

嵌入式系统重构过程中功能构件提取技术研究

刘铁铭¹, 蒋烈辉¹, 张媛媛², 程三军³

(1 解放军信息工程大学 四院, 河南 郑州 450001; 2 72495 部队 通信科, 河南 郑州 450002;

3 河南省人民检察院, 河南 郑州 450004)

摘 要: 针对嵌入式系统重构过程中遇到的功能构件提取不准确、重用率低的情况, 提出了一种适合于嵌入式系统功能构件提取的工作流程. 建立了符合嵌入式系统特点的构件模型, 并给出了模型的形式化描述. 根据构件模型, 采用基于耦合度度量的聚类分析方法对固件代码的功能构件进行逐层实现和提取. 提出了基于模型验证的功能构件验证方法. 对比实验表明该方法极大地提高了嵌入式系统重构过程中功能构件提取的准确率, 并提升了原有系统的可重用性.

关键词: 嵌入式系统; 重构; 逆向工程; 功能构件; 软件体系结构; 耦合度

中图分类号: TP311.5

文献标识码: A

文章编号: 1000-7180(2016)05-0015-07

Research on Functional Component Extraction in Embedded System Reconstruction

LIU Tie-ming¹, JIANG Lie-hui¹, ZHANG Yuan-yuan², CHENG San-jun³

(1 The No. 4 College, PLA Information Engineering University, Zhengzhou 450001, China;

2 Automation Station, Unit 72495, Zhengzhou 450002, China;

3 People's Procuratorate of Henan Province, Zhengzhou 450004, China)

Abstract: Aimed at the inaccuracy and low reuse rate of functional component extraction in embedded systems reconstruction, a procedure suitable for embedded system functional component extraction is presented in this paper. A component model is established conforming to the characteristics of embedded systems, and a formal description of the model is given. According to the component model, firmware code functional component is implemented and extracted layer by layer using cluster analysis method based on coupling measurement. Functional component verification method is proposed based on model verification. By contrast experiment, this method greatly improves the accuracy of functional component extraction in embedded systems reconstruction, and enhances the reusability of original system.

Key words: embedded system; reconstruction; reverse engineering; function component; software architecture; coupling

1 引言

目前,随着电子与信息技术的飞速发展,嵌入式系统在各行业得到广泛的应用,人们希望能够通过嵌入式系统解决本行业复杂度更高、计算量更大的问题. 因此对嵌入式设备的要求也从原有的功能、开发

周期及成本等方面逐渐向高性能、稳定性及可靠性等方面转化,尤其是对应用在军事、航空航天、国家安全等特殊领域的嵌入式设备的安全性提出了更高的要求. 在某些特殊领域和敏感行业必须要确保国外引进的嵌入式设备的安全性. 在使用某些国外进口的核心嵌入式设备之前,一定要对其进行安全性检测. 与此

同时,针对嵌入式系统的后门发现、漏洞挖掘以及遗产系统的改进与升级等方面均需利用系统逆向工程的方法对关键核心的嵌入式设备进行功能验证、结构分析甚至是设计恢复与重新构建.本文中所述的重构(Reconstruction)与正向程序设计中的重构(Refactoring)概念不同.后者主要是通过调整正向程序设计代码来改善系统整体的安全性、稳定性、可靠性和可重用性等,使整个系统的体系结构更趋合理.而本文的重构主要是指在逆向工程领域通过对已有系统进行逆向分析,获取其组成构件,根据需求对其进行抽象、改造并重新构造新系统的过程.

笔者所在课题组前期已经对嵌入式系统中针对固件的压缩格式识别、操作系统类型识别、系统代码与用户代码的剥离等内容进行了深入的分析.针对嵌入式系统重构过程中软件功能构件的发现与提取,本文提出了一种模型化的方法.通过对嵌入式系统中的功能构件进行建模,归纳系统功能构件的特点、描述嵌入式系统功能构件的接口与属性,发现嵌入式设备软件功能的原始架构.然后分别通过耦合度与内聚强度的分析,划分功能构件间与构件内部的界限,为功能构件的提取及嵌入式系统的重构提供参考依据.

2 功能构件提取流程定义

从已有嵌入式设备的软件系统中提取出可重用的功能模块是功能构件获取的一种主要方法,也是嵌入式系统功能与结构重构领域的一个重要研究方向.嵌入式系统重构过程中,在对系统深入理解的基础上针对嵌入式软件的特点,本文提出了一种适合于嵌入式设备的软件功能构件的提取流程.该流程分为以下五个步骤:固件代码提取、固件代码预处理、功能构件模型建立、根据模型逐层实现、功能构件验证,如图1所示.其中,固件代码提取及代码预处理过程见文献[1-3].

该流程在固件代码预处理之后,首先根据嵌入式的特点对系统进行功能构件模型的建立,其次根据模型的结构和接口规范对预处理后的固件代码进行功能构件提取,然后根据构件的功能进行模块化验证,如果验证效果不理想则重新对提取规则进行调整,并再次进行功能构件提取,直至验证效果理想并进行最终入库.

3 功能构件模型

构件的设计是软件复用方法的核心,主要思想源自工业自动化过程中的产品部件.在软件开发过

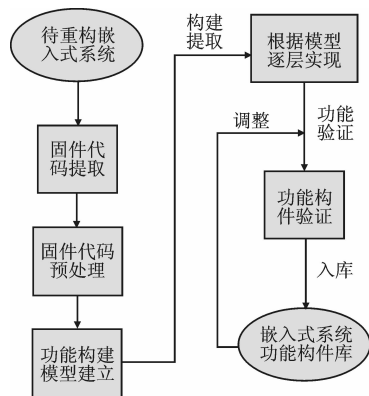


图1 嵌入式系统功能构件提取流程

程中最初使用构件化思想时,构件并没有一个清晰完整的定义,不同的研究目的和不同的应用领域,其功能构件的外延与具体内涵也有所差别.在逆向分析领域,针对嵌入式系统进行重构时所关注的构件与正向程序设计时构件所包含的内容就不尽相同.近年来,随着软件开发方法及软件逆向工程等技术的发展,产生出了一些新的关于软件功能构件的定义,其中比较有代表性的定义有以下几种.

(1)业内知名学者 Brown 与 Walinau 把软件构件定义为^[4-5]一个几乎独立的,非平凡的,可以替换的系统组成部分,它在定义完整的体系结构下实现某种特定功能,是一个与上下文无关的功能组合单元.

(2)国际上第一部软件构件专著的作者 Szyper-ski 把构件定义^[6]为一个可单独生产、获取和部署的二进制单元,具有规范接口和确定的上下文,它们之间可以相互作用工程一个功能系统,并能够被独立部署和被第三方组装.

(3)CMU/SEI 对软件构件的定义是^[7]:一个不透明的功能模块,可以被第三方使用,还要符合一个完整的功能模型.

嵌入式系统可裁剪性决定了其通常是由多个子系统构成,具有较高的构件化特征.而其嵌入式的需求与特性决定了嵌入式系统功能构件与通用的软件构件在接口、结构、性能、功能等方面会有很大的差异.本文的嵌入式构件是指实现了一定嵌入式系统功能的一组规范的、封装完整的、易进行二次重构的、具有嵌入式系统特性的二进制代码单元,其是组成嵌入式系统的功能组成单位.

3.1 构件模型定义

经过多年的研究,学术界在构件模型的研究领域取得了很多的成果.目前构件的主要代表模型有3C模型^[8]、SOFA、REBOOT、Fractal以及青鸟模型

等.在工业模型领域,以OMG的CORBA^[9]、SUN的J2EE/EJB^[10]、Microsoft的COM/DCOM^[11]为代表的构件模型也已经得到广泛应用和一致的认可.但是在嵌入式系统应用领域,由于嵌入式平台的多样性,以及嵌入式系统对实时性、内存消耗、稳定性及可靠性等因素的考虑,上述的常规构件模型不适用于嵌入式系统.特别是针对嵌入式设备进行系统逆向分析和逆向重构过程中,为了准确地划分和提取到系统中不同功能的构件,必须针对嵌入式系统的特点对构件模型进行定义.

3.1.1 基本原则

通过对嵌入式系统进行深入的研究,可认为一个构件模型应该满足以下基本原则.

(1)表达能力足够强.模型是对客观对象的抽象,合适的抽象层次十分重要,必须既能表现本质又不能过于细化,既要有足够的表达能力又不能过于复杂.

(2)简单性.这意味着构件模型要易于理解,它和表达能力强是一对矛盾,必须合理权衡.

(3)一致性和完备性.一致性是指构件模型作为一个独立的功能载体必须具有统一的描述接口、组成结构以及交互方式.完备性是指不存在构件模型所不能描述的功能构件.

(4)实用性.构件模型应方便分类、容易实现和检索.

(5)扩展性.在保持构件模型本身一致性和完备性的前提下,模型可以随着嵌入式系统功能的增强而变化.

3.1.2 构件的形式化描述

从普适性与严谨的角度出发,本文采用形式化方法对嵌入式系统功能构件模型进行描述.使用类似编程语言语法定义的形式化方法和语义表示来描述功能构件的接口规范以及行为逻辑,避免了图形符号或自然语言等非形式化描述方法在描述系统时产生的二义性.采用形式化描述方法不仅可以突出嵌入式系统中每个功能构件的功能角色和组成特性,同时还可以反映出某个特定功能构件在整个系统所处的位置,并且可以反映出整个嵌入式系统的框架结构.本文定义的嵌入式设备功能构件模型的形式化描述如下:

构件 ::= <构件的基本信息,构件的调用规约,构件的实现方法>;

构件的基本信息 ::= <构件的功能信息,构件的属性信息,构件的运行信息>;

构件的调用规约 ::= <构件的接口部分,构件的结构部分>;

构件的接口部分 ::= <该构件所提供的接口,该构件所要请求的接口>;

构件的结构部分 ::= <该构件所使用的交叉参考,该构件的内部流程>;

构件的实现方法 ::= <原子构件的实现,复合构件的实现>;

功能构件的接口部分分为该构件对外所提供的接口服务和该构件对外所要请求的接口.构件实现方法中的原子构件的实现,是指最基本的简单构件,而复合构件的实现则是原子构件经过组装而成的新的复杂构件.在嵌入式设备中,一个完整软件系统是由若干原子功能构件和复合功能功能构件组合而成的.因此对嵌入式系统功能构件进行形式化的递归描述,就可以对一个嵌入式软件系统的架构进行定义:

软件架构 ::= 构件 | <构件,软件架构>;

如果在功能构件进行组装时引入连接器(connector,指构件之间连接部分的实现),软件架构也可以按如下方式进行描述:

软件架构 ::= 构件 | <构件,连接器,软件架构>;

由于嵌入式系统的特殊性,在功能构件模型的描述中必须要引入对可靠性、实时性、资源限制性、安全性等非功能因素的描述,所以本文在功能构件的形式化描述中引入了针对嵌入式特性的说明,具体描述如下:

构件 ::= <构件的基本信息,构件的调用规约,构件的实现方法,构件的嵌入式特性>;

嵌入式特性是对功能构件的非功能性说明,其中包括了嵌入式功能构件的可靠性、实时性、资源限制性、安全性等因素的需求和约束条件.

3.1.3 模型的建立

嵌入式系统的软件实现是与硬件组成密不可分的.所以在功能构件模型设计时应尽可能考虑硬件组成和底层操作系统的因素,最大限度地满足在对嵌入式系统进行重构的过程中对功能构件的约束.另外,功能模型的设计也应该充分地考虑到嵌入式系统的可剪裁性及模块化等特性.

根据I Cube理论^[12],嵌入式系统功能构件被描述成三层结构.第一层为服务接口层,主要描述该功能构件对外提供给其他构件的服务接口;第二层为核心程序层,主要描述该功能构件对其他构件服务接口的需求;第三层是实现层,主要描述该功能构

件的具体行为逻辑. 在此基础上, 本文根据嵌入式系统的特性, 在设计功能构件模型时加入了配置层, 主要目的是使提取出的功能构件具备可剪裁性. 实现层也根据嵌入式特性被分为硬件抽象层和操作系统抽象两层. 硬件抽象层对不同硬件进行逻辑抽象; 操作系统抽象层对不同的嵌入式 OS 进行逻辑抽象. 本文的嵌入式功能构件模型的应用程序接口层和构件核心程序描述层与 I Cube 理论^[12] 相同. 嵌入式系统功能构件结构模型如图 2 所示.

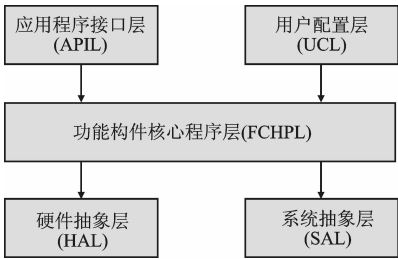


图 2 嵌入式功能构件结构模型

该嵌入式系统功能构件模型不仅具有分层结构设计的优点, 同时还拥有如下特点:

- (1) 针对底层, 屏蔽了不同 OS 和不同硬件的特性;
- (2) 可以通过对配置层进行特性化设置对抽取出来的功能构件进行裁剪, 适应不同的重构目的;
- (3) 定义了独立描述的应用程序接口层 (AP-IL), 符合嵌入式功能构件的工作特性.

3.2 构件属性选取

在固件代码预处理之后, 得到的就是系统源文件经过编译之后对应的二进制机器码文件. 经过反汇编及反编译处理, 可以得到对应的反汇编代码甚至是类 C 的伪代码. 为了能够按照上面定义的功能构件模型的要求提取出有效的构件实体, 需要对模型进行嵌入式特征的选取, 并根据这些特征在反汇编代码中的位置对固件代码进行划分, 从而有效地判别出固件的组成及其在整个系统架构中所处的位置.

通过对嵌入式系统的深入分析, 我们发现有些固件代码的属性可以影响到功能构件提取的最终结果. 本文将功能构件的子程序调用、被引频次、参数传递、返回值、API 调用、特殊指令调用等设定为可以影响固件提取的属性. 具体的属性及其含义如表 1 所示.

表 1 中定义的每个属性本质上就是一组数据结构, 用来记录与该功能构件相关的信息, 通过这些属

性值可以反应出一个功能构件的真实状态.

表 1 构件属性及其含义

属性名称	属性含义
子程序调用	记录构件内部共调用了多少个其他子程序及其调用处地址、目标构件、目标地址、调用次数等信息
被引频次	该构件在其他功能构件中被引用的次数、被引用的地址、调用构件信息
参数传递	该构件参数列表, 包括参数数量、类型及顺序
返回值	对于有多个返回点的构件, 记录其返回的地址
API 调用	记录构件内部共调用了多少个 API 及其名称、调用处地址、调用次数等信息
特殊指令调用	某些特殊用途的构件 (如加密功能构件) 往往会频繁且有规律的使用一些特殊指令 (如位移指令、异或指令、中断调用指令等), 记录其指令类型、调用频次等

下面以子程序调用属性为例说明该属性结构的定义.

```
typedef struct Node
{
    int SourceAddr;//记录调用处的地址
    int DestAddr;//记录本次调用的目的地址
    char[128] ComponetName;//本次调用目的功能构件的名字
    int ComponetID;//本次调用目的功能构件的 ID 号
    struct Node * Next;//指向本功能构件包含的下一个子程序调用的结构体
} CallAddressList;//子程序调用地址列表结构体定义

typedef struct Node
{
    int ComponetID;//该功能构件的 ID 号
    int SubCallingNum;//该功能构件包含的子程序调用个数
    CallAddressList * calladdrlist;//指向该功能构件的子程序调用地址列表
    struct Node * Next;//指向下一个功能构件的子程序调用实体
} SubroutineCalling;//功能构件子程序调用属性描述结构体
```

该定义为一个典型的十字链表结构, 如图 3 所示. 横向为待重构的嵌入式系统中每个功能构件的子程序调用属性描述结构体, 纵向为某个功能构件所包含的所有子程序调用的具体信息的描述.

4 模型的逐层实现

在庞大的固件代码中如何准确地划分并提取出功能构件是学术界一直探讨的热点及难点问题. 针对结构性很弱的反汇编代码, 本文采用基于耦合度

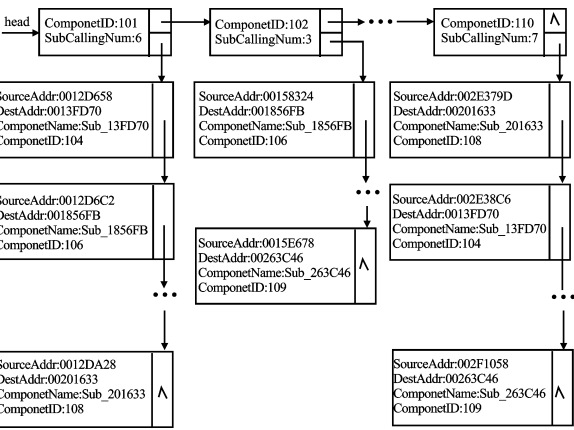


图 3 功能组件子程序调用属性数据结构示意图

度量的聚类分析方法在固件代码中进行功能构件的划分.

4.1 模块间耦合的定义

耦合是对各功能模块之间联系紧密程度的一种度量方法. 根据耦合度的定义,耦合可以大致被分为七种类型,具体如表 2 所示.

表 2 模块间耦合强度的划分

弱耦合关系 weak coupling relationship	非直接耦合关系	indirect coupling relationship
	数据耦合关系	data coupling relationship
	特征耦合关系	stamp coupling relationship
中度耦合关系 moderate coupling relationship	控制耦合	control coupling relationship
较强耦合关系 stronger coupling relationship	外部耦合	external coupling relationship
	公共耦合关系	common coupling relationship
强耦合关系 strongest coupling relationship	内容耦合	content coupling

如果两个功能模块相互之间没有任何的信息交换且属于同级模块不存在相互调用,那么这两个功能模块之间为非直接耦合关系(indirect coupling relationship),如图 4 中功能模块 A 与功能模块 B 之间的关系. 图 4 中模块 A 调用了功能模块 C 与功能模块 D, A 是通过参数列表的方式调用 C 与 D 的,并与 C 和 D 进行数据的传递. 如果功能模块 A 在调用功能模块 C 的过程中,数据传递的参数全部都是简单的数据类型,则 A 与 C 之间构成数据耦合关系(data coupling relationship);如果在功能模块 A 在调用功能模块 D 的过程中,数据传递的参数是相对复杂的数据结构,那么功能模块 A 与功能模块

D 之间构成特征耦合关系(stamp coupling relationship).

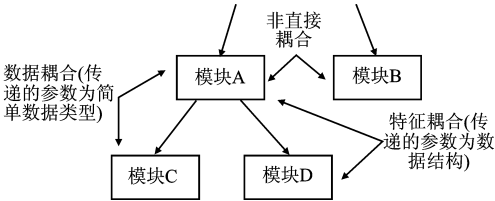


图 4 弱耦合关系(weak coupling relationship)示意图

控制耦合关系(control coupling relationship)属于中等强度耦合关系(moderate coupling relationship). 控制耦合关系在个功能模块间传递的不是简单数值类型的数据,而是对信号进行控制的信号量或开关值. 如果两个模块之间存在控制耦合关系,首先调用模块(calling module)需要把一个控制信号传递给被调模块(called module),被调用模块根据控制信号规定的含义来选择相应的操作. 在这种耦合关系的情况下,调用模块(calling module)必须要知道被调模块(called module)的内部组成结构或控制流程. 外部耦合关系(external coupling relationship)与公共耦合关系(common coupling relationship)属于较强耦合关系(stronger coupling relationship). 在一个嵌入式系统中,如果存在若干功能模块可以访问同一个全局变量(global variable),则这些功能模块之间就构成了外部耦合关系;如果这些功能模块可以访问同一个全局数据结构(global data structure),则这些功能模块之间就构成了公共耦合关系. 外部耦合关系与公共耦合关系的属于较强耦合度关系的原因如图 5 所示. 假定 module C、module D 和 module F 对全局数据结构 S 都具有读写权限. module C 从全局数据结构 S 中读出数据后启动 module D,对全局数据结构 S 进行处理. 如果 module D 存在 bug 则其处理的全局数据结构 S 的结果也会出错. 那么当 module F 在后续控制流程中再次处理数据结构 S 时就产生蔓延性错误. 虽然 module F 本身并不存在问题,但是整个的问题是在调用 module F 时才被发现的,而 module D 才是问题的症结所在,所以 module C、module D 和 module F 这些模块间的耦合度很高.

功能模块间最强的耦合关系称为内容耦合关系(content coupling relationship). 如图 6 所示,如果一个功能模块可以直接读取或修改另外一个功能模块中的数据,或者允许一个功能模块在执行

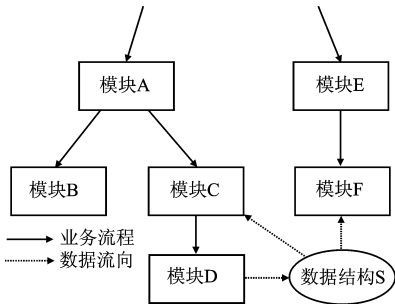


图 5 公共耦合关系示意图

控制流程是可以直接跳转到另外一个模块中去，称它们之间就构成了内容耦合关系。在内容耦合关系中，当后序功能模块进行修改时可能会影响到前序功能模块的执行结果，所以在针对嵌入式系统进行功能构件提取过程中尽量避免这类耦合关系的出现。如果发现功能模块间出现内容耦合关系，应尽可能调整验证模型，将其划归至一个构件中。

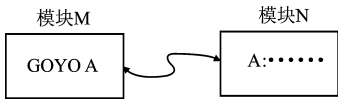


图 6 内容耦合关系示意图

4.2 耦合度度量模型的建立

在对耦合关系进行分类之后，还需要针对嵌入式系统的特点建立耦合度度量模型。嵌入式系统中功能模块之间主要是通过公共内存、全局变量、文件、消息及模块接口等方式进行数据通信的。通过这些方式进行数据通信时，比较容易产生功能模块间的耦合。根据功能模块间产生耦合度的类别及日常的工程经验，按照模块接口、公共内存、全局变量、文件、消息的顺序定义它们对耦合度的影响。为了能够更清晰直观地对耦合度关系进行量化，将耦合度关系最低的定义为 0，最高的定义为 1。功能模块与自身的耦合度被定义为 1；如果两个功能模块之间没有数据通信，则这两个功能模块之间的耦合度被定义为 0；其他则视情况在 0 到 1 之间进行取值。在耦合度度量时，必须为影响度量结果的因素赋不同的权值，才能计算出最终的耦合度度量结果。依据大量实验及实践经验，对上述几种信息传递的方式赋予如表 3 所示的权值。

假设有 m 个功能模块，那么这 m 个功能模块之间的所有耦合度关系值就可以用一个 $m \times m$ 的矩阵 R 来表示，即：

表 3 模块间耦合度度量权值分配表

信息通信方式	权值	取值范围
消息	0.2	0~0.2
文件	0.4	0.2~0.4
全局变量	0.6	0.4~0.6
公共内存	0.8	0.6~0.8
模块接口	1	0.8~1

$$R = \begin{bmatrix} r_{11}, r_{12}, \dots, r_{1m} \\ r_{21}, r_{22}, \dots, r_{2m} \\ \dots, \dots, \dots, \dots \\ r_{m1}, r_{m2}, \dots, r_{mm} \end{bmatrix}.$$

其中，

$$r_{ij} = \begin{cases} 1(i=j) \\ \frac{\sum_{l=1}^5 \sum_{k=1}^s C_{lk}^{ij} * m_{lk}^{ij}}{1 + \sum_{l=1}^5 \sum_{k=1}^s C_{lk}^{ij} * n_l^{ij}} (i \neq j) \end{cases}.$$

在表达式 r_{ij} 中， s 为第 L 个因素中的关系个数， n_l^{ij} 为第 L 个因素中的权重， m_{lk}^{ij} 为第 i 个功能模块与第 j 个功能模块之间的第 L 个因素中第 k 个关系的权重， C_{lk}^{ij} 为第 i 个功能模块与第 j 个功能模块之间的第 L 个因素中第 k 个关系的个数。

通过上述模型的定义，就可以计算出模块之间的耦合度。根据实际检测效果，把地址相邻且耦合度大于 0.75 的模块合并为一个构件是相对比较合理的。在进行构件划分之后，就可以按照图 2 中模型的定义，将不同功能的构件按层次分配至各个层中。

5 功能构件验证

为了保证功能构件提取的质量，在提取流程的最后，设计了一个功能构件验证的环节。本文采用目前比较流行的模型验证方法来对提取的构件及其属性进行验证。在嵌入系统功能构件提取过程时，功能构件间的信息交互和数据结构的多值性都将导致功能构件的状态空间爆炸问题。文献[13]中介绍的基于等价理论的缩减状态空间方法能够较好地解决该问题。通过等价理论(Equivalence Theory)的应用，可以将状态空间控制在一个可以接受的范围内。同时，文献[14-17]分别在正向程序设计和系统逆向分析领域对有关模型的验证进行了比较详细的描述。本文通过设计针对功能构件模型的验证的基本过程，提高了构件提取的质量和精度。在验证过程中如果发现错误或属

性值偏低等情况,则可以通过调整策略重新进行构件提取,直到达到满足重构需求为止.图 7 描述了功能构件模型验证的基本过程.

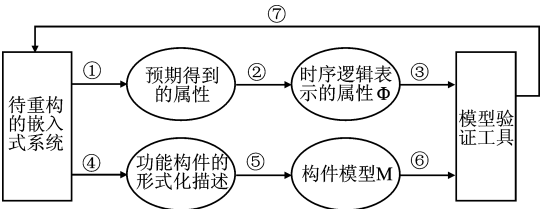


图 7 功能构件模型验证基本过程

图 7 中的过程说明如下:

- ①表示动态获取功能构件属性的过程;
- ②表示将功能构件属性转化为某种逻辑公式的过程;
- ③表示逻辑公式作为某种模型验证工具的输入;
- ④表示动态获取功能构件规约的过程;
- ⑤表示将功能构件规约转化为某种模型验证工具的输入;
- ⑥表示把构件模型的相关数据作为验证工具的输入;
- ⑦将模型验证的中间结果反馈给待重构的嵌入式系统,可以在重构过程中帮助系统发现错误并及时调整提取方法.

嵌入式系统功能构件模型验证过程可以重复循环地进行,直到提取出来的功能构件能够满足该系统重构所需要的要求为止.

此处的模型验证是基于 3.1 节定义的模型及 3.2 节定义的构件属性的,是一种自动化的属性验证方法.由于采用的是动态验证方法,所以功能构件属性是在系统运行过程中动态获取的.在嵌入式系统功能构件模型验证过程中,一般包括以下三个步骤.

- (1)嵌入式系统功能构件验证过程的系统建模:选择一种模型验证工具的描述语言动态模拟待验证的嵌入式系统,并利用工具构建功能构件模型 M;
- (2)属性规约:利用上述选定的模型验证工具形式化描述待验证的嵌入式系统及其功能构件的属性,并得到时序逻辑公式 Φ ;
- (3)模型验证:这是个动态的过程,通过输入功能构件模型 M 和时序逻辑公式 Φ 使系统运行,而选定的模型验证工具将会根据 M 和 Φ 得到错误或正确两种结果.过程(2)中说明了如果功能构件模型 M 是满足时序逻辑 Φ 的,那么 $M \models \Phi$;而过程(3)

则会产生系统运行的结果.这种在系统动态运行过程中得到的验证结果为重构过程中发现并改正构件提取错误提供了重要信息.

6 实验与结果分析

分别在 ARM、PPC 和 MIPS 体系结构下的嵌入式系统中与传统构件提取方法进行了对比.具体对比内容与结果如表 4 所示.由于功能构件提取数量的多少很大程度上依赖于待重构系统源码编写质量,所以为了保证固件代码的规范性,本次实验选取的固件代码都是知名厂家的固件代码.

表 4 不同体系结构之间提取结果的对比

体系结构	固件代码大小/kB	本方法提取构件数量	本方法提取构件可重用率/%	传统方法提取构件数量	传统方法提取构件可重用率/%
ARM	65 536	135	89	178	77
PPC	131 072	230	92	192	82
MIPS	32 768	63	90	81	78

在 ARM 和 MIPS 体系结构的构件提取时,本方法比传统方法提取出来的构件数量要少,原因是构件验证过程中通过调整,将耦合度较高的相邻模块进行了合并.在针对 PPC 体系结构的构件提取时,本方法提取的构件数量比传统方法提取出来的构件数量要多,主要原因是根据构件模型的分层定义,对各层代码进行了功能细化,将原有的大构件划分成了若干个功能独立、内聚性强的小构件.综合上表结论中可以看出,利用本文方法可以显著的提升提取构件的准确性和可重用率.

7 结束语

通过该构件模型的定义,可以在针对嵌入式系统进行重构的过程中提取出相对独立的功能构件.在保证功能单个构件对外描述不变的情况下可以对该功能构件进行修改及拆分等重构工作.针对多个功能构件进行重构时,在保证这些功能构件整体对外接口规范不变的前提下,可以对这些构件进行整合,从而节约系统成本,提升整体效率.

参考文献:

[1] 张平. 面向固件的嵌入式操作系统识别技术研究[D]. 郑州:解放军信息工程大学,2012.
[2] Brown A W, Wallnau K C. The current state of CBSE [J]. IEEE Software, 1998,15(5):37-46.

性、分布广泛性、准确性的特点. 在得到精确匹配点后,准确的估算出全局运动模型参数,将抖动视频帧配准成统一背景,计算量降低到原来的28%左右,对于后续的动目标检测打下坚实的基础.

参考文献:

- [1] 田伟刚,郭雷,黄雷. 一种应用于图像配准中的点特征匹配算法[J]. 微电子学与计算机, 2008, 25(3): 172-174.
- [2] Hellier P, Barillot C. Coupling dense and landmark-based approaches for non-rigid registration [R]. France, IRISA research report, PI 1368;30, 2000.
- [3] 胡觉晖,李一民,潘晓露. 改进的光流法用于车辆识别与跟踪[J]. 科学技术与工程, 2010, 10(23): 5814-5817.
- [4] Granharov V, Kleijn W B, Georgiev A. Sub-pixel registration of noisy images[C]//2006 IEEE International Conference on Acoustics, Speech and Signal Processing, CVonference Location. Toulouse, 2006, (2): 11.
- [5] Kybic J, Unser M. Bias minimizing filter design for gradient-based image registration[J]. Signal Process-

ing: Image Communication, 2005, 20(6): 554-568.

- [6] Chen J, Zou L H, Zhang J, et al. The comparison and application of comer detection algorithms[J]. Journal of Multimedia, 2009, 4(6): 435-441.
- [7] Yijian Pei, Hao Wu, Jiang Yu, et al. Effective image registration based on improved harris corner detection [C]// IEEE Trans. Image Process, 2010 International Conference on Information Networking and Automation (ICINA), Conference Location. China, Kunming, 2010; V1-93-V1-96.
- [8] Hathaway D H, Meyer P J. Video image stabilization and registration [P]. US 6560375B1, 20030506.
- [9] Davis J W, Sharma V. Background-subtraction in thermal imagery using contour saliency [J]. Int. J. Computer Vision, 2007, 21(2): 161-181.

作者简介:

刘珊珊 女, (1991-), 硕士研究生. 研究方向为视频处理, 图像配准. E-mail: xianghan@stu. xjtu. eud. cn.

李凡 男, (1982-), 博士, 副教授. 研究方向为视频处理与通信.

(上接第21页)

- [3] Szyperski C, gruntz D, Murer S. Component software: beyond object-oriented programming [M]. 3nd Ed. Berlin: Addison Wesley, 2012.
- [4] Bachmann F, Bass L, Buhrman C, et al. Technical concepts of component-based software engineering [R]. USA, Pittsburgh CA, Software Engineering Institute of Carregie Mellon University, 2013.
- [5] Latour, Wheeler T, Frakes B. Descriptive and prescriptive aspects of the 3Cs model: SETA1 working group summary[R]. New York, CASE Centre, Syracuse University, 2013.
- [6] Object Management Group. CORBA component model specification, version 3. 0[EB/OL]. [2015-08-10]. <http://www.omg.org>.
- [7] Java TM 2 Platform. Enterprise edition specification, version 1. 7 [EB/OL]. [2015-08-03]. <http://java.sun.com/j2eeldocs.html>.
- [8] 潘爱民. COM原理与应用[M]. 北京:清华大学出版社, 2002: 23-56.
- [9] Chang C K, Kim S. P: a petri-net based specification method for architectural components [C] // Proceed-

ings of the 23rd Annual International Computer Software and Application Conference USA, Washington, 1999: 396-402.

- [10] 郑志. 动态软件体系结构建模与模型验证方法的研究[D]. 大连:大连理工大学, 2010.
- [14] 徐春晓. 嵌入式软件构件提取与组装技术研究与应用[D]. 苏州:苏州大学, 2009.
- [15] 黄会群. 面向程序理解的转换系统研究[D]. 长沙:长沙理工大学, 2005.
- [16] 魏志广. 基于C语言的逆向工程的分析与实现[D]. 天津:河北工业大学, 2004.
- [17] 陈昊鹏. 软件逆向工程技术研究[D]. 西安:西北工业大学, 2001.

作者简介:

刘铁铭 男, (1977-), 博士研究生, 副教授. 研究方向为系统逆向工程、嵌入式系统. E-mail: fxliutm@163. com.

蒋烈辉 男, (1967-), 博士, 教授, 博士生导师. 研究方向为系统逆向工程、计算机体系结构.

张媛媛 女, (1979-), 硕士, 工程师. 研究方向为计算机应用技术.