http://myfj.cnjournals.com myfj_sadri@163.com (021)20866796

DOI: 10.19416/j.cnki.1674-9804.2024.01.024

一种基于耦合关系搜索的更改影响评估方法

陈 刚* 王文升

(中国航空工业集团公司第一飞机设计研究院,西安 710089)

摘 要: 问题报告和更改控制是 DO-178C 对民用飞机机载软件配置管理过程的基本要求,而更改影响分析则一直是机载软件更改控制过程中的一项难点问题。对更改影响分析的不彻底可能造成受影响软件部件的不正常运行,对军用飞机和民用飞机机载设备安全性的影响不容忽略。为了快速、有效地定位机载软件更改后对其它软件单元和相关软件生命周期资料的影响,使用有向图将软件单元之间的耦合关系进行抽象,并针对有向图的节点和边设计相关数据结构进行数字化存储,使用逆向搜索的方式对有向图数据进行遍历搜索,快速量化软件原始更改所带来的潜在影响。

关键词: 耦合;依赖关系;更改影响分析;DO-178C;软件单元

中图分类号: TP311.5 **文献标识码**: A **OSID**:

0 引言

机载软件由于其复杂的使用环境、不断改进的需求和技术要求以及法规的变化等等,使其软件更改在系统维护期间持续发生。民用飞机机载软件适航标准 RTCA DO-178C 对软件提出了更改控制要求[1],其中更改影响分析(change impact analysis,简称 CIA)环节要求能够确定软件更改影响到的所有软件生命周期数据,并据此对软件的不同数据进行针对性修改,以避免更改不完整带来的软件缺陷。军用飞机及其他安全相关行业也对软件的更改影响分析有较高的要求,但在实践过程中,由于更改影响分析不完善带来的软件问题依然难以避免。

针对该问题,RTCA DO-178C 给出了一种基于追踪关系的软件更改影响分析方法,该方法利用软件生命周期数据之间的追踪关系可以确定与当前软件更改内容相关的所有纵向生命周期资料,但对软件单元(如函数、模块或组件)之间的相互影响却并未覆盖。

对于机载系统而言,除了关注软件更改本身所涉及的生命周期数据的同步更改,更关键的是找出与被更改软件单元相关联的其他软件单元或其它相关系统的软件配置项。针对这种软件单元或软件配置项之间的更改影响分析,目前常见方法如下:

1) 基于模型的影响分析方法[2-4]。该方法使用基于模型的方法对软件需求和软件设计单元之间的关联关系进行建模,更适用于前期采用模型进行需求捕获或软件设计的情况。

2) 基于语法依赖的影响分析方法[5-7]。该方法以面向对象语言的语法特点为出发点,基于封装、继承等特性分析类、对象之间的依赖关系,适用于使用面向对象语言的情况。

3) 基于动态执行行为的影响分析方法[8]。该方法以程序执行过程中的运行行为为分析对象,使用插桩方式获取所需的执行关系,从而判断与更改软件单元有动态调用关系的受影响单元。

本文以机载软件为对象,将 DO-178C 中要求的追踪关系与软件单元之间的耦合关系相结合,最终划定软件更改的影响域,提高了软件影响分析的彻底性。

* 通信作者. E-mail: lankai1716@163.com

基金: 航空科学基金(2017ZG03023),基于 DO-178C 的机载软件供应商管理办法

引用格式: 陈刚,王文升. 一种基于耦合关系搜索的更改影响评估方法[J]. 民用飞机设计与研究,2024(1):151-156. CHEN G, WANG W S. A change impact assessment method based on coupling searching[J]. Civil Aircraft Design and Research, 2024(1):151-156(in Chinese).

1 基于耦合关系分析的更改影响评估算法

1.1 基本概念定义

有向图使用节点和有向边构成模型来记录复杂系统变量之间的影响关系,在图像处理[9]、故障搜索[10]方面均有较多的应用。本文使用有向图作为耦合关系的载体,使用到的相关概念定义如下:

1) 将进行更改影响分析之前,为解决软件生命周期过程中发现的软件缺陷或错误而计划更改的目标软件单元的集合定义为软件原始更改集,使用字母 Minit 表示。

2) 将更改影响分析之后,确定的所有需更改的软件单元的集合定义为软件更改集,使用字母 M 表示,则 Minit ∈ M。

3) 定义更改有向图 Gm = (M, R),用于表示软件更改集 M 中所有软件更改单元及其之间的耦合依赖关系。在图中,任意节点 Mi 表示一个需更改的软件单元,任意边 Ri 表示有向关系〈Mi, Mj〉,表示软件单元 Mi 由于使用了 Mj 提供的服务或使用了 Mj 修改后的数据而受到影响。推论可得:

设受影响集合 Maffect = M - Minit,则:

∀ Mi ∈ Maffect, ∃ Mj ∈ Minit,使得〈Mi, Mj〉∈ R

定义 4:定义基本有向图 G = (V, E),用于表示软件更改之前,根据软件架构建立的所有软件单元集合 V 和软件单元之间的耦合依赖关系集合 E。则由定义可知 M ∈ V, R ∈ E,更改有向图 Gm 为基本有向图 G 的子图。

更改影响评估是以软件原始更改集为输入,从基本有向图中搜索确认更改有向图,并最终获取软件更改集的过程。

1.2 建立基本有向图

软件的耦合关系指的是软件不同单元之间的依赖关系[11],主要体现在以下几个方面:

1) 软件单元之间的调用关系;

2) 软件单元间的参数传递;

3) 多于一个组件使用的全局变量;

4) 软件执行过程被中断服务程序中断而产生的依赖关系[12],主要对实时性要求较高的软件单元可能产生影响。

在为耦合关系建立基本有向图之前,可以通过以下几个活动大幅降低软件单元之间的耦合程度,降低更改影响评估的复杂度。

1) 标准约束:在标准文件中增加设计约束,可以大幅降低软件的耦合程度,由此提高软件更改影响分析的确定程度。可以考虑的因素包括但不限于:

a) 限制全局变量的使用。如图 1 所示,软件单元与全局变量之间的读写关系一般包括一写一读、多写一读、一写多读、多写多读[13]等。不用或者少用全局变量,能够减少不同软件单元读写同一全局变量带来的对软件代码执行结果的间接影响;如果确实需要使用全局变量,应至少将读写关系限制在一写一读、一写多读以内,禁止多写操作,同时在软件设计文档的数据字典说明中对全局变量的名称、用途和不同函数的使用关系进行详细描述,保证后续分析工作的正确执行。

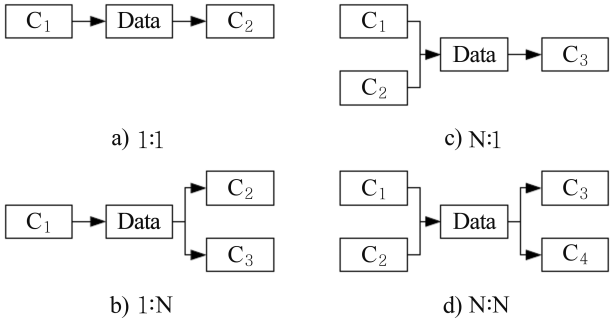

图 1 四种数据依赖类型

b) 限制同一变量在多处赋值的情况。同一变量被多个函数赋值,对不同函数间的执行顺序有较高的要求,一旦读写顺序发生变化,变量的赋值结果会发生不可预估的变化,属于执行次序的强耦合关系,应限制使用,降低更改影响评估的复杂度。

c) 限制直接跳转语句的使用。直接跳转语句可能会打破原有函数间的执行次序,对执行次序的强耦合关系造成破坏(如上一条的赋值次序问题),从而使软件的执行过程出错或产生潜在漏洞。

d) 限制中断嵌套。多优先级的软件中断嵌套在软件实际执行过程中会发生不确定的运行时嵌套,在软件运行前很难穷尽分析所有可能的嵌套关系,因此一旦中断服务程序的内容受到更改

的影响,则该影响可能会被级联扩大,使软件的更改影响分析不能丧失预见性,导致潜在缺陷的产生。

2) 软件架构设计:在软件架构设计时,明确组件间的接口、调用关系、调度策略、中断处理逻辑等。必要时采用控制流图、数据流图、时序图等图表方式表达;详尽的软件架构设计数据可以作为基本有向图建立的基础。

作为更改影响评估的基础,软件基本有向图的建立是不可缺少的,本文按照以下步骤建立基本有向图:

1) 确定软件单元粒度。对于机载软件(不考虑面向对象语言)来说,软件单元根据实际情况可以按照软件配置项、软件部件或软件函数的粒度进行划分。由于基于耦合关系的更改影响分析更多地注重软件单元外部的影响,因此对软件单元内部的更改后回归测试需使用所有相关测试用例进行重新测试。一般建议以函数为软件单元进行基本有向图的分析建立。

2) 确定软件单元集合 V。根据软件设计文档或软件源代码,确定软件中涉及的所有软件单元(如软件函数)的集合。

3) 确定软件单元之间的调用耦合关系。根据软件的控制流图或源代码,确定软件单元之间的每一个调用关系 Ei,并根据调用与被调用关系确定有向图 Ei 的方向。

4) 确定软件单元之间的数据耦合关系。根据软件数据字典或源代码,确认软件单元外部的所有数据列表,并遍历该数据列表,确认软件单元之间的读写影响关系 Ej,并使读取该数据的软件单元指向所有写入该数据的软件单元。

5) 设计数据结构存储有向图关系数据,数据结构形式如下:

```
Struct Edge {
    String SourceNode;
    String DestNode;
}
```

则 Edge Ei={Vi,Vj}表示一条由 Vi 指向 Vj 的边。以结构数组的形式可以存储整个有向图中多个边的信息。

以以下代码段为例,根据以上方法建立有向图,如图 2 所示。

```
static int x:
static int y:
void main()
{
    init ():
    x=x+1;
    v1():
}
void init ()
{
    x= 0:
    y= 0:
}
void v1()
{
    v2():
}
void v2()
{
    if (x==1) v3():
}
void v3()
{
    y= y+1;
}
```

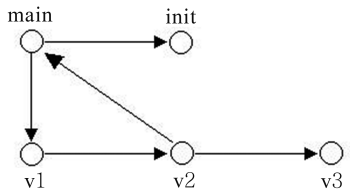

图 2　软件代码及其基本有向图

由该基本有向图产生的数据库存储数据如表 1 所示。

表 1　示例有向图存储表

序号	SourceNode	DestNode
1	main	init
2	main	v1
3	v1	v2
4	v2	main
5	v2	v3

1.3　计算生成更改有向图和更改集

对于基本有向图 G,在已知原始更改集 Minit 的情况下可知:

$\forall Mj \in Minit$,若 $\exists <Ei,Mj> \in E$,则

$$Ei \in Maffect$$

由节点 EiMj 和有向边〈EiMj〉构成的子图是更改有向图 Gm 的一部分。

因此根据更改需求确定 Minit 后,对 Minit 进行遍历,$\forall Mj \in Minit$ 使用逆向搜索算法找到有向边对应的所有受影响节点集合 Maffect,并与原始更改集合并形成最终的软件更改集 M。算法以伪代码描述如下:

```
list M = Minit
for each Mi in M
list Edgelist=getData("DestNode=Mi")
  if Edgelist. count( ) 〈〉 0
    for each edge in Edgelist
```

```
        if (! M.exist(edge.SourceNode))
            M.add(edge.SourceNode)
        End each
    End if
End each
```

算法输出节点集合 M 为软件更改集。

根据以上算法，假设图 2 中 main 函数需要更改，则经过影响分析后软件更改集 M 为(main，v2，v1)，算法演进过程如图 3 所示，图中●表示软件更改集 M 中的节点，未加入该集合之前显示为○。图 3(c)中黑色填充的节点和节点间的边构成了最终的更改有向图 Gm。

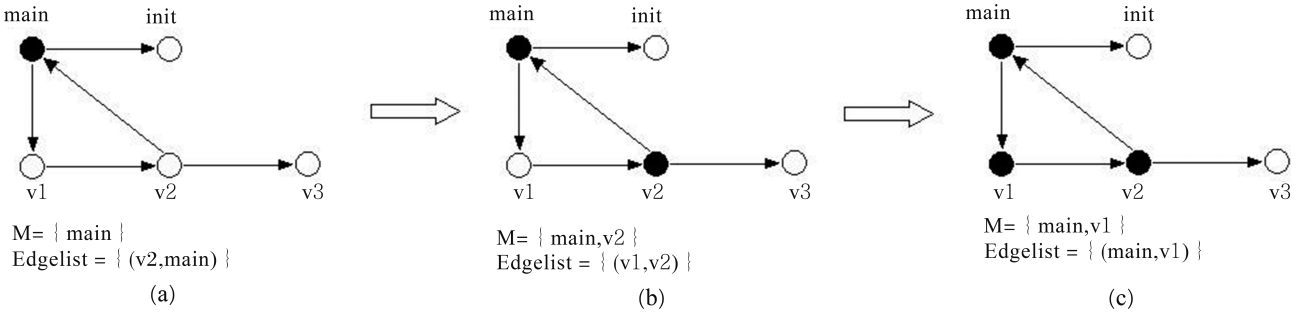

图 3　有向图逆向搜索算法过程演进图

图 3(a)项中，将 M 初始化为假定需要更改的函数 main，根据条件“DestNode = main”在表 1 所示库中搜索，得到有向边集合 Edgelist = {(v2，main)}，将节点 v2 加入集合 M；

图 3(b)项开始时，M = {main，v2}，遍历至 M 的第二项，即 v2。根据条件“DestNode = v2”在表 1 所示库中搜索，得到有向边集合 Edgelist = {(v1，v2)}，将节点 v1 加入集合 M；

图 3(c)项开始时，M = {main，v2，v1}，遍历至 M 的第三项，即 v1。根据条件“DestNode = v1”在表 1 所示库中搜索，得到有向边集合 Edgelist = {(main，v1)}，没有需要加入 M 的新节点，M 此时遍历结束。

2　受影响软件生命周期资料的确认

上节所示算法给出的更改影响结果 M 为受影响软件单元的集合，属于软件源代码的一部分，但是按照 DO-178C 对民用飞机机载软件的要求，软件的更改应同步到所有受影响的软件生命周期资料。

本文以 DO-178C 中要求的软件追踪关系作为分析基础，以软件更改集 M 中涉及的每一项软件单元 Mi 为对象，进行生命周期数据的纵向拓展分析。

如图 4 所示为 DO-178C 中要求的软件生命周期数据的追踪关系。

1) 建立软件追踪关系，具体活动包括：

a) 软件需求分析阶段：建立软件高层需求与系统需求之间的双向追踪关系，建立追踪关系时应同时检查软件高层需求与系统需求之间的一致性和软件高层需求对系统需求的满足性，如有部分需求无法追踪至系统需求，应标识为派生需求，并提交安全性人员分析这些需求对系统安全性目标的影响。

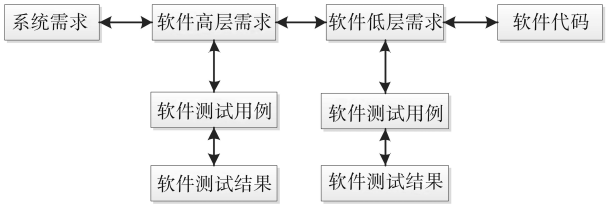

图 4　软件生命周期资料的追踪关系

b) 软件设计阶段：建立软件低层需求与软件高层需求之间的双向追踪关系，建立追踪关系时应同时检查软件低层需求与软件高层需求之间的一致性和软件低层需求对软件高层需求的满足性，如有部分需求无法追踪至高层需求，应标识为派生需求，并提交安全性人员分析这些需求对系统安全性目标的影响。

c) 软件编码阶段：建立软件源代码与软件低层需求之间的双向追踪关系，同时不允许有无法追踪的代码产生。

d) 软件测试阶段：分别针对软件高层需求和软件低层需求编写测试用例，并在测试用例执行完成后建立软件高/低层需求与软件测试用例、软件

测试用例与软件测试结果之间的双向追踪关系。

e）追踪关系的阶段评审:在软件需求评审、软件设计评审和软件编码的同行评审活动开展时,应同时针对该阶段产生的追踪关系进行评审,确保追踪关系的正确性。

根据追踪关系,确定与软件更改集 M 中每一项软件单元代码 Mi 有追踪关系的软件低层需求、高层需求以及相关测试用例,并根据更改内容实施这些相关资料的同步更改。

3　结论

软件更改影响分析向来是软件更改控制过程的难点问题,影响分析的不彻底也给航空和其它高安全领域的软件带来很多潜在的不安全因素。另外,精确的更改影响分析方法在软件开发阶段也能起到很好的缺陷预防作用[14]。目前的更改影响分析过程主要依靠分析人员的经验,本文提出的方法把代码间的横向耦合关系和生命周期数据之间的纵向追踪关系相结合,提供了一种基于有向图的软件更改影响分析搜索算法,并给出了算法的基本数据存储结构和搜索逻辑。为软件更改过程的精确控制和彻底验证奠定了基础,对机载软件和其它高安全领域的软件更改过程有很高的参考价值。

参考文献:

[1] Radio Technical Commission for Aeronautics. Software considerations in airborne systems and equipment certification:RTCA/DO-178C[S]. [S.l.]:RTCA, 2011.

[2] 陶传奇, 李必信, GAO,等. 基于模型的构件软件修改影响分析[J]. 软件学报, 2013, 24(5): 942-960.

[3] BRIAND L C,LABICHE Y, O'SULLIVAN L, et al. Automated impact analysis of UML models[J]. Journal of Systems and Software, 2006, 79(3): 339-352.

[4] 王映辉, 王立福. 软件体系结构演化模型[J]. 电子学报, 2005, 3(8): 1381-1386.

[5] 高灿, 侯秀萍, 孙士明. 基于抽象语法树的修改影响分析方法[J]. 长春工业大学学报(自然科学版), 2012, 33(4): 387-390.

[6] 杨鹤标, 陈小强. 对象依赖关系在变更影响分析中的应用[J]. 软件导刊, 2018, 17(10): 153-156,160.

[7] 孙小兵, 李必信, 陶传奇. 基于 LoCMD 的软件修改分析技术[J]. 软件学报, 2012, 23(6): 1368-1381.

[8] 刘震, 缪力. 基于动态调用图的 Java 程序修改影响分析技术[J]. 湖南师范大学自然科学学报, 2011, 34(6): 26-30.

[9] 崔宾阁, 孟翱翔. 基于最近邻有向图的遥感图像快速分割算法[J]. 计算机科学, 2013, 40(10): 274-278.

[10] 杨帆, 萧德云. 控制系统的 SDG 模型描述及故障传播分析[J]. 控制与决策, 2009, 24(7): 1001-1006.

[11] RTCA. Supporting Information for DO-178C and DO-278A:RTCA/DO-248[S]. [S.l:s.n.], 2011.

[12] 张欢. 基于并发程序切片的修改影响分析[D]. 南京: 东南大学, 2017.

[13] Certification Authorities Software Team(CAST). Clarification of structural coverage analyses of data coupling and control coupling:Position Paper CAST-19[R]. [S.l.:s.n.],2004.

[14] 谈晶晶. 修改影响分析技术在 C 程序的缺陷预防中的应用研究[D]. 南京: 南京大学, 2011.

作者简介

陈　刚　男,硕士,高级工程师。主要研究方向:软件工程过程及机载软件适航取证技术。E-mail: lankai1716@163.com

王文升　男,硕士,高级工程师。主要研究方向:软件工程管理。E-mail: 13488181331@163.com

A change impact assessment method based on coupling searching

CHEN Gang * WANG Wensheng

(AVIC First Aircraft Institute, Xi'an 710089, China)

Abstract: Problem reports and change control are two basic requirements of DO-178C for the configuration management activities of civil aircraft airborne software, and change impact analysis has always been a difficult issue in airborne software change control procedure. Incomplete analysis of the impact of changes may cause abnormal operation of affected software components, and the impact on the safety of military and civilian aircraft onboard equipment cannot be ignored. In order to quickly and effectively allocate the change impacts on other software units and on related software life cycle data, this article uses directed graphs to abstract the coupling relation between software units, and digitalizes and stores data structures related to vertexes and edges design in directed graphs. Reverse search is used to traverse the directed graph data, quickly quantifying the potential impact of original software changes.

Keywords: coupling; dependency; change impact analysis; DO-178C; software unit

* Corresponding author. E-mail: lankai1716@163.com