文章编号:1671-4598(2020)01-0026-04 DOI:10.16526/j.cnki.11—4762/tp.2020.01.007 中图分类号:TP391 文献标识码:A

基于 VC33 航天软件集成测试技术研究

牛颖蓓,左 芸
(中国电子科技集团公司 第三十二研究所,上海 201808)

摘要:针对航天软件高可靠性的特点,以发现程序中的错误为重要目的;为了使得整个软件得到多次多层的测试,发现存在于单元间接口的诸多问题,验证单元接口的一致性、正确性而采用集成测试技术;集成测试方法主要包含非渐增式和渐增式两种测试方法;渐增式测试方法主要分为自顶向下、自底向上和三明治集成 3 种方法;根据航天型号软件极特殊的运行环境和特点,基于 VC33 航天型号软件采用深度优先自顶向下集成测试方法;实例验证,该方法能发现和改正模块接口错误,减少残留错误,降低航天软件开发风险与代价以及保证航天软件质量。

关键词:软件测试;集成测试;航天软件;自顶向下

Research on Inetgrated Testing Technology of Aerospace Software Based on VC33

Niu Yingbei, Zuo Yun
(The 32nd Research Institute of China Electronics Technology Group Corporation, Shanghai 201808, China)

Abstract: Aiming at the high reliability of aerospace software, it is important to find errors in the program. In order to make the whole software get multi-layer and multi-test, find many problems existing in the interface between units, the integrated testing technology is adopted to verify the consistency and correctness of the interface between units. Integration testing methods mainly include non-incremental and incremental testing methods. The incremental method can be divided into top-down, bottom-up and sandwich integration. According to the special operating environment and characteristics of the aerospace software, the depth-first top-down integrated test methods are adopted for the aerospace software based on VC33. Example show that this method can detect and correct module interface errors, reduce residual error, reduce the risk and cost of aerospace software development and ensure the quality of aerospace software.

Keywords: software test; integration testing; aerospace software; top-down

0 引言

航天工程,包括运载火箭、卫星、飞船和武器等型号都涉及巨大的投入、重要的使命乃至国家的安危[1]。特别是承担过程控制、系统指挥、数据处理等关键任务的软件,有大量是实时嵌入式软件。嵌入式控制器既需要快速地完成复杂控制算法的运算,又需要很高的控制精度。DSPVC33 由于具有高效的实时运算速度和强大的数据运算处理能力,越来越广泛地应用于航空航天领域。伴随着基于 DSPVC33 航天软件可靠性要求越来越高,基于 DSPVC33 航天软件的测试技术要求也越来越高。实时嵌入式软件给软件测试造成了很大的困难,使得软件测试技术和方法的运用非常艰难,有时甚至是不可行的,因此选择一种合适的测试技术和方法显得尤为重要。

随着国外芯片及传感器的智能化、网络化及微型化发展,使得采用虚拟仪器构建测试系统测试软件应运而生,目前我国的测试技术正逐步向智能化、网络化方向发展。

收稿日期:2019-03-12; **修回日期**:2019-07-30。

作者简介:牛颖蓓(1982-),女,河南南阳人,高级工程师,主要从事航天软件测试技术、信号处理技术方向的研究。

在软件测试中,为了使得整个软件得到多次多层的测试[2],发现存在于单元间接口的诸多问题,验证单元接口的一致性、正确性而采用集成测试技术,集成测试采取系统性的技术,这是一个渐进的过程,从一个单元开始,逐步把单元集成为部件,把部件和部件集成为软件配置项。在这个过程中,对接口的测试和对部件的测试交叉进行,从而使得整个软件得到多层次的测试。根据航天型号软件可靠性、安全性要求高,又有极特殊的运行环境,通过结合具体航天型号软件研制实践,形成基于 DSPVC33 航天型号软件适用的集成测试技术和方法。

1 集成测试方法

集成测试方法是进行单元模块组装的方法和步骤[3]。集成测试方法包含渐增式和非渐增式两大类,渐增式测试又可分为自顶向下、自底向上和三明治集成 3 种方法。

1.1 非渐增式

把几十甚至上百个单元联接在一起,先分散测试,再集中起来一次完成组合和测试,对各个模块测试结束为止,对整个程序进行组合时才能发现错误,需要构造驱动模块和桩模块。适用于在一个做得很好的、高内聚的设计中,

模块间的相互作用很小，而且十分小心地详尽地说明了接口，那么接口错误将可保持在最低限度。

以下面程序结构图说明非渐增式集成方法过程：

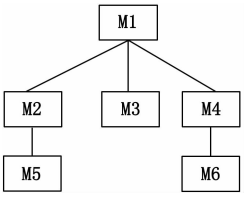

图 1　程序结构图

非渐增式以下列方式进行：首先对六个模块中的每一个执行单元测试，对单元的测试次序可以顺序地进行或平行地进行。最后把单元组合或结合成一个程序。根据单元在结构图的地位，对模块 M2 配备了驱动模块 D1 和桩模块 S1，对模块 M4 配备了驱动模块 D3 和桩模块 S2，对模块 M3、M5、M6 只配备了驱动模块 D2、D4、D5，对主模块 M1，配备了桩模块 S3、S4、S5。分别进行单元测试后，按结构图 1 形式联接起来，进行集成测试。

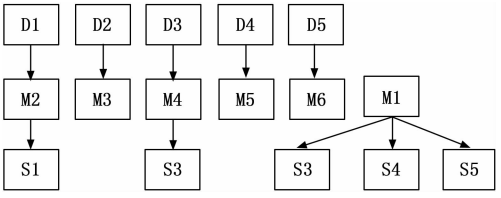

图 2　非渐增式集成测试

1.2　渐增式

渐增式测试不是单独地测试每个单元，而是首先把下一个要被测试的单元同已测试的单元集合组合起来，然后再测试。典型的分为自顶向下、自底向上和三明治集成 3 种。

1.2.1　自顶向下集成

自顶向下集成测试从主程序开始。被主程序调用的下层单元都作为桩函数出现，桩就是模拟被调用单元的一次性代码[4]。把附属主控模块的子模块孙模块单元等集成起来的方式有深度优先和广度优先两种。

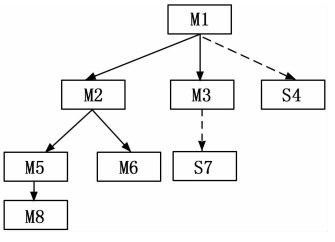

图 3　深度优先自顶向下测试

深度优先方法是先把结构中的一条主要的控制路径上的全部模块完全集成起来。主要路径的选择与特定的软件应用特性有关，可以尽量选取程序主要功能所涉及的路径。如图 3 所示，选择左侧路径，先集成 M1、M2 及 M5，下一步集成 M8，如果 M2 的某个功能需要也可以先集成 M6，然后在联结中间和右面的路径。

广度优先方法从结构的顶层开始逐层向下集成。把上一层模块直接调用的模块集成进去，然后对每一个新集成进去的模块再把其直接调用的模块集成进去。如图 3 所示，从 M1 出发，先集成 M2、M3 及桩模块 S4，接着是 M5、M6 这一层，以此类推。

自顶向下的集成过程为以下五个步骤：

1）用主控制模块做测试驱动模块，用桩模块代替所有直接被主控模块调用的模块。

2）根据所选择的集成方法（深度优先或广度优先）以及新模块的选择原则，每次用一个实际单元替代一个被调用的桩模块，并开发该单元可能需要的桩模块。

3）每组装一个新模块，测试一个。

4）完成一组测试后，用实际模块替换另一个桩模块；并为该实际模块开发必要的桩模块。

5）为了避免引入新的错误，再次复用以前使用过的测试用例进行测试，即重复以前执行过的部分或全部测试。

1.2.2　自底向上集成

自底向上集成测试是按照自顶向下顺序的镜像，不同的是，桩模块由模拟功能分解树上一层单元的驱动模块代替。在自底向上集成测试过程中，首先从分解树的叶开始，并用编写的驱动测试[5]。驱动模块单元中的一次性代码比桩模块单元中的少。大多数系统在接近叶时都有较高的扇出数，因此在自底向上集成测试过程中，不需要同样数量的驱动模块单元，不过代价是驱动模块单元比较复杂。

过程如图 4 所示，D1、D2、D3、D4 是模块集合 1、2、3、4 的驱动模块，集合 1、2、3 上属于 M2，去掉 D1、D2、D3 将这 3 个集合直接与 M2 接口；同样地，在集合 4 与 M3 接口前去掉 D4，M2 与 M3 最后与 M1 接口。

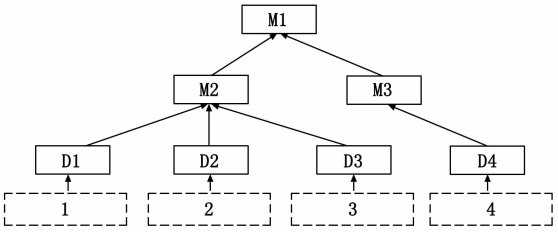

图 4　自底向上测试示意图

自底向上集成过程可用以下步骤实现：

1）把底层模块组装成实现特定软件子功能的集合。

2）为每个集合设计一个驱动模块单元，作为测试控制程序，明确测试用例的输入和输出。

3）对模块集合进行测试。

4）按结构自底向上的顺序，用真实模块替换驱动模块，将模块集合组装起来形成新的模块，再进行测试，直到全部完成。

1.2.3 三明治集成

三明治集成测试是自顶向下和自底向上两种集成测试的组合。如果通过分解树考虑三明治集成测试，则只需要在子树上进行大爆炸集成[6]。桩模块单元和驱动器模块单元的开发工作都比较小，不过作为大爆炸集成的后果，在一定程度上增加了定位缺陷的困难。对于有多个关键模块，从两头开始向中间进行，关键模块单元选用自底向上集成测试方法，需要考虑编写测试驱动模块和测试桩模块的工作量[7]。

除了大爆炸集成测试，基于分解的方法在直觉上很清楚，使用通过测试的组件构建。发现失效后，就可以怀疑最新加入的单元模块。集成测试方法很容易根据分解树追踪[8]。整个机制是根据结构集成单元模块，假设正确行为来自个体正确的单元模块和正确的接口。给定分解树所需集成测试会话数的计算公式如下：

$$\text{会话} = \text{节点} - \text{页} + \text{边} \tag{1}$$

对于自顶向下集成方法，需要开发（节点－1 个）桩模块；对于自底向上集成方法，需要开发（节点－叶子）驱动器模块[9]。

1.3 选取集成测试策略

单个模块分别都可以单独运行，可是这些模块集成在一起就不能正常工作，主要原因是因为这些模块在相互集成时接口间会引入新的问题。部分数据经过接口很可能丢弃；这个模块单元对那个模块单元可能造成不正常影响；几个子功能相互组合起来不能完成主要的功能；误差经过不断积累达到难以接受的程度；全局数据类型定义出现了数据覆盖之类的错误等。集成测试的目的是尽可能暴露单元测试时难以暴露的结构性错误。问题定位究竟是这个模块是否破坏了哪个模块的功能，或者是部分数据通过接口时是否丢失，或者是子功能组合起来难以实现主要功能，或者是误差是否进行反复多次积累等。集成测试按设计要求把通过的各个单元测试的模块集成在一起进行测试，以便发现与内部接口相关联的各种错误。

进行集成测试需要确定关键单元模块，对这些关键单元模块进行集成测试。关键单元模块应具有以下特征之一：1）满足软件功能需求；2）有清晰明确定义的指标要求；3）具备控制决策功能；4）较复杂、较容易产生错误的模块；

集成测试时需要考虑如下要素：1）使用哪一种集成测试方法来进行；2）是否需要特别的软硬件设备和环境在测试过程中；3）各个单元模块集成测试的顺序；4）集成测试的顺序是否与模块代码的设计编写进度一致；

进行集成测试需要使用集成测试用例，进行一次集成测试，就需要生成一系列集成测试用例。测试的重点是设计有效的测试用例。由于完全测试具有不确定性，因此发现尽可能多的错误使用有限的测试用例就显得非常重要。设计和执行测试用例需要遵守下面几个重要原则：

1）设计测试用例要明确输入数据应同时明确程序的预期输出；2）设计测试用例应该系统地科学地进行，不能随便设计；3）测试用例输入需要包含不合理的数据和合理的数据；4）不仅要检查程序是否做了不应该做的事情，而且还要检查程序是否做了该做的事情；5）记录合理的测试用例，为了以后复用；

非增量式集成测试的弊端是对每个错误的定位和纠正特别困难。一次集成可能发现一大堆错误，有可能在改正一个错误的同时引入另一个错误，新旧错误混杂，容易出现混乱。定位出错的原因和位置更加困难。

自顶向下集成的弊端是需要建立桩模块。使桩模块模拟实际子单元模块的功能有些困难，因为桩模块在接收了所测模块发送的指令后需要按照它所代替的实际子模块返回相应的值，这加强建立桩模块的难度，导致增添额外的桩模块测试。并且输入/输出单元模块和较复杂算法大都在底层，这是很容易有问题的模块单元，一旦有这些问题，将导致无数次回归测试。而自顶向下集成测试方法的好处是能比较早地对发现问题。选用采用深度优先自顶向下集成或是广度优先自顶向下集成，是根据软件部件的特点进行决定。深度优先自顶向下集成是把主控制路径上的模块单元集成在一起，选择哪一条路径作为主控制路径一般根据问题的特性确定。广度优先自顶向下集成方法是沿控制层次结构水平地向下移动。

自底向上集成方法的弊端是直到最后一个模块加上去后程序才形成一个实体。作为一个实体存在程序一直不能，在自底向上集成和测试的过程中，最后才能接触到对主要部分的控制。这种方式的好处是不需要建立桩模块，需要建立驱动模块单元，相对来说建立驱动模块单元一般比建立桩模块单元容易，由于涉及到输入/输出的模块单元和较复杂算法是最先进行集成，这样能尽早地测试容易出问题的部分。此外自底向上集成测试可以提高测试效率，同时进行多个单元模块集成测试。

由于航天型号软件大多有相当部分复杂的科学计算或数据处理，领域应用背景强，不易编写桩模块，故而尽量不采用需编写桩模块的办法。因此不采用非渐增式方法集成。航天型号软件算法复杂，可靠性、安全性要求高，其中故障处理路径、降级处理路径、错误处理路径、最长执行时间路径等都是重要的功能、性能关键路径，这些重要特殊的路径需要高强度仔细的测试，因此控制输入数据，测试特殊路径的能力很重要。所以不采用渐增式方法的三明治集成测试方法和自底向上集成测试方法。深度优先和广度优先自顶向下集成方法的主要区别在于组装次序，根据航天软件关键模块功能高内聚的特点，采用深度优先自顶向下集成方法，不采用广度优先自顶向下集成方法。

2　实例验证

根据航天型号极特殊的运行环境和特点，基于 VC33 航天型号软件选择采用深度优先自顶向下集成测试方法。以通讯控制软件初始化功能为例，应用该方法如下。

1）根据该软件设计，通讯控制软件划初始化功能主要由 4 个部件 process、inithard、selftest、initvar 组成，其中 inithard 部件由 start _ timer、delay _ us、init _ Gpio、initXint、setTimer、aceInitBRT、initTimer、InitTable、initFlash、memCopy、aceSetmode、acewait、acesetTllcmd、acegetRTaddr、aceCtrl _ WD、acesoftReset、initBRT、aceGetSubadd 共 18 个函数组成。初始化模块划分图如下：

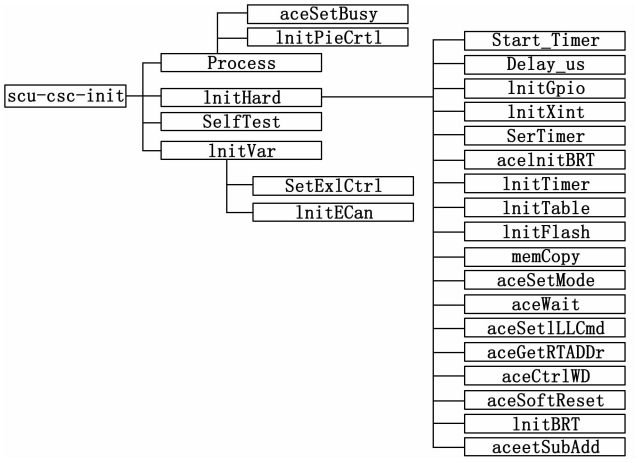

图 5　软件模块划分图

2）使用测试工具 Testbed 进行分解集成部件，SCU－CSC－init 部件采用深度优先“自顶向下”的集成测试方法，由于 SCU－CSC－init 部件调用关系比较复杂，采用按分支添加函数进行集成测试，当被调用函数有多个上层调用函数时，分别考虑单元接口调用情况，从而达到遍历的目的，对尚未添加函数人为进行插桩处理，从而明确函数调用关系，同时根据函数的正常情况和异常情况并考虑边界情况进行综合设计用例。通讯控制软件初始化模块路径流程图如图 6 所示。

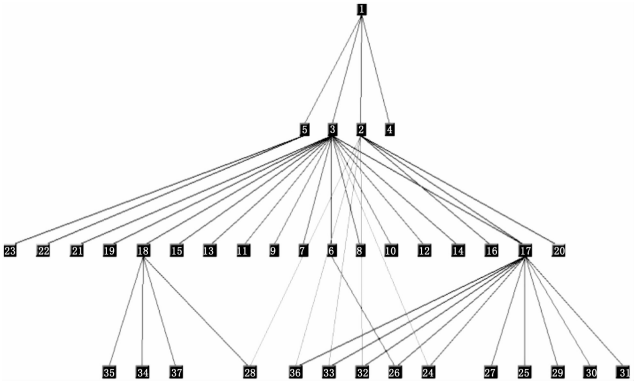

图 6　路径流程图

3）在函数 InitHard 调用函数 MemCopy 时，选择基于路径测试方法查看语句覆盖率的覆盖情况，语句覆盖率 S 未达到 100%，如果出现未覆盖的函数 MemCopy（struct IMU _ Static _ Model * ps _ IMU _ Static _ M _ i，struct Sum _ Model _ IMU _ Diagnose * ps _ sum _ M），分析没有完全覆盖的原因是因为部分指向寄存器地址的指针使用引起，不存在软件缺陷。通讯控制软件初始化模块过程基于路径语句覆盖率如图 7 所示。

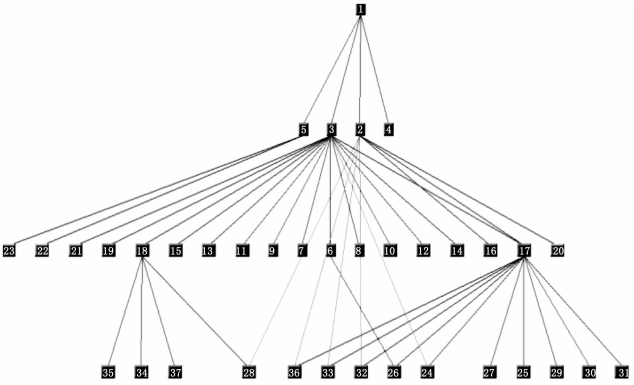

图 7　基于路径语句覆盖率

3　结束语

根据航天型号极特殊的运行环境和特点，基于 VC33 航天型号软件选择采用深度优先自顶向下集成测试方法。以指导工程测试人员针对高可靠性航天软件进行测试方法的选择。由实验结果可知，航天软件集成测试采用深度优先自顶向下集成测试方法，可以准确地反映出被测软件存在于单元间接口的诸多问题，使得整个软件得到多层次的测试，反映出软件可靠和缺陷的关系。通过实践验证该方法的可行性与有效性。同时，从新一代运载火箭飞行控制软件实验的分析结果也可以看出，利用本文方法选取的新一代运载火箭飞行控制软件测试方法进行集成测试，能够得到比较满意的系统高可靠性和质量保证。

参考文献：

[1] 宿　斯. 软件项目管理［M］. 北京：机械工业出版社，2010.
[2] 利用 Sping 来进行集成测试［Z］. 上海：上海泽众软件科技有限公司，2013.
[3] 周元哲. 软件测试［M］. 北京：清华大学出版社，2013.
[4] 新语译. 有限软件测试［M］. 北京：清华大学出版社，2003.
[5] 周元哲. 软件测试基础［M］. 西安：西安电子科技大学出版社，2011.
[6] 王丹丹. 软件测试方法和技术实践教程［M］. 北京：清华大学出版社，2017.
[7] 陈建潮. 软件测试方法与技术［M］. 北京：中国铁道出版社，2018.
[8] 蔡立志. 软件测试导论［M］. 北京：清华大学出版社，2016.
[9] 朱少民. 全程软件测试［M］. 北京：人民邮电出版社，2019.