



(12) 发明专利

(10) 授权公告号 CN 112181800 B

(45) 授权公告日 2025. 04. 08

(21) 申请号 202010636930.1

(22) 申请日 2020.07.03

(65) 同一申请的已公布的文献号
申请公布号 CN 112181800 A

(43) 申请公布日 2021.01.05

(30) 优先权数据
10-2019-0081419 2019.07.05 KR

(73) 专利权人 现代自动车株式会社
地址 韩国首尔
专利权人 起亚自动车株式会社
梨花女子大学校产学协力团

(72) 发明人 张训 蔡贤娥 崔炳珠

(74) 专利代理机构 北京路浩知识产权代理有限公司 11002

专利代理师 崔龙铉 赵赫

(51) Int.Cl.
G06F 11/3668 (2025.01)

(56) 对比文件
US 2014351797 A1, 2014.11.27
US 2019050308 A1, 2019.02.14

审查员 李婧

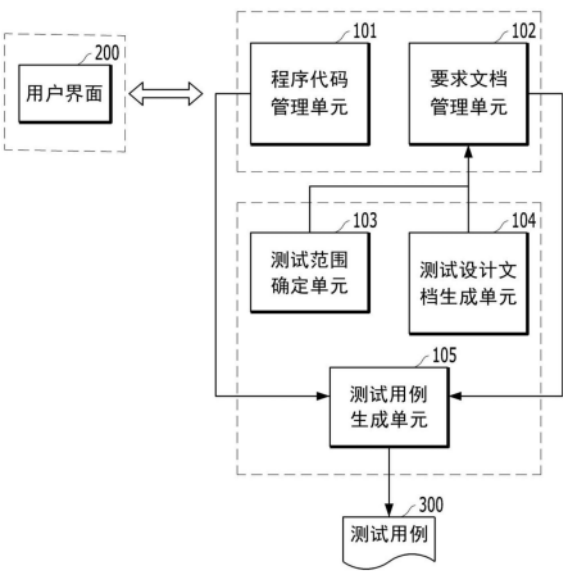
权利要求书2页 说明书11页 附图5页

(54) 发明名称

车辆功能测试装置及其控制方法

(57) 摘要

本发明公开一种用于基于缺陷注入技术生成测试用例的测试装置及其控制方法。该方法包括：基于软件详细设计文档，识别要测试的程序中的至少一个函数；基于与识别的至少一个函数的功能相关的可能发生的缺陷位置 and 要注入缺陷位置的缺陷类型，生成测试设计文档；基于生成的测试设计文档和程序的源代码，搜索要注入的缺陷位置；确定缺陷注入方式和缺陷类型；以及预测将与缺陷注入方式和缺陷类型相对应的缺陷注入应用于搜索到的位置的预期结果以生成测试用例。



1. 一种测试装置的控制方法,包括:
基于软件详细设计文档,识别要测试的程序中的至少一个函数;
基于与识别的所述至少一个函数的功能相关的可能发生的缺陷位置 and 要注入所述缺陷位置的缺陷类型,生成测试设计文档;
基于生成的所述测试设计文档和所述程序的源代码,搜索要注入的所述缺陷位置;
确定缺陷注入方式和所述缺陷类型;以及
预测将与所述缺陷注入方式和所述缺陷类型相对应的缺陷注入应用于搜索到的所述缺陷位置的预期结果以生成测试用例,
其中,生成测试设计文档包括:
基于所述软件详细设计文档,确认所述缺陷注入前的正常状态;以及
在发生缺陷时,基于技术安全要求即TSR,确认搜索/解决机制。
2. 根据权利要求1所述的方法,其中,
识别至少一个函数包括:
在单元测试的情况下,识别单元函数。
3. 根据权利要求2所述的方法,其中,
生成测试设计文档包括:
提取与识别的所述单元函数的功能相关的可能发生的所述缺陷位置。
4. 根据权利要求1所述的方法,其中,
识别至少一个函数包括:
在综合测试的情况下,对具有调用关系的函数进行整理。
5. 根据权利要求4所述的方法,其中,
生成测试设计文档包括:
提取在整理的所述函数之间调用时可能发生的所述缺陷位置。
6. 根据权利要求1所述的方法,其中,
所述测试用例包括标识符即ID、与所述缺陷注入相对应的行、所述缺陷类型、缺陷代码和所述预期结果中的至少一个。
7. 根据权利要求6所述的方法,其中,所述方法包括:
生成所述缺陷代码以违反所述正常状态。
8. 根据权利要求1所述的方法,其中,
识别至少一个函数包括:
识别反映安全机制的函数和与反映所述安全机制的所述函数具有调用关系的函数。
9. 一种非暂时性计算机可读记录介质,所述非暂时性计算机可读记录介质中存储用于执行测试装置的控制方法的程序,所述方法包括:
基于软件详细设计文档,识别要测试的程序中的至少一个函数;
基于与识别的所述至少一个函数的功能相关的可能发生的缺陷位置 and 要注入所述缺陷位置的缺陷类型,生成测试设计文档;
基于生成的所述测试设计文档和所述程序的源代码,搜索要注入的所述缺陷位置;
确定缺陷注入方式和所述缺陷类型;以及
预测将与所述缺陷注入方式和所述缺陷类型相对应的缺陷注入应用于搜索到的所述

缺陷位置的预期结果以生成测试用例,

其中,生成测试设计文档包括:

基于所述软件详细设计文档,确认所述缺陷注入前的正常状态;以及
在发生缺陷时,基于技术安全要求即TSR,确认搜索/解决机制。

10.一种软件测试装置,包括:

处理器;以及

非暂时性存储器,用于存储所述处理器能够执行的指令,

其中所述处理器被配置为:

基于与安全机制相关的函数和调用关系从要测试的程序中提取测试范围,

基于所述测试范围和至少一个要求文档来生成包括要注入的缺陷位置、缺陷类型和要违反的正常状态中的至少一个的测试设计文档,

基于要测试的所述程序的源代码和所述测试设计文档来生成测试用例,

基于软件详细设计文档来在所述测试范围中搜索安全机制,并且

基于缺陷类型系统表来确定要注入的所述缺陷位置。

11.根据权利要求10所述的测试装置,其中,

所述处理器被进一步配置为:基于软件详细设计文档中的函数标识符和与所述函数标识符相对应的函数的行来管理所述源代码。

12.根据权利要求10所述的测试装置,其中,

所述处理器被进一步配置为:在单元测试的情况下,识别单元函数作为测试范围,在综合测试的情况下,对具有调用关系的函数进行整理,并且识别整理的所述函数作为测试范围。

13.根据权利要求12所述的测试装置,其中,

所述处理器被进一步配置为:对于所述单元测试来提取与识别的所述单元函数的功能相关的可能发生的所述缺陷位置,并且对于所述综合测试来提取在整理的所述函数之间调用时可能发生的所述缺陷位置。

14.根据权利要求10所述的测试装置,其中,

所述处理器被进一步配置为:从要注入的所述缺陷位置排除未发生与所述安全机制相关的错误类型的所述缺陷位置。

15.根据权利要求10所述的测试装置,其中,

所述处理器被进一步配置为:在发生缺陷时,通过文本处理从用自然语言编写的至少一个要求文档中确认搜索/解决机制。

16.根据权利要求10所述的测试装置,其中,

所述测试用例包括标识符即ID、缺陷注入行、所述缺陷类型、与要违反的所述正常状态相对应的缺陷代码和预期结果中的至少一个。

车辆功能测试装置及其控制方法

技术领域

[0001] 本公开涉及一种车辆功能安全性测试装置及其控制方法,并且更特别地,涉及一种能够基于缺陷注入(fault injection)技术生成测试用例(test case)的测试装置及其控制方法。

背景技术

[0002] 本节中的陈述仅提供与本公开有关的背景信息,并且可能不构成现有技术。

[0003] 作为车辆功能安全性标准的ISO 26262推荐缺陷注入测试(Fault Injection Test, FIT)作为确定在进行软件单元测试和综合测试时安全机制是否正常运行的方法。然而,在仅通过对车辆用软件的功能安全的要求条件的标准化来确保用于验证软件的功能安全性的缺陷注入测试的可靠性方面存在局限性。

[0004] 为了克服这种局限性,可能需要一种能够在执行车辆用软件的缺陷注入测试时通过应用在设计阶段导出的软件设计文档和安全性要求文档来生成测试用例的方法。

发明内容

[0005] 因此,本公开涉及一种车辆功能测试装置及其控制方法,基本上消除了由于现有技术的局限和缺点而导致的一个或多个问题。

[0006] 本公开提供一种提供更方便的功能的车辆测试装置及其控制方法。

[0007] 本公开提供一种能够基于缺陷注入技术生成测试用例的测试装置及其控制方法。

[0008] 一种测试装置的控制方法可以包括:基于软件详细设计文档(Software Detailed Design),识别要测试的程序中的要测试的至少一个函数;基于与识别的要测试的至少一个函数的功能相关的可能发生的缺陷位置和要注入缺陷位置的缺陷类型,生成测试设计文档;基于生成的测试设计文档和要测试的程序的源代码,在提取的缺陷位置中搜索要注入的缺陷位置;确定缺陷注入方式和缺陷类型;以及预测将与缺陷注入方式和缺陷类型相对应的缺陷注入应用于搜索到的缺陷位置的预期结果(expected result)以生成测试用例。

[0009] 在本公开的另一方面,一种软件测试装置可以包括:测试范围确定单元,被配置为基于与安全机制相关的函数和调用关系从要测试的程序中提取测试范围;测试设计文档生成单元,被配置为根据测试范围和至少一个要求文档来生成包括要注入的缺陷位置、缺陷类型或要违反的正常状态(normal condition)的测试设计文档;以及测试用例生成单元,被配置为基于要测试的程序的源代码和测试设计文档来生成测试用例。

[0010] 根据本文提供的描述,其它应用领域将变得显而易见。应理解的是,描述和具体示例仅旨在用于说明的目的,并不旨在限制本公开的范围。

附图说明

[0011] 为了可以充分地理解本公开,现在将参照附图描述通过示例的方式给出的本公开的各种形式,其中:

- [0012] 图1是示出本公开的一种形式的测试装置的配置的示例的框图；
- [0013] 图2是示出本公开的一种形式的通过用户界面输入的程序的形式示例的视图；
- [0014] 图3是示出本公开的一种形式的测试装置执行的单元测试过程的示例的流程图；
- [0015] 图4是示出本公开的一种形式的测试装置执行的综合测试过程的示例的流程图；
- [0016] 图5是示出本公开的一种形式的测试设计文档的形式示例的视图；以及
- [0017] 图6是示出本公开的一种形式的测试用例的形式示例的视图。
- [0018] 本文描述的附图仅出于说明目的，并且不旨在以任何方式限制本公开的范围。

具体实施方式

[0019] 以下描述本质上仅是示例性的，并不旨在限制本公开、应用或用途。应理解的是，在所有附图中，相应的附图标记表示相同或相应的部件和特征。

[0020] 除非另有说明，否则本文中使用的术语“包括”或“包含”应解释为不排除其它元件，而是进一步包括这些其它元件。另外，在整个说明书中，相同的附图标记表示相同的组成元件。

[0021] 在本公开的一种形式中，提出一种测试装置及其控制方法，在执行用于验证安装在车辆的组件中的软件的安全性的缺陷注入测试时，通过输入要求/设计文档和程序代码来自动生成必要的测试用例。

[0022] 特别地，在本公开的一种形式中，提出一种测试装置及其控制方法，在执行基于汽车开放系统结构(AUTomotive Open System Architecture, AUTOSAR)的软件的缺陷注入测试时，通过文本处理从要求/设计文档和程序代码自动生成有意义的测试用例。优选地，要求/设计文档用自然语言编写。

[0023] 首先，将参照图1描述本公开的一种形式的测试装置的配置。

[0024] 图1是示出本公开的一种形式的测试装置的配置的示例的框图。

[0025] 参照图1，本公开的一种形式的测试装置包括：程序代码管理单元101，用于存储/管理要测试的程序的代码；要求文档管理单元102，用于存储/管理关于要测试的程序的文档；测试范围确定单元103，用于基于与安全机制相关的函数和调用关系从要测试的程序中提取测试范围；测试设计文档生成单元104，用于根据测试范围和要求文档生成包括要测试注入的缺陷位置和缺陷类型以及要违反的正常状态的测试设计文档；以及测试用例生成单元105，用于基于程序代码和测试设计文档生成测试用例300。

[0026] 这里，测试装置可以通过用户界面200接收或输出与测试相关的信息。例如，用户界面200可以从诸如键盘或鼠标的输入装置获取用于生成测试用例的各种信息，并且可以通过显示装置或打印装置显示/输出测试设计文档、生成测试用例的结果、存储在服务器中的其它内容。

[0027] 在下文中，将更详细地描述本公开的一种形式的测试装置的各个组件。

[0028] 首先，程序代码管理单元101存储测试程序的代码，使得在基于代码生成缺陷注入测试用例时包括具体的缺陷注入位置(injecting location)。这里，测试程序的代码可以表示源代码。在这种情况下，将验证安装在实际车辆组件(例如，控制器)中的软件的源代码，而不仅仅验证基于系统要求的模型，从而可以确保测试的可用性和有效性。

[0029] 将参照图2和表1描述程序代码管理单元101的具体的代码管理形式。图2是示出本

公开的一种形式的通过用户界面输入的程序的形式的示例的视图。

[0030] 参照图2,可以通过用户界面200将包括三个函数410、420和430的程序400输入到测试装置。因此,程序代码管理单元101可以管理/存储以下面的表1所示的形式输入的程序400。

[0031] [表1]

[0032]	行	代码
	001_1	int var2;
	001_2	g_var+=1;
	001_3	var2=FuncB();
	001_4	FuncC(var2);
	002_1	return 5;
	003_1	a=a+2;

[0033] 在上面的表1中,行(line)项以“函数ID_函数中的行的编号”的形式存储,并且函数ID表示软件详细设计文档中的函数ID。总之,从表1中可以看出,在软件详细设计文档中,每个函数具有单独的ID来进行管理。

[0034] 返回参照图1,要求文档管理单元102可以管理/存储用于生成测试用例300的中间步骤中的所有文档以及为了生成测试用例300从用户获取的文档,并且可以存储文档之间的可追溯性所需的信息。

[0035] 测试范围确定单元103可以利用存储在要求文档管理单元102中的软件详细设计文档来确定整个程序的要测试的范围。可以根据与安全机制的相关性,即,根据程序是否对应于反映技术安全要求(Technical Safety Requirement, TSR)的函数以及与该函数具有调用关系的函数,来确定测试范围。这里,技术安全要求可以在ISO 26262标准中定义。

[0036] 测试设计文档生成单元104可以利用关于测试范围中包括的函数的软件详细设计文档来搜索安全描述(Safety Description, SD)。测试设计文档生成单元104从与SD相对应的FIT相关信息将缺陷注入位置添加到测试设计文档中。此时,当在注入位置未发生与SD相关的错误类型时,将该注入位置排除。当确定注入位置时,测试设计文档生成单元104可以基于缺陷类型系统表(FIT缺陷)从SD形式将要注入的缺陷类型(注入细节(injecting details))添加到测试设计文档中。缺陷类型系统表的具体形式在下面的表2中示出。

[0037] [表2]

[0038]

测试步骤		缺陷发生位置		缺陷类型		ID	
单元测试	可运行	变量	全域变量	数据错误	无效值	1	
SW-SW 综合测试	可运行-可运行	调用函数	参数	数据错误	无效值	2	
				调用语法	程序流程错误	未调用函数	3
					绕过函数调用	4	
					非法指示	5	
		被调用函数	全域变量	数据错误	无效值	6	
				访问错误	无效地址	7	
			返回语句	数据错误	无效值	8	
				程序流程错误	非法指示	9	
				计时错误	数据丢失	10	
					CPU时钟讹误		11
	SWC - RTE				调用函数	参数	数据错误
计时错误		数据延迟	13				
		数据丢失	14				

[0039]

		调用语法	程序流程错误	未调用函数	15	
				绕过函数调用	16	
				非法指示	17	
			计时错误	CPU时钟讹误	18	
		被调用函数	全域变量	数据错误	无效值	19
				访问错误	无效地址	20
			返回语句	数据错误	无效值	21
				程序流程错误	非法指示	22
				计时错误	数据丢失	23
					无响应	24
					CPU时钟讹误	25
				不对称错误	不对称值	26
	SWC - BSW	调用函数	参数	数据错误	无效值	27
				计时错误	数据延迟	28
			调用语法	程序流程错误	未调用函数	29
					绕过函数调用	30
					非法指示	31
				计时错误	CPU时钟讹误	32
		被调用函数	全域变量	数据错误	无效值	33
				访问错误	无效地址	34
			返回语句	数据错误	无效值	35
				程序流程错误	非法指示	36
				计时错误	数据丢失	37
					无响应	38
					CPU时钟讹误	39
				不对称错误	不对称值	40
SWC - ECU HW	调用函数	参数	数据错误	无效值	41	
			计时错误	数据延迟	42	
		调用语法	程序流程错误	未调用函数	43	
				绕过函数调用	44	
				非法指示	45	
			计时错误	CPU时钟讹误	46	
	被调用函数	全域变量	数据错误	无效值	47	
			访问错误	无效地址	48	
		返回语句	数据错误	无效值	49	
			程序流程错误	非法指示	50	
			计时错误	数据丢失	51	
				无响应	52	
				CPU时钟讹误	53	
			不对称错误	不对称值	54	

[0040] 参照表2,通过将具有单独分配的ID的缺陷类型分类为多个类别来定义缺陷类型系统表(FIT缺陷)。例如,最大类别是测试步骤,子缺陷类型由测试单元根据缺陷发生位置和缺陷类型类别定义。当然,缺陷类型系统表是示例性的,不一定与表2中所示的相同。

[0041] 当添加了要注入的缺陷类型(注入细节)时,测试设计文档生成单元104可以从技

术安全要求 (TSR) 中确认用于在缺陷发生时执行搜索/解决的机制以最终生成测试设计文档。此时,测试设计文档生成单元104可以通过文本处理从用自然语言编写的TSR、下面将描述的SDD等中导出用于在缺陷发生时执行搜索/解决的机制。

[0042] 接下来,测试用例生成单元105可以基于由测试设计文档生成单元104生成的测试设计文档和存储在程序代码管理单元101中的程序代码来生成测试用例300。测试用例可以包括ID、注入缺陷(行、类型和代码)和预期结果。这里,注入缺陷行(injected fault line)可以通过在程序代码中搜索测试设计文档的注入位置来提取,并且注入缺陷类型可以选择生成(Generation,G)和突变(Mutation,M)中的一个。可以生成注入缺陷代码以违反测试设计文档的正常状态项,并且可以通过测试设计文档的处理(handling)项来生成预期结果。

[0043] 如表2所示,测试可以主要分为单元测试和综合测试。单元测试可以表示单个函数的测试,综合测试可以表示彼此相关(例如,调用关系)的两个以上函数的测试。在下文中,将基于上面参照图1描述的测试装置的配置,参照图3和图4描述单元测试和综合测试过程。

[0044] 图3是示出本公开的一种形式的测试装置执行的单元测试过程的示例的流程图。

[0045] 在描述图3中所示的流程图的顺序之前,将首先描述输入值和输出值。

[0046] 软件详细设计文档是指为了更容易地执行用于缺陷注入测试的测试目标标识过程以及随后的测试用例生成过程而补充的软件详细设计文档(SDD)。下面的表3中示出了软件详细设计文档的示例。

[0047] [表3]

[0048]

SWC	可运行		描述		映射SWR	TSR	FIT相关信息			
							全域变量	返回语句	被调用函数	参数
SWC.c	UD_001	FuncA()	SWUD_001.01	FuncA应定义0至10范围的g_var1。		TSR_001 TSR_002	g_var1			
			SWUD_001.02	var2的值应不大于10。		TSR_003 TSR_004			FuncB()	
						TSR_003 TSR_004			FuncC()	var2
	UD_002	FuncB()	SWUD_002.01	返回5	SWR_001			return 5;		
	UD_003	FuncC()	SWUD_003.01	从所给参数加2。	SWR_002					

[0049] 参照表3,SDD可以包括关于构成软件的函数的映射结构,相关技术安全要求(TSR)信息以及与缺陷注入测试相关的全域变量、返回语句、调用函数和参数信息等。

[0050] 优选地,基于以下注意事项来编写SDD。

[0051] 软件详细设计文档必须包括关于内部设计的安全机制的描述(安全描述(SD))以及软件的功能行为,并且必须用自然语言编写,并且SD的编写形式基于如下面的表4所示的AUTOSAR错误类型来分类。

[0052] [表4]

[0053]

AUTOSAR错误类型	SD形式
数据错误	数据值要求
程序流程错误	程序流程要求

访问错误	访问地址要求
计时错误	计时约束要求
不对称错误	不对称要求

[0054] 表4的细节可以如下定义。

[0055] 1) 数据值要求

[0056] -SWC应定义C1至C2范围的VAR1。

[0057] -VAR1的值应不大于(小于)C1。

[0058] 2) 程序流程要求

[0059] -应确保通过BSW/RTE正确读取VAR1。

[0060] -应确保将VAR1正确转换为CAN/SPI。

[0061] 3) 访问地址要求

[0062] -VAR1的读取/写入应通过BSW/RTE完成。

[0063] 4) 计时约束要求

[0064] -如果检测到VAR1上的缺陷超过C1,则SWC应在命令上设置VAR1。

[0065] 5) 不对称要求

[0066] -应确保输出VAR1的正确路由。

[0067] 在以上定义中,SWC表示软件组件,VAR#表示变量,C#表示常量,CAN/SPI表示与CAN/SPI接口相关的函数,BSW/RTE表示BSW/RTE层函数。

[0068] 此外,缺陷类型系统表(FIT缺陷)与上述参照表2描述的相同。

[0069] 接下来,技术安全要求可以表示ISO 26262标准中定义的技术安全要求(TSR)。下面的表5中示出了TSR的示例。

[0070] [表5]

技术安全要求	
TSR_001	检查g_var1的值。
TSR_002	如果g_var1<0或g_var1>10, 则弹出警告消息。
TSR_003	检查var2的值
TSR_004	如果var2>10, 则将var2设置为0。

[0072] 此外,程序代码可以表示由诸如MATLAB或Simulink的基于模型的开发支持工具自动生成或由开发人员直接编写的程序代码。

[0073] 参照图3,在单元测试步骤中,首先,测试范围确定单元103可以将软件单元(即,函数)识别为软件缺陷注入测试(SW FIT)的单元测试目标(S21)。测试目标优选是软件组件(Software Component,.c文件)中的函数中反映安全机制的函数。识别的测试目标可以以FIT目标系统表的形式在要求文档管理单元102中存储/管理。

[0074] 随后,测试设计文档生成单元104可以提取与单元函数的功能相关的可能发生的缺陷位置(S22)。这里,可能发生的缺陷可以表示导致软件偏离安全状态(safe state)的缺陷。具体地,测试设计文档生成单元104可以对于被识别为测试目标的函数通过软件详细设计文档来识别函数中的安全机制和与安全机制相关的全域变量,并且可以提取由于全域变

量的值的污染(潜在的缺陷原因)而可能发生的缺陷。

[0075] 另外,当确认系统中存在能够搜索并解决在先前步骤(S22)中提取的可能发生的缺陷的安全机制时,测试设计文档生成单元104可以确定在缺陷类型系统表(FIT缺陷)中定义的类型(例如,表2中的54种FIT缺陷类型)中的可适用缺陷类型(S23)。

[0076] 随后,测试设计文档生成单元104可以确认缺陷注入前的状态(S24)。这里,缺陷注入前的状态是指缺陷注入前的函数或变量的状态,即正常状态,并且可以将测试用例设计为偏离该状态。可以通过软件详细设计文档(SDD)确认缺陷注入前的状态,并且可以以确认如下面的表6所示的正常状态表的形式配置具体方法。

[0077] [表6]

[0078]	SD形式	SD细节	正常状态
	数据值要求	SWC应定义C1至C2范围的VAR1。	C1至C2范围
		VAR1的值应不大于(小于)C1。	不大于(小于)C1
	程序流程要求	应确保通过BSW/RTE正确读取VAR1。	正确读取VAR1
		应确保将VAR1正确转换为CAN/SPI。	正确转换为CAN/SPI
	访问地址要求	VAR1的读取/写入应通过BSW/RTE完成。	通过BSW/RTE读取/写入
	计时约束要求	如果检测到VAR1上的缺陷超过C1,则SWC应在命令上设置VAR1。	在C1中转换VAR1
	不对称要求	应确保输出VAR1的正确路由。	VAR1的正确路由

[0079] 参照表6,正常状态由基于SD形式(form)的SD细节(detail)定义。

[0080] 接下来,测试设计文档生成单元104参考在系统级设计步骤中编写的技术安全要求规范,确认是否存在当实际发生缺陷时能够搜索/解决的安全机制(S25)。

[0081] 通过以上过程(S22至S25),测试设计文档生成单元104可以生成测试设计文档。

[0082] 当生成测试设计文档时,测试用例生成单元105可以在程序代码中搜索在生成测试设计文档时提取的缺陷位置(S26)。

[0083] 当搜索到要注入的缺陷位置时,测试用例生成单元105可以确定作为编译型缺陷注入方式的突变(M)和生成(G)中的缺陷注入方式(S27)。

[0084] 接下来,测试用例生成单元105可以确定要注入到实际代码中的缺陷的细节(S28)。这里,优选地将注入的缺陷设计成偏离在生成测试设计文档时确认的缺陷注入前的正常状态。

[0085] 随后,测试用例生成单元105可以预测缺陷注入后的系统/软件的结果(S29)。此时,在实现安全机制的软件中执行缺陷注入,从而预期结果可以是错误处理机制的操作。

[0086] 通过以上过程,测试用例生成单元105可以生成FIT测试用例。

[0087] 接下来,将参照图4描述综合测试过程。

[0088] 图4是示出本公开的一种形式的测试装置执行的综合测试过程的示例的流程图。

[0089] 由于图4所示的过程类似于图3所示的单元测试过程,因此为了简化说明书,将仅描述不同之处。

[0090] 在综合测试过程中,代替应用于单元测试的SDD,每个AUTOSAR层的设计文档(Design Document for each AUTOSAR layer)用作测试范围确定单元103和测试设计文档生成单元104的输入值。因此,可以将函数识别过程(S21)替换为对测试目标中具有调用关系的函数进行整理的过程(S31),并且可以将提取与单元函数的功能相关的可能发生的缺陷位置的过程(S22)替换为提取在函数之间调用时可能发生的缺陷位置的过程(S32)。

[0091] 后续过程(S33至S39)分别对应于上述过程(S23至S29),因此将省略重复的描述。

[0092] 在下文中,将参照图2所示的程序再次描述测试装置的操作。

[0093] 首先,测试人员通过用户界面200接收技术安全要求(TSR)和软件详细设计文档(SDD),并将TSR和SDD存储在要求文档管理单元102中。TSR可以具有表5中所示的形式,SDD具有表3中所示的形式。

[0094] 类似地,所示程序的源代码可以通过用户界面200存储在程序代码管理单元101中,并且如上所述,存储形式可以转换为表1的形式。

[0095] 在单元测试的情况下,测试范围确定单元103可以仅将函数FuncA() 410、FuncB() 420和FuncC() 430中实现安全机制(即,反映TSR)的FuncA() 410确定为测试目标。

[0096] 在综合测试的情况下,测试范围确定单元103可以将与实现安全机制的FuncA()的调用关系(例如,FuncA()和FuncB()之间的函数调用或FuncA()和FuncC()之间的函数调用)确定为测试目标。

[0097] 在单元测试的情况下,测试设计文档生成单元104可以将函数FuncA()的全域变量“g_var1”确定为注入位置的候选。由于SDD中描述的内容对应于SD形式的“数据值要求(data value requirement)”,并且可以根据表2在全域变量发生数据错误,因此测试设计文档生成单元104可以选择“g_var1”作为缺陷注入位置。

[0098] 在综合测试的情况下,测试设计文档生成单元104可以在FuncA()和FuncB()之间的函数调用中将函数调用语法FuncB()和FuncB()的返回语句“return5;”确定为注入位置的候选,并且在FuncA()和FuncC()之间的函数调用中将函数调用语法FuncC()和参数“var2”确定为注入位置的候选。由于SDD中描述的内容对应于“数据值要求”,而根据表2只能在全域变量、参数和返回语句发生数据错误,因此从缺陷注入位置排除FuncB()和FuncC()。

[0099] 图5中示出了由测试设计文档生成单元104通过上述过程生成的测试设计文档的形式。

[0100] 图5是示出本公开的一种形式的测试设计文档的形式的示例的视图。

[0101] 参照图5,在单元测试510的情况下,测试设计文档中可以定义单个函数的ID、注入位置、缺陷类型、正常状态、检测和处理值,在综合测试520的情况下,测试设计文档中可以定义具有调用关系的函数中与缺陷注入位置相对应的两个函数中的每一个的ID、注入位置、缺陷类型、正常状态、检测和处理值。

[0102] 接下来,将参照图6描述基于如图5所示生成的测试设计文档的测试用例生成单元105的操作。

[0103] 图6是示出本公开的一种形式的测试用例的形式的示例的视图。

[0104] 参照图6,在单元测试610的情况下,测试用例生成单元105可以确定作为注入缺陷代码的代码,通过该代码,“g_var”的位置“001_2”变为与测试设计文档510的“TS_001_1”相关的注入缺陷行,并且g_var的值变为作为边界值的-1和11,以违反测试设计文档的“正常状态(0至10范围)”。预期结果可以变为测试设计文档的处理项TSR_002。

[0105] 在综合测试620的情况下,测试用例生成单元105可以确定作为注入缺陷代码的代码,通过该代码,返回语句“return5;”的位置“002_1”变为与测试设计文档520的“TS_001_2”相关的注入缺陷行,并且返回值变为作为边界值的11,以违反测试设计文档520的“正常状态(不大于10)”。预期结果可以变为测试设计文档的处理项TSR_004。从测试设计文档520的“TS_001_3”的测试用例生成过程也类似于“TS_001_2”,因此将省略重复的描述。

[0106] 在本公开的一种形式中,测试装置的程序代码管理单元101和要求文档管理单元102中的至少一个可以由单独的装置替代,或者可以被单独地实现为程序记录装置(存储器等)或电子控制单元(ECU)。

[0107] 另外,除了输出自动生成的测试用例之外,测试用例生成单元105还可以自动执行并确定诸如ECU的程序目标的缺陷注入测试,可以输出缺陷注入测试的结果,并且可以在缺陷注入时由于缺少缺陷和安全(fail and safe)逻辑而发生程序错误时通过推荐缺陷和安全逻辑来执行平滑的代码修改。

[0108] 此外,在图3或图4中,可以输入过去主要在用于批量生产的车辆的车辆用软件中发生的缺陷模式来代替SDD,并且可以将测试设计文档生成单元104配置为不仅确认缺陷发生时搜索/解决机制的存在,而且在省略搜索/解决机制的情况下推荐适当的搜索/解决机制。

[0109] 上述本公开的形式具有以下效果。

[0110] 本公开的形式涉及一种用于生成基于AUTOSAR的软件的测试用例的方法和装置,其中可以通过文本处理从用自然语言编写的要求(TSR、SD等)文档自动生成源代码的用于确认软件中的安全机制的操作的测试用例,从而提高便利性。

[0111] 另外,在本公开的一种形式中,可以确保在利用缺陷注入技术执行基于AUTOSAR的软件的单元测试和综合测试时测试过程的可靠性并且可以缩短开发周期。

[0112] 此外,可以将基于AUTOSAR的软件中可能发生的缺陷类型定义为AUTOSAR标准,可以规定软件安全要求的形式,从而可以客观地准备主观干预的要求并且可以通过文本处理自动生成测试用例。

[0113] 以上描述的本公开可以被实现为存储在计算机可读记录介质中的计算机可读程序。计算机可读介质可以是以计算机可读方式存储数据的任何类型的记录装置。计算机可读介质可以包括例如硬盘驱动器(HDD)、固态硬盘(SSD)、硅磁盘驱动器(SDD)、只读存储器(ROM)、随机存取存储器(RAM)、光盘只读存储器(CD-ROM)、磁带、软盘和光学数据存储装置。

[0114] 从以上描述显而易见的是,如上所述构造的本公开的一种形式的测试装置能够更方便且可靠地验证车辆用软件的安全。

[0115] 特别地,在本公开的一种形式中,自动生成用于确认源代码的安全机制的操作的测试用例,从而可以确保在基于缺陷注入技术执行测试时测试过程的可靠性并且可以缩短开发周期。

[0116] 本公开的描述本质上仅是示例性的,因此不脱离本公开的实质的变型旨在在本公

开的范围内。这样的变型将不被视为脱离本公开的宗旨和范围。

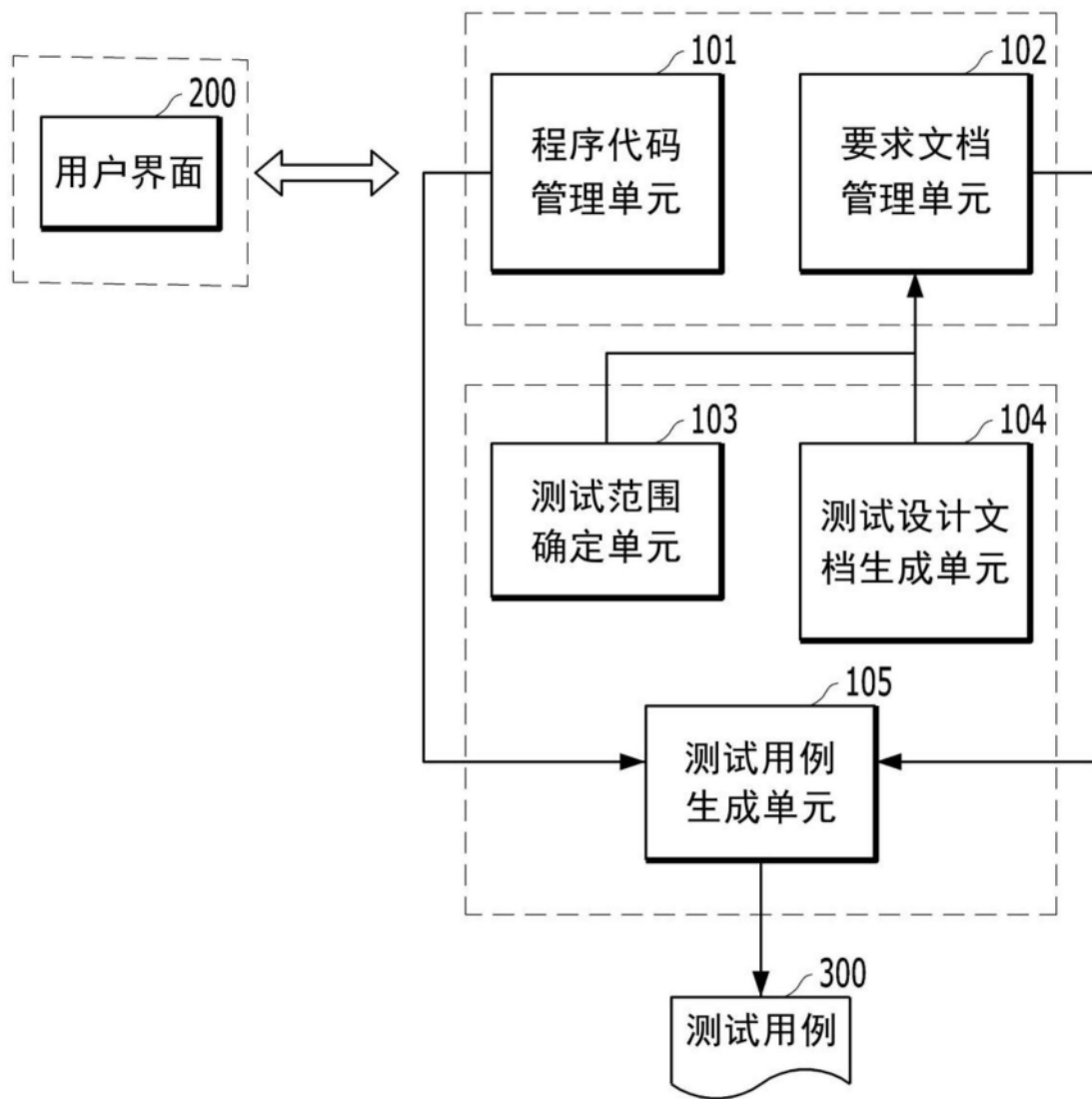


图1

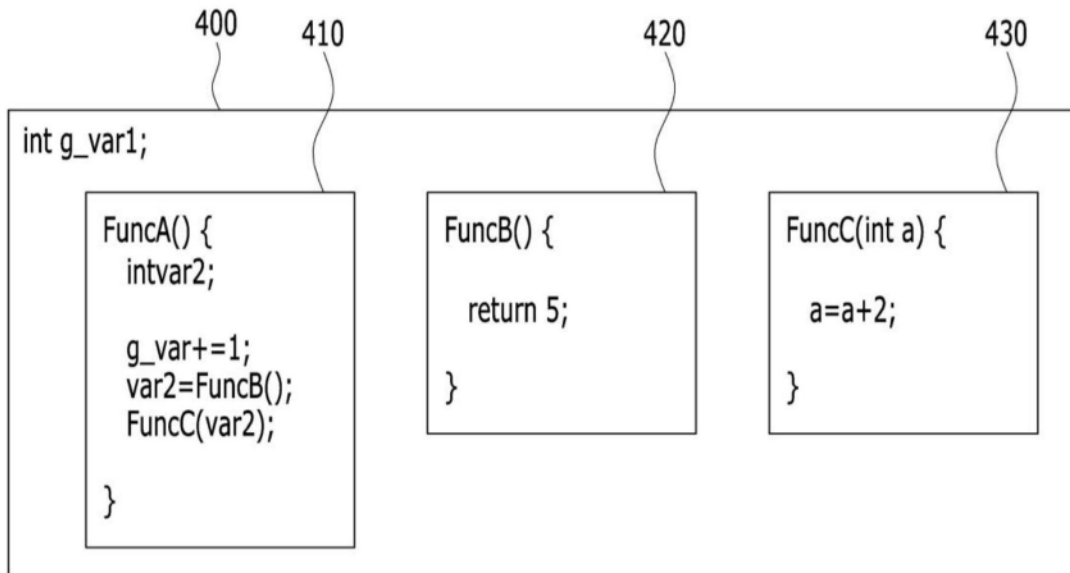


图2

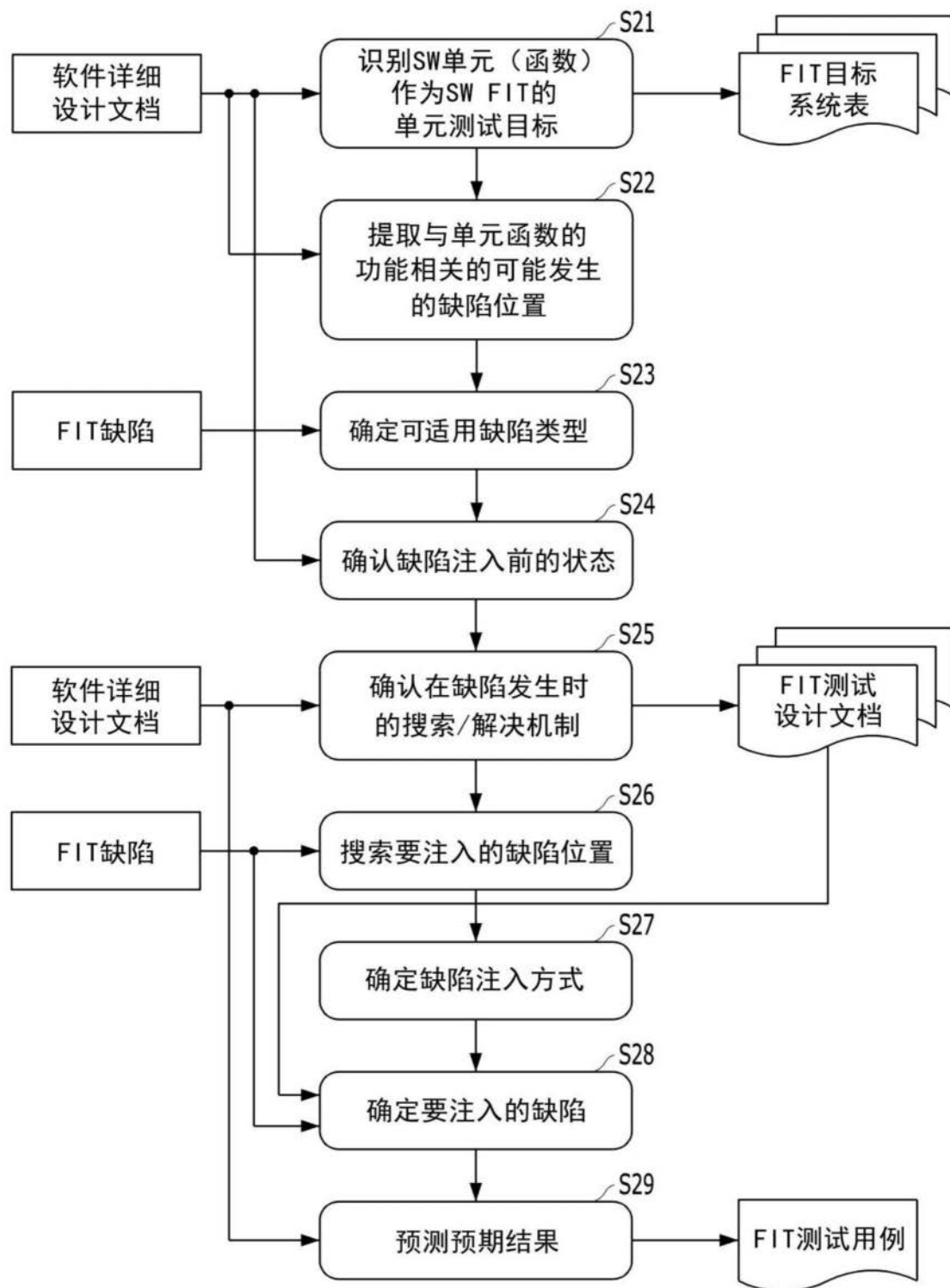


图3

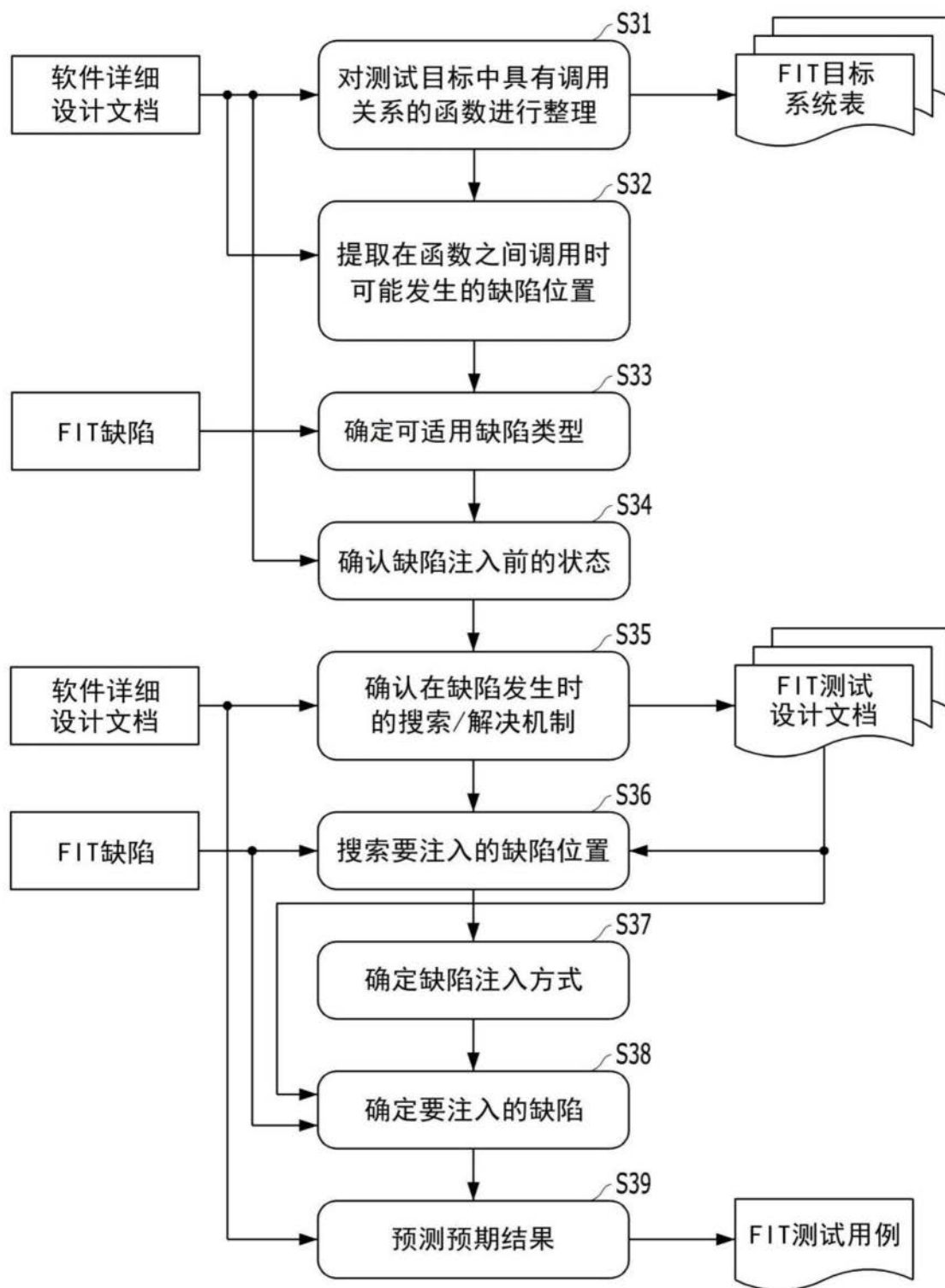


图4

SWC	可运行	ID	注入位置		注入细节 (缺陷类型)	正常状态	检测	处理
SWC.c	FuncA()	TS_001_1	G	g_var1	无效值	0至10范围	TSR_001	TSR_002
	FuncB()	TS_001_2	R	return 5;	无效值	不大于10	TSR_003	TSR_004
	FuncA()	TS_001_3	P	var2	无效值	不大于10	TSR_003	TSR_004

图5

SWC	可运行	ID	注入缺陷			预期结果
			行	类型	代码	
SWC.c	FuncA()	TS_001_1_1	001_2	G	g_var = -1;	弹出警告消息
		TS_001_1_2	001_2	G	g_var = 11;	弹出警告消息
	FuncB()	TS_001_2_1	002_1	M	return 11;	将var2设置为0
	FuncA()	TS_001_3_1	002_1	M	FuncC(11)	将var2设置为0

图6