系统及时地适应任务环境的改变,确保系统的可靠性和安全性,避免因系统失效而导致的灾难性故障,从而提高软件系统的可预测性。

1 软件模态

软件模态(Mode)是指一个软件系统功能在运行时可操作配置的状态,是对软件系统进行的逻辑划分。划分可以依据软件系统不同的运行阶段、运行的不同任务场景以及软件系统内部功能模块的聚合度。在一个模态内,可对系统资源和属性进行配置。一个软件系统可以以多种可配置的状态运行,并且不同模态之间在某种条件下能够进行模态迁移。系

到稿日期:2011-03-23 返修日期:2011-05-17 本文受国家自然基金重点课题(60736017),国家863计划课题(2009AA01Z147)资助。
覃杨森(1986-),硕士生,主要研究方向为嵌入式软件,E-mail:qinyangsen991@126.com;董云卫(1968-),男,教授,博士生导师,主要研究方向为嵌入式软件设计、验证与测试。

统根据软件模态当前的运行状况，计算得出软件系统当前模态的安全性、可靠性等系统状态特征来决定系统模态跃迁。本文着眼于软件模态迁移机制研究，通过评估不同软件模态运行时的系统可信属性，结合软件功能的模态划分对软件系统的功能动态重构进行指导，以减少系统运行的故障，避免系统失效，保障软件系统的安全性和可靠性。

1.1 软件模态的组成

随着软件规模的不断扩大及其功能的不断增多，软件系统被分为多个层次。其中，使命是为达到特定目标而聚合在一起的可以满足高性能和高可靠性要求的计算任务集合。从软件需求分析的角度，使命完成了软件需求分析阶段定义的各个软件需求。软件系统根据完成任务的不同被分为多个使命，使命是对软件粗粒度的划分。从软件设计的角度，单一使命由一个或多个功能模块完成，各个功能实现并完成使命的基础。从软件开发角度，根据构件化软件开发[4]的原理，功能是由一个或多个构件实现，构件是对软件细粒度的划分。这样，软件系统就存在使命-功能-构件这样的 3 层结构，如图 1 所示。

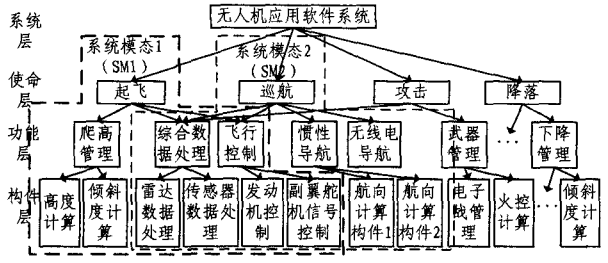

图 1 系统模态示意图

根据软件结构的不同层次，软件模态也被分为多个层次。在系统层次，软件系统具有多种使命。软件系统为完成某一种使命需要调用计算资源，执行相应软件功能的运行方式就成为系统的一种模态，记为 SM_i(System Mode)，i 表示系统的第 i 个模态。系统完成每个使命都会执行相应的模态，并且该模态只有处于激活状态，对应的使命才能处于运行状态。

在使命层次，一个使命的完成可以用不同的软件功能组合执行。因此，即便是同一使命也可以有多种运行方式，称其为使命模态，记为 M_iM_j(Mission Mode)，它表示使命 M_i 的第 j 个模态。在使命模态下，多个功能可以处于同一种使命模态，并且，只有处于同一个使命模态下的软件功能才可以处于同一运行状态。

在功能层次，一个功能的执行由多种软件构件组合完成，每一种软件构件的组成方式就称为使命下的功能模态，记为 $M_iF_jM_k$(Mission Function Mode)，它表示使命 i 下的功能 j 的第 k 个模态。一个构件可以处于一种或多种功能模态中，当且仅当该模态处于激活状态，该功能模态对应的构件才处于运行状态。

1.2 软件模态的迁移

软件系统由多个使命构成，不同的使命表示了软件不同的任务和性能要求。每个使命由一个或多个功能模块构成，功能模块是完成特定功能的单元。基于构件化开发的软件系统的功能由多个构件组成。如图 1 所示，一架智能化无人驾驶飞机的应用系统由{起飞、巡航、攻击、降落} 4 个使命组成，它的运行包含如下几个运行模态：开始时执行起飞模态，完成起飞使命；当飞机爬升到规定的高度之后，切换到巡航模态，完成巡航使命，在巡航过程中搜寻目标；如果在巡航过程中发现目标，飞机就自动切换到攻击模态，完成攻击使命，实现对目标的打击；当目标摧毁后，它又切换到巡航模态，继续搜寻目标；当任务完成后就可以返回，系统就切换到返航降落模态，完成降落使命。

由于模态具有层次性的特点，下面将分别阐述模态的迁移在不同层次发生的情形。

1.2.1 系统模态的迁移

使命模态间不同的配置方式形成不同的系统模态。当软件系统的使命需要发生切换时，通过模态迁移实现使命的切换。如图 1 所示，无人机应用软件系统具有 4 种使命：起飞、巡航、攻击和降落；系统具有 4 种模态，分别为 SM_1(系统模态 1)，SM_2(系统模态 2)，SM_3(系统模态 3)和 SM_4(系统模态 4)。起飞使命处于 SM_1 下，巡航使命处于 SM_2 下，攻击使命处于 SM_3 下，降落使命处于 SM_4 下。

若使命处于某一系统模态下，则完成该使命的各个功能模块和实现各功能的构件均处于该系统模态下，如图 1 所示。爬高管理是完成起飞使命的功能之一，高度计算构件和倾斜度计算构件是完成爬高管理功能的两个构件。起飞使命处于 SM_1 模态下，意味着爬高管理和高度计算构件、倾斜度计算构件都处于系统模态 SM_1 下。若软件系统当前执行的是起飞使命，当软件系统需要由起飞使命切换至巡航使命时，就要发生系统级的模态迁移，软件系统由模态 SM_1 迁移至模态 SM_2。

1.2.2 使命模态的迁移

功能模态间不同的配置方式形成不同的使命模态。使命模态之间通过发生模态迁移实现使命下对功能的运行配置。如图 2 所示，巡航使命由 4 种功能模块完成：飞行控制、综合数据处理、惯性导航和无线电导航。巡航使命包含两种模态，分别为使命模态 1(M_2M_1)和使命模态 2(M_2M_2)。惯性导航、飞行控制和综合数据处理处于 M_2M_1 下，无线电导航、飞行控制和综合数据处理处于 M_2M_2 下。

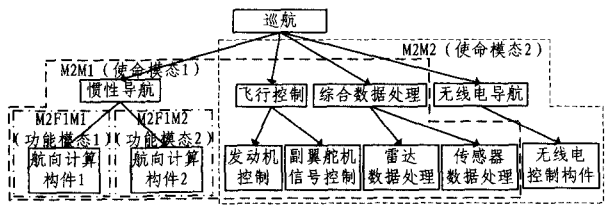

图 2 使命模态示意图

巡航使命的模态可由模态 M_2M_1 迁移至模态 M_2M_2，实现功能运行的配置。当功能处于某一使命模态下时，完成该功能的各个构件均处于该使命模态下。如图 2 所示，综合数据处理功能下的雷达数据处理构件和传感器数据处理构件均处于对应的使命模态下。

1.2.3 功能模态的迁移

构件间不同的配置方式形成不同的功能模态。功能模态之间通过发生模态迁移实现功能下对构件的运行配置。如图 2 所示，惯性导航功能由两个构件完成：航向计算构件 1 和航向计算构件 2，并包含两种模态。航向计算构件 1 处于功能模态 1($M_2F_1M_1$)下，航向计算构件 2 处于功能模态 2($M_2F_1M_2$)下。功能 1 的模态可由功能模态 1($M_2F_1M_1$)迁移至功能模态 2($M_2F_1M_2$)，实现构件的配置。

2 软件系统动态重构

根据以上所划分的软件模态层次结构，进行软件模态动态重构配置有如下 3 种方式。

(1)当软件系统的使命发生切换时，系统模态发生迁移，从而实现切换之后的使命。在图 1 中，起飞使命处于的系统模态为系统模态 1(SM_1)，巡航使命处于的系统模态为系统模态 2(SM_2)。无人机起飞并爬升到一定的高度后需要实现巡航使命，系统模态则由系统模态 1(SM_1)切换至系统模态 2(SM_2)，此时系统发生使命级别的动态重构配置。

(2)同一个使命下，若某一功能模块不再满足系统性能需求，则系统须发生重构，由其它冗余的功能模块替换该功能模块。实现这种重构的关键是存在功能层次的冗余，如图 2 所示。图中惯性导航功能与无线电导航功能冗余，巡航使命当前的激活模态为使命模态 1(M_2M_1)。系统能够发生功能级别的重构，巡航使命的模态由 M_2M_1 迁移至 M_2M_2，此时惯性导航功能不再处于激活模态，而无线电导航功能处于激活模态，从而实现了功能之间的动态重构配置。

(3)同一功能下，若某一构件不再满足系统性能需求，系统须发生重构，由其它冗余的构件模块替换该构件模块。实现这种重构的关键是存在构件层次的冗余，如图 2 所示，图中惯性导航功能下的两个构件：航向计算构件 1 和航向计算构件 2 冗余。由于航向计算构件 2 和航向计算构件 1 冗余，因此软件系统能够发生重构，惯性导航功能的模态由 $M_2F_1M_1$ 迁移至 $M_2F_1M_2$。此时航向计算构件 1 不再处于激活模态，而由航向计算构件 2 处于激活模态，从而实现了构件之间的动态重构配置。

3 动态重构实施过程

3.1 动态重构框架

可动态重构的软件系统通过改变软件系统的运行模态进行软件构件的动态配置。在软件重用[5]领域，对于不能动态重构的软件系统，如果构件不满足重用需求，则该构件不能进行重用。但是对于可动态重构的软件系统，由于其具有多种模态，因此当某一模态下软件构件不满足重用需求时，需要判断其他模态下的构件是否满足重用的功能和性能要求。若某一模态下的构件满足重用需求，表示在该模态下能够进行重用，则可以将软件系统中的模态切换，完成对软件系统的重构。针对可重构系统的特性，本文提出了基于模态的动态重构方法[6]，如图 3 所示。

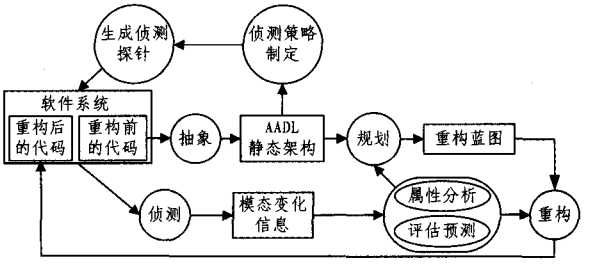

图 3 软件动态重构框架

在图 3 的重构框架中，方法对软件系统的代码进行架构抽象，即抽象出软件架构的 AADL[7]静态模型。软件架构具有相应形式的构件集合，其中包括数据构件、执行构件和连接构件。分析 AADL 的静态模型，针对软件系统制定合理的系统运行侦测方案，目的在于实时获取系统运行的动态信息，从而得到软件系统模态的变化情况。根据侦测到的模态变化信息，对软件系统架构进行性能分析，评估架构在各个模态下的可靠性和安全性，并预测出软件模态可靠性和安全性的变化趋势，判断是否会出现失效状态和运行故障。根据各模态可靠性和安全性的评估结果，进行重构规划，得到软件的重构蓝图。重构蓝图由一系列信息表组成，用以描述软件架构模态的迁移以及相关属性。依据重构蓝图中描述的软件模态、模态迁移以及可靠性、安全性评估结果，验证当前模态是否满足重用性能要求。如果满足要求，则可以将该架构进行重用，否则需要对软件架构进行重构，使架构满足性能的要求。

3.2 动态重构实施

AADL 中的模态描述是从构件、连接以及属性等的配置关系，描述一个系统或者构件可选择的操作状态。在模态描述中，模态被激活意味着当前处于激活模态的一组构件以及可用于传输数据和控制的构件连接被激活。当一个构件具有多个模态时，模态迁移用来描述事件的到来引起模态的改变。模态迁移通过对构件内部配置变化的描述来说明动态行为。模态的迁移通过模态迁移事件进行触发，模态迁移事件可以是外部接受的事件，也可以由构件本身产生，例如端口中的事件、事件数据的到来引起模态迁移。AADL 描述的重构通过模态迁移实现，当软件系统需要进行重构时，通过系统各个级别的模态迁移实现资源的重新配置。

图 1 所描述的软件“使命-功能-构件” 3 层结构在 AADL 模型中对应为不同的软件构件。对于软件系统，一个软件系统对应为 AADL 中的系统构件；构成软件系统的各个使命对应为 AADL 中的子系统构件，各个子系统构件为系统构件的子构件；完成各个使命的功能对应为 AADL 中的进程构件，各个进程构件作为子系统构件的子构件；完成各个功能的构件对应为 AADL 中的线程、线程组构件。前面提到，软件重构是通过模态迁移实现的。在 AADL 中，使用 mode 对模态进行描述，mode 之间的迁移描述模态迁移。

本文以功能的重构描述和构件的重构描述为例，说明如何用 AADL 中 mode 之间的迁移来描述出模态间的迁移过程。在“使命-功能-构件” 3 层结构中，资源冗余包括功能层次的冗余以及构件层次的冗余。对于功能层次的冗余，使用 AADL 中的子系统构件描述软件系统中需要完成的使命，这样该子系统构件只有具有多个模态才能进行重构。冗余的功能模块处于不同的子系统模态中。如图 4 所示，子系统 Mission. imp 具有两个模态 MM_1 和 MM_2，其中 MM_1 是初始模态，在事件 switch_to_MM_2 触发下，子系统模态由 MM_1 迁移为 MM_2；在事件 switch_to_MM_1 触发下，子系统模态由 MM_2 迁移为 MM_1。

图 5 描述的两个功能冗余层次的进程构件均处于不同的模态下。使命 Mission 具有两个模态 MM_1 和 MM_2，并具有两个相互冗余的进程子构件 Version1 和 Version2。其中 Version1 处于使命模态 MM_1 下，Version2 处于模态使命 MM_2 下。子系统构件可以进行重构，由模态 MM_1 迁移至 MM_2，此时，构件 Version2 处于激活模态，而构件 Version1 不再处于激活模态。

system implementation Mission. imp
modes
MM_1:initial mode;
MM_2:mode;
MM_1-[switch_to_MM_2] -> MM_2;
MM_2-[switch_to_MM_1] -> MM_1;

图 4 使命模态描述

system implementation Mission. imp
subcomponents
Version1:process Func. v1 in modes(MM_1);
Version2:process Func. v2 in modes(MM_2);
end Mission. imp;

图 5 冗余的功能处于不同模态

对于构件层次的冗余,在 AADL 中,以进程构件描述功能模块,以线程、线程组构件描述构件模块。进程构件只有具有多个模态才能进行重构,冗余的线程构件处于不同的进程模态中。如图 6 所示,进程 Function. imp 具有两个模态 MFM_1 和 MFM_2,其中 MFM_1 是初始模态;在事件 switch_to_MFM_2 触发下,该进程的模态能够由 MFM_1 迁移为 MFM_2;在事件 switch_to_MFM_1 触发下,该进程的模态能够由 MFM_2 迁移为 MFM_1。

process implementation Function. imp
modes
MFM_1:initial mode;
MFM_2:mode;
MFM_1-[switch_to_MFM_2] -> MFM_2;
MFM_2-[switch_to_MFM_1] -> MFM_1;

图 6 功能模态描述

图 7 描述的两个构件冗余层次的线程构件均处于不同的模态下。功能 Function 具有两个模态 MFM_1 和 MFM_2,并具有两个线程子构件 Version1 和 Version2。其中 Version1 处于模态 MFM_1 下,Version2 处于模态 MFM_2 下。初始情况下,功能 Function 处于模态 MFM_1 下,构件 Version1 处于激活状态;进程构件可以进行重构,由模态 MFM_1 迁移至 MFM_2,此时构件 Version2 处于激活模态,而构件 Version1 不再处于激活模态。

process implementation Function. imp
subcomponents
Version1:thread Component. v1 in modes(MFM_1);
Version2:thread Component. v2 in modes(MFM_2);
end Function. imp;

图 7 冗余的构件处于不同模态

3.3 架构重构评估

在进行架构性能评估时,由于本文采用 AADL 模型,因此这里主要说明 AADL 模型的性能分析方法。本文使用了 Ana-Elena Rugina 提出的对 AADL 架构模型进行可靠性分析的方法[8]。这种方法的主要特点是:将 AADL 架构模型转换为 GSPN(广义随机 Petri 网)模型;制定了一套转换规则,对架构模型的可靠性进行量化分析。具体分析过程可参考文献[9]。

3.4 动态重构蓝图制定

在传统工程方法学中,工程师首先分析产品需求,然后开发可理解的、描述产品设计的工程蓝图(Blueprint)[10],最后再根据这个蓝图指导开发实际的产品。根据本文设计的软件架构动态重构框架,在完成架构抽象之后,对软件架构进行规划生成重构蓝图。重构蓝图由一系列的信息表组成。重构信息表能够描述软件架构的组成结构、架构发生模态迁移时的属性信息,通过重构信息表能够对重构进行规划。重构信息表包括:系统组成表、系统模态重构信息表、使命模态重构信息表以及功能模态重构信息表。系统组成表描述的是系统的基本组成,包括软件系统层次的划分、各个层次所处的模态等;系统模态下的重构信息表描述的是各个系统模态之间的迁移信息以及在各个系统模态下的系统属性信息;使命模态下的重构信息表描述的是各个使命模态之间的迁移信息以及在各个使命模态下的系统属性信息;功能模态下的重构信息表描述的是各个功能模态之间的迁移信息以及在各个功能模态下的系统属性信息。

系统组成表如表 1 所列。系统组成表共分 5 项,包括使命栏、系统模态栏、功能栏、使命模态栏、构件栏和功能模态栏。使命栏为系统的各个使命;系统模态栏为使命所处的系统模态;功能栏为该使命包含的各个功能;使命模态栏为相应的功能所处的使命模态;构件栏为功能细分的各个构件;功能模态栏为构件所处的功能模态。

表 1 系统组成表

使命	系统模态	功能	使命模态	构件	功能模态
使命 1	SM_1	Func1	M_1M_1	Component 1	$M_1F_1M_1$
				Component 2	$M_1F_1M_1$,$M_1F_1M_2$
		Func2	M_1M_1	Component 3	$M_1F_2M_1$
			M_1M_2	Component 4	$M_1F_2M_2$
使命 2	SM_2	Func3	M_2M_1	Component 5	$M_2F_3M_1$
			M_2M_2	Component 6	$M_2F_3M_2$
		Func4	M_2M_2	Component 7	$M_2F_4M_1$
				Component 8	$M_2F_4M_2$

系统模态重构信息表能够描述系统模态的迁移信息。系统模态重构信息如表 2 所列。在表 2 中,"源系统模态"栏记录的是模态迁移之前的系统模态;"源系统属性"栏记录的是在源系统模态下的系统属性,包括可靠性、安全性等;"目标模态"栏记录的是源系统模态可以迁移到的系统模态;"目标系统属性"栏记录的是在目标模态下的系统属性,包括可靠性、安全性等;"迁移条件"栏记录的是系统模态由源系统模态迁移至目标系统模态的条件。

表 2 系统模态重构信息表

源系统模态	源系统属性		目标系统模态	目标系统属性		迁移条件
	可靠性	安全性		可靠性	安全性	

以图 4 为例,按照如上的信息表构造方式,可以构造出如表 3 所列的使命模态重构信息表。其中,Rs 为源使命模态 MM_1 的可靠性值,Ss 为源使命模态 MM_1 的安全性值;Rt 为目标使命模态 MM_2 的可靠性值,St 为目标使命模态 MM_2 的安全性值。

表 3 使命模态重构信息表

源使命模态	当前系统属性		目标使命模态	目标系统属性		迁移条件
	可靠性	安全性		可靠性	安全性	
MM_1	Rs	Ss	MM_2	Rt	St	switch_to_MM_2

(下转第 194 页)

长。

结束语 本文针对多维流数据，提出了一种基于信息熵的噪声检测算法 EDLOF。依据局部噪声点的思想，在数据属性的信息增益的基础上，给出了数据点的局部离群熵的定义，并根据数据点的局部离群熵寻找到流数据中的噪声数据。实验 1 在 UCI Zoo 数据集上将 EDLOF 算法与 ENBROD 算法进行对比，实验表明 EDLOF 算法对噪声数据的检测有较高的准确性，而且对数据的规模没有要求，并且其对于训练集中未出现的未知噪声数据类型有着较好的泛化能力。实验 2 在 UCI CUP99 数据集上将 EDLOF 算法与 DSOKP 算法进行对比，实验表明在正确率相差不大的情况下，由于数据的过度拟合等原因，EDLOF 算法检测出的正常数据比 DSOKP 算法检测出的多，但是 EDLOF 算法有着更高的执行效率。

参 考 文 献

[1] Hawkins D. Identification of Outliers[M]. Chapman and Hall, London: Chapman and Hall, 1980

[2] 金澈清，钱卫宁，周傲英. 流数据分析与管理综述[J]. 软件学报，2004，15(8)：1172-1181

[3] Breunig M M, Kriegel H-P, Ng R T, et al. LOF: Identifying Density-Based Local Outliers[A]// Proc. ACM SIGMOD 2000, Dallas[C]. ACM Press. New York: ACM, 2000: 93-104

[4] 倪巍伟，陈耿，陆介平，等. 基于局部信息熵的加权子空间离群点检测算法[J]. 计算机研究与发展，2008，45(7)：1189-1194

[5] 于绍越，商琳. 基于信息熵的相对离群点的检测方法：ENBROD[J]. 南京大学学报：自然科学版，2008，44(2)：212-218

[6] 徐翔，刘建伟，罗雄麟. 离群点挖掘研究[J]. 计算机应用研究，2009，26(1)：34-40

[7] 薛安荣，鞠时光. 局部离群点挖掘算法研究[J]. 计算机学报，2007，30(8)：1455-1463

[8] 熊家军，李庆华. 信息熵理论与入侵检测聚类问题研究[J]. 小型微型计算机系统，2005，26(7)：1163-1166

[9] 张月琴. 滑动窗口中数据流频繁项集挖掘方法[J]. 计算机工程与应用，2010，46(16)：132-134

[10] 曾颖，罗可，邹瑞芝. 基于 K-均值聚类和凝聚聚类的离群点查找方法[J]. 计算机工程与应用，2009，45(26)：131-133

[11] 毛国君，宗东军. 基于多维数据流挖掘技术的入侵检测模型与算法[J]. 计算机研究与发展，2009，46(4)：602-609

[12] 倪巍伟，陆介平，陈耿，等. 基于 k 均值分区的数据流离群点检测算法[J]. 计算机研究与发展，2006，43(9)：1639-1643

[13] http://archive.ics.uci.edu/ml/datasets/Zoo, http://archive.ics.uci.edu/ml/machine-learning-databases/zoo/

[14] http://kdd.ics.uci.edu/databases/kddcup99/

(上接第 178 页)

依据以上的系统组成表和重构信息表，可以分别作出系统模态层、使命模态层以及功能模态层的重构蓝图规划。以表 1 中的系统组成为例，得到的系统模态规划蓝图如图 8 所示。

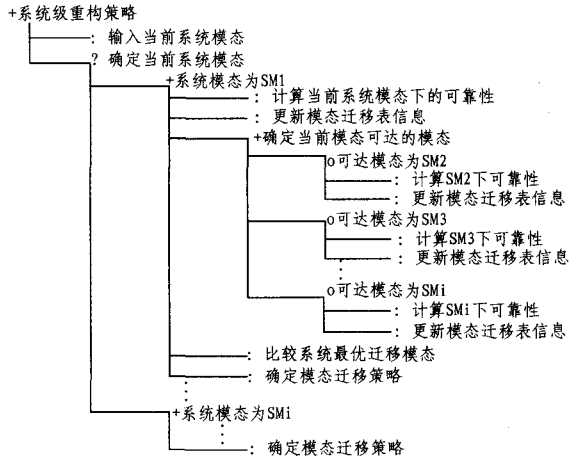

图 8 系统模态重构蓝图

在图 8 所示的重构蓝图中，输入当前系统模态之后，进入多选流程。若当前系统模态为 SM_1，则计算当前系统模态 SM_1 下的软件可靠性、安全性等属性，并在重构信息表中更新属性信息。接着判断当前系统模态 SM_1 可以迁移的模态。若可达模态 SM_i，则计算系统模态 SM_i 下的软件可靠性、安全性等属性，并在重构信息表中更新属性信息。当所有的可达模态计算完毕之后，比较系统最优的模态迁移，并确定模态迁移策略。按照图 8 所示，同样可以得到使命模态和功能模态的规划蓝图。系统按照规划蓝图，完成在系统模态层、使命模态层和功能模态层的动态重构。

结束语 本文基于软件系统架构的层次关系，研究了复杂嵌入式计算任务的运行模态表示方法，分析了嵌入式软件系统任务模态的迁移关系，提出了基于 AADL 软件体系结构的嵌入式软件模态划分方法，制定了系统动态重构蓝图，并设计了基于模态的嵌入式软件动态重构实施方法。基于软件架构的模态分析及其动态重构，提高了复杂嵌入式软件系统的可靠性、安全性和重用性。

参 考 文 献

[1] ARINC SPECIFICATION 653-2, Avionics Application Software Standard Interface[S]

[2] River W. VxWorks[EB/OL]. http://www.windriver.com/products/vxworks/, 2009

[3] Craveiro J. Integration of generic operating systems in partitioned architectures[D]. University of Lisbon, 2009

[4] 胡军. 构件化嵌入式软件设计的分析与验证[D]. 南京：南京大学，2005

[5] Perry D E. Software engineering and software architecture[C]// Feng Yu-lin, ed. Proceedings of the International Conference on Software: Theory and Practice. Beijing: Electronic Industry Press, 2000: 1-4

[6] Wang Geng, Zhou Xing-she, Dong Yun-wei. Studying on AADL-based Architecture Abstraction of Embedded Software[C]// The 8th IEEE International Conference on Embedded Computing. 2009

[7] Feiler P H, Gluch D P, Hudak J J. The Architecture Analysis & Design Language(AADL): An Introduction[M]. Carnegie Mellon University, 2006

[8] Rugina A-E, Kanoun K, Kaâniche M. A System Dependability Modeling Framework Using AADL and GSPNs[R]. LAAS-CNRS. 2007

[9] Zhang Chen-yu, Yang Zhi-yi, Dong Yun-wei. Research and Assessment of the Reliability of a Fault Tolerant Model Using AADL[R]. 2008 Advanced Software Engineering & Its Applications, 2008

[10] 刘建宾. 过程蓝图设计方法学[M]. 北京：科学出版社，2005