文章编号:1671-4598(2019)09-0057-04 DOI:10.16526/j.cnki.11—4762/tp.2019.09.013 中图分类号:TP273 文献标识码:A

嵌入式软件自动化测试及管理系统研究

阳长永，王月波，代 林
(中国西南电子技术研究所，成都 610036)

摘要：随着嵌入式软件技术的发展，嵌入式软件规模日益扩大、复杂程度越来越高，给软件测试提出了挑战；针对复杂嵌入式软件，设计一种软件自动化测试及管理系统，融入软件工程化管理过程，将自动化测试与测试管理过程相结合，打造一个无缝连接的系统；自动化测试及管理系统包括：接口设计、测试策划、测试设计、测试执行、测试结果显示与记录、缺陷管理和回归测试，以及数据分析和知识库等；在此基础上，设计并实现了自动化测试及管理系统，并在某测试项目中进行了应用；实践证明，使用自动化测试及管理系统开展嵌入式软件测试可以有效提高测试效率和质量。

关键词：复杂嵌入式软件；自动化测试；测试流程；系统架构

Research of Automated Testing and Management System for Embedded Software

Yang Changyong, Wang Yuebo, Dai Lin
(Southwest China Institute of Electronic Technology, Chengdu 610036, China)

Abstract: With the development of embedded software technology, embedded software is expanding in scale and complexity, which poses challenges for software testing. Considering software engineering management process, designed a software automated test and management system for complex embedded software. This system includes: interface design, software test planning, software test design, software test implementation, software test result display and documentation, defect management and regression testing, and data analysis and knowledge base. Then designed the system framework, and developed the system, which was applied in a test project. Practice has proved that using the system to carry out embedded software testing can effectively improve test efficiency and quality.

Keywords: complex embedded software; automated testing; test process; system framework

0 引言

随着信息技术的飞速发展和用户要求的不断提高，嵌入式软件向综合模块化发展[1]，规模日益扩大、复杂程度越来越高，软件的质量对整个产品的质量起到了决定性的作用。软件测试是保障软件质量的重要手段。目前嵌入式软件的配置项测试和系统测试以手工测试为主，手工测试具有创造性，能举一反三地对系统逻辑、功能进行验证，但对于复杂嵌入式软件，系统交联复杂、接口数据众多，采用手工测试存在以下问题：一是影响测试的准确性，二是测试效率低[2]。同时，对于一些特殊的性能测试、压力测试和强度测试等，采用手工测试，则很难或根本无法实施。随着软件开发周期的日益缩短以及软件系统的日趋复杂，引入自动化测试，可以缩短软件开发周期，降低成本，同时高质量地完成测试任务，提高软件质量。

目前嵌入式软件自动化测试的研究主要有基于GUI的自动化测试[3-4]，测试模式分为录制回放、数据驱动和关键字驱动[5]；基于模型驱动的自动化测试[6]；针对特定业务的自动化框架设计[2,7]，以及基于分布式仿真测试环境的嵌入式软件自动化测试[8]等。本文在此基础上提出一种面向复杂嵌入式软件的自动化测试方法，融入软件工程化管理过程，将自动化测试与测试管理过程相结合，打造一个无缝连接的系统，提高测试效率和质量。

1 自动化测试及管理系统设计

自动化测试就是将繁琐的、重复的手工操作利用策略、工具等实现自动化的测试活动[9]。自动化测试不是完美无缺的，它需要和手工测试互相配合、优势互补，才能发挥出各自的优势。手工测试是整个测试的核心和基础，自动化测试无法完全替代手工测试，但能够间接辅助手工测试提高测试效率，保证测试质量。

针对复杂嵌入式系统，被测软件有着复杂的交联环境，存在多个配测对象，配测对象可能是模拟器，也可能是真实的设备，且包含不同类型的交联接口等。为适应这样的现状，面向复杂嵌入式软件的自动化测试及管理系统的基本设计思想可以归纳为如下几条。

1）资源共享：开发与测试人员共享接口数据：开发人员设计、修改ICD后，测试人员能第一时间获得最新信息；

收稿日期:2019-03-22； **修回日期**:2019-04-09。
专利:自动化测试嵌入式软件的方法(CN201410655457)。
作者简介:阳长永(1983-),女,重庆人,硕士,工程师,主要从事机载软件测试技术和软件安全性技术研究,以及软件测试工作方向的研究。

开发和测试人员在同一个平台进行软件缺陷生命周期管理：测试人员提交软件缺陷后，开发人员能马上得到缺陷信息，缺陷修复后，缺陷的状态变化和修改方法也能第一时间告知测试人员；

2）手工测试与自动化测试互相配合：测试策划和测试设计以手工设计为主，自动生成测试数据为辅，充分发挥人的主观能动性和创造性；

3）测试设计与测试执行不分离：测试设计的数据，通过测试脚本驱动，发送给被测软件，作为测试执行的输入激励，测试设计与执行不再分离，测试结果自动记录，提高测试的效率和结果的可信性；

4）支持分布式测试：系统支持分布式的测试驱动和数据监控；

5）测试文档自动生成：测试策划、测试设计的用例/数据，以及系统自动记录的测试结果，通过导入不同的模板，自动生成符合要求的测试文档；

6）建立知识库：积累可复用的测试用例和典型缺陷等知识，提升组织的测试水平。

2 自动化测试及管理流程

本文运用软件工程化的思想将接口设计、测试策划、测试设计、测试执行、测试结果、缺陷管理和回归测试集成到一个系统进行管理[10]，自动化测试及管理流程如图 1 所示。

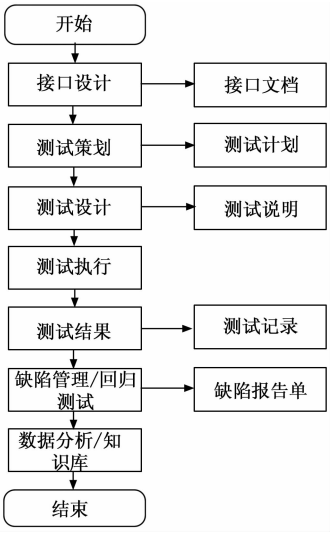

图 1 自动化测试及管理流程

2.1 接口设计

开发人员使用接口设计模块进行 ICD 设计，基于模块化和复用方面的考虑，接口设计的元素分为数据元、数据块、数据组、逻辑块和接口消息模块几部分，数据元为最小的数据单元，数据块由多个数据元组成，代表一个有意义的字段，数据组由多个数据块或数据元组成，代表一个更大的含义。一条接口消息模块就是一条完整的 ICD，由多个数据组、数据块或数据元组成；逻辑块用来进行特殊逻辑设计，限定不同元素之间的特殊关系，如变长设计、有效性设计等；根据项目不同，数据元和数据块之间的级数可以配置，以方便开发人员根据项目情况进行扩展设计。每条 ICD 都要包括数据的源 ID 和目的 ID，以及适用的接口类型。设计好的 ICD 存储在数据库里，后台生成标准的 XML 格式文件，以供测试数据自动生成、测试数据解析和接口文档自动生成使用。

2.2 测试策划

测试人员根据开发设计的 ICD、软件研制任务书、软件需求规格说明书、用户手册等相关资料，进行测试策划。测试策划包括建立测试交联环境图，制定测试类型，设计测试项或测试子项，测试用例名称以及测试用例描述。

在自动化测试系统中，测试人员的主要工作集中在测试策划和测试设计上。在测试策划中，首先建立测试交联环境图，设置被测软件模块之间的接口类型，并为每一个软件模块设置一个与 ICD 对应的 ID 号。建立环境图后，测试人员根据需求规格说明等被测软件的依据文档，进行功能分解和测试分析，提取测试需求，设计测试项和测试子项。测试类型包括功能测试、性能测试、接口测试、强度测试等，测试类型下面是测试项，测试项下面是测试子项，一个测试子项包含多个测试用例，测试项和测试子项之间的级数是可配置的。测试用例在测试策划时要填写测试用例名称和测试意图，即测试用例描述。测试子项的测试描述就是其下多个测试用例的描述合在一起，每个测试用例描述的就是一个测试点。自动化测试系统支持根据不同格式的模板生成测试计划中测试需求分析的内容。

2.3 测试设计

测试人员对测试用例开展具体的测试输入输出设计。对于嵌入式软件，大部分的测试输入和输出都是一系列的数据。因此测试设计的主要内容就是给每个用例设计不同的输入输出数据。在用例设计之前，需设置用例的属性，用例属性包括独立用例和关联用例。新建的测试用例默认为独立用例，如果设计的测试用例是进行场景或业务流测试，则需将其属性设置为关联用例。

测试设计分为三步，第一步为测试数据自动生成，自动化测试系统根据 ICD 中的数据元素，如枚举值、范围值、分段枚举等，结合特殊的逻辑关系自动生成测试数据，测试数据生成策略包括等价类划分法、边界值分析法、正交法等，生成的测试数据包括枚举值的所有值，范围值的上下边界值、中间值和边界外的异常值，以及所有的特殊关系，对于没有特殊关系的多个数据元素使用正交法生成测试数据。

第二步为独立测试用例设计，测试人员将本用例涉及到的 ICD 拖入到测试设计界面，界面根据 ICD 的源和目的 ID，建立本用例的交联运行图，双击图上的连线即可设计测试数据，也可以选取自动化测试系统自动生成的测试数据，同时可以设置每条测试数据的属性和延时，数据属性

包括事件触发、周期触发、消息触发、数据监听等，数据属性可以添加和配置。自动生成的测试数据主要针对枚举值、边界范围值等，其他经验值或故障注入的数据需要测试人员单独进行设计。

第三步为关联测试用例设计，测试人员可以使用场景分析法，基于场景或业务流，将不同的测试用例通过流程图的方式联系在一起，设计场景或业务流测试用例，也可以对不同的状态转换进行测试。独立测试用例主要针对单个功能点进行测试，关联测试用例将不同功能点的用例连在一起，可以模仿用户在实际使用过程中的场景，进行场景或业务流的测试。在关联测试用例设计界面上，可以看到所有的独立测试用例，测试人员设计好流程框图后，将需要的独立测试用例拖入到对应的框图中即可。测试用例设计示意图如图 2 所示。

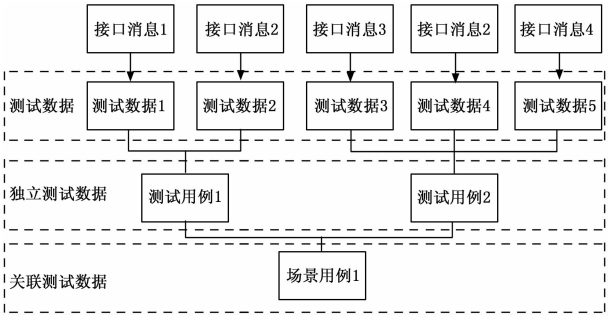

图 2　测试用例设计示意图

在界面设计好测试数据及属性后，自动化测试系统自动生成测试脚本和测试步骤，步骤的描述围绕数据的发送、接收展开，以数据为中心，格式统一。自动化测试系统支持根据不同格式的模板生成测试说明文档，其中测试用例的描述在测试策划时由测试人员设计，测试步骤由自动化测试系统自动生成。

2.4　测试执行

测试执行平台由测试交联环境图、测试用例、总线路由共同组成。测试执行分为三步，第一步为数据分发，测试人员设置执行某个测试用例后，由总线路由的总控端根据设计的交联环境图和测试用例，得到涉及的模块和接口总线类型，初始化相关的代理端，同时将输入的激励/测试数据分发给对应的代理端；第二步为代理端驱动接口通信模块，代理端根据交联环境图，将从总控端收到的输入测试数据发给数据接口驱动模块，由接口驱动模块将输入测试数据发送给被测软件；第三步为代理端监控被测软件的响应，并将响应的报文，即输出的测试数据回传给总控端。一个总控端可以控制多个代理端，代理端的个数与交联环境图中交互的模块数有关，一个代理端监控一个点对点的数据传输。测试执行示意图如图 3 所示。

测试用例的执行支持单个测试用例的执行和多个测试用例的顺序执行。测试人员勾选要执行的测试用例或测试用例集，启动测试执行，系统调用脚本自动完成测试执行。

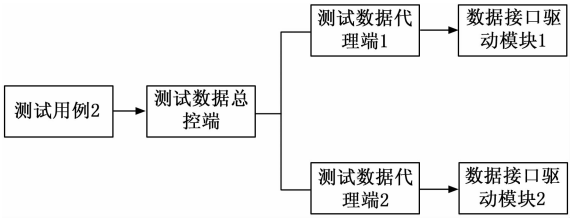

图 3　测试执行示意图

2.5　测试结果显示与记录

总控端收到代理端传回的数据后，进行显示，并根据 ICD 进行自动解析，界面显示所有的交互数据，双击选中的数据后，可以看到数据的解析，直观地让测试人员了解测试的执行情况，同时支持对显示的数据进行过滤，以便于观察。

系统收到代理端传回的数据后，与测试用例中预期的测试结果报文进行比较，如果相同则显示用例步骤通过 Pass，如果不相同则显示该步骤失败 Fail。任意步骤 Fail 则判该用例 Fail。所有的测试结果自动记录到数据库里，系统支持根据不同格式的模板生成测试记录。

2.6　缺陷管理和回归测试

测试人员要对测试结果进行确认，特别是批量执行的测试用例，需对每个 Fail 的测试用例的每个 Fail 的步骤进行确认，以便排除非被测软件的故障。确认是被测软件缺陷的，在对应的测试用例后提交缺陷记录单。提交缺陷记录单后，软件项目经理查看到缺陷信息，对缺陷进行确认和分配，对应的开发人员可以参考缺陷追踪的测试用例进行问题复现，并修改缺陷，修改后的缺陷状态标识为已修复，测试人员对已修复的缺陷进行回归验证。

在回归测试中，如果 ICD 没有变更，回归时可以复用前期设计的测试用例，并自动进行回归测试执行；如果 ICD 的数据元素发生了变化，自动化测试系统自动标识出受影响的测试用例，提示测试人员对用例进行修改后再执行回归测试。测试用例在不同测试阶段所做的修改，自动化测试系统会将其标识为不同的版本存入数据库，以追溯数据修改过程和版本管理。回归测试通过后关闭已修复的缺陷。对于回归中仍未修改到位的缺陷，rebound 给开发，开发再次进行修改，之后测试人员再次进行回归。整个项目回归完成后，缺陷的状态有三类：已修复、撤回和遗留。自动化测试系统支持根据不同格式的模板导出缺陷记录单。

2.7　数据分析和知识库

测试结束后对测试过程进行总结，对测试用例数、测试通过率、缺陷率等数据进行统计、输出规定格式的报表；同时将设计精巧的、可复用的测试用例纳入典型测试用例库，将发现的有价值的缺陷纳入典型缺陷库，形成知识库，如此沉淀、积累测试经验，以便于后期在做类似项目的测试时，进行借鉴，以持续提高。

3 系统架构及设计

面向复杂嵌入式软件的自动化测试及管理系统是一个面向信息和数据驱动的测试系统，测试系统采用分层架构，包括测试管理层、消息服务层、总线路由层和驱动层，具体如图 4 所示。

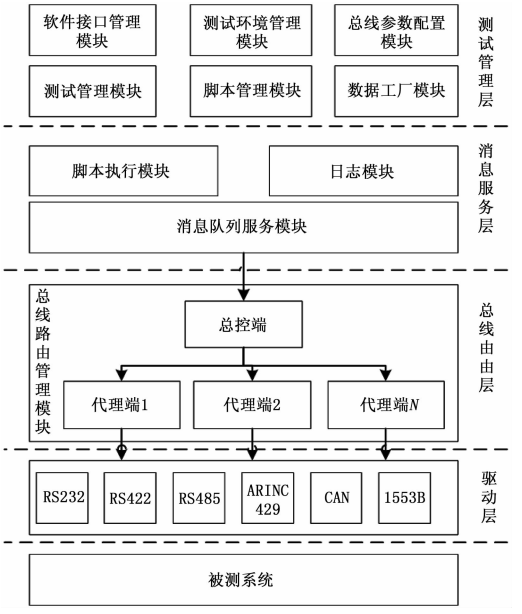

图 4 自动化测试系统架构图

测试管理层包括软件接口管理模块、总线参数配置模块、测试环境管理模块、测试管理模块、脚本管理模块、数据工厂模块；消息服务层包括脚本执行模块、日志模块、消息队列服务模块；总线路由层包括一个总控端和多个代理端；驱动层包括测试涉及到的不同类型的接口驱动。

软件接口管理模块对接口设计进行管理；总线参数配置模块对涉及到的不同类型的总线配置参数进行管理；测试管理模块对测试策划、测试用例设计、测试执行、测试结果显示、缺陷管理流程进行管理；测试环境管理模块对测试环境、测试执行策略、测试轮次等进行管理；脚本管理模块对测试用例脚本、测试调度脚本、测试驱动脚本和公共函数进行管理；数据工厂模块对数据库进行管理，存储的数据包括：测试用例/测试数据、测试执行结果、测试缺陷、测试分析的数据、知识库等。

测试管理层为 BS 结构，包括服务器和客户端，客户端分布在不同的电脑上，与服务器通过以太网连接，支持多人同时开展接口设计、测试用例设计等。

系统界面和框架采用 QT 语言实现，自动化测试脚本使用 Python 语言。部分模块采用开源系统，如缺陷管理模块集成 BugFree 等。

4 实验结果与分析

设计的自动化测试及管理系统在某型机载综合处理设备控制软件测试中进行应用，图 5 为接口设计、交联环境、测试策划、测试设计和测试执行的综合图例。

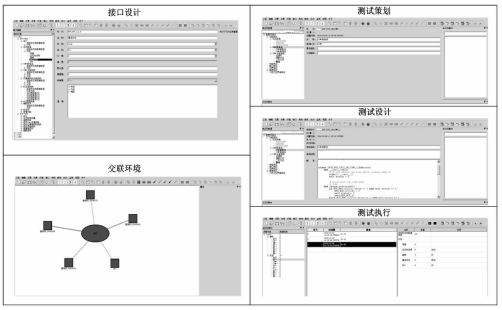

图 5 某型机载综合处理设备控制软件测试图例

使用自动化测试及管理系统开展软件测试，开发人员设计好接口后，测试人员设计交联环境图，搭建测试环境，部分测试模拟器不需要再额外开发；开展测试设计时，可以在系统中直接调用部分自动生成的测试数据，也可以在此基础上手动修改成需要的测试数据，测试人员的主要精力集中在测试策划和测试设计上，设计好测试用例的测试数据及属性后，系统自动生成测试脚本和测试步骤；测试执行时支持批量测试用例的执行，可以实现人休息而设备不休息，白天进行测试设计晚上自动执行测试等；测试结果由系统自动记录，高效、准确，并且可以自动进行测试结果数据统计等。

实践证明，使用自动化测试及管理系统开展嵌入式软件测试可以有效提高测试效率，具体如表 1 所示。测试执行和回归测试效率提高了 80%，整体项目测试效率提高了 65%。

表 1 自动化测试前后效率对比(单位:人日)

对比项	人工测试	自动化测试	时间减少	效率提升
环境搭建	5	2	3	60%
用例设计	12	8	4	25%
用例执行	20	4	16	80%
回归测试	5	1	4	80%
结果统计	2	0.5	1.5	75%
合计	44	15.5	28.5	65%

5 结束语

本文运用软件工程化的思想，将接口设计、测试策划、测试设计、测试执行、测试结果、缺陷管理和回归测试集成到一个系统进行管理，测试人员的主要工作集中在测试策划和测试设计上，测试用例设计好后，系统自动生成测试脚本，支持批量测试用例的自动执行，自动收集、监控测试过程数据，自动记录测试结果，并根据模板自动生成相应的测试文档，实现了一个具备过程管理、信息发布、缺陷跟踪、知识积累等功能的高效一体化测试，测试过程设计更加简便快捷，提高了软件测试的效率和质量。

(下转第 75 页)