

面向嵌入式通信设备的 Ch 自动化测试框架设计^①

毛玉书, 胡维华

(杭州电子科技大学 计算机学院, 杭州 310018)

摘要: Ch 是一种 C/C++ 脚本语言, 用 Ch 脚本语言对通信设备进行测试具备简单, 快速, 可直接嵌入等优点。RobotFramework 是基于 Python 语言开发的开源的自动化测试系统, 它具有简单明了的上下级结构, 可灵活重用的 KeyWord, 独立于平台和应用, 具有很强的扩展性等多种优点。首先介绍构建与实现了一个适合通信设备的 Ch 脚本测试新框架, 并利用 RobotFramework 极强的扩展性, 把 Ch 测试脚本无缝融合进 RobotFramework, 融合后的 Ch 测试框架同时具备了 Ch 脚本和 RobotFramework 的优点。

关键词: Ch; RobotFramework; 自动化测; KeyWord; 嵌入式通信设备

Ch test Automation Framework Oriented to Embedded Communication Equipment

MAO Yu-Shu, HU Wei-Hua

(College of Computer Science, Hangzhou Dianzi University, Hangzhou 310018, China)

Abstract: This is a script language of C/C++. Employing Ch script language to test communication equipment is simple, rapid, directly embedded, etc. RobotFramework is a test automation system based on the Python language. It has simple structure of upper and lower levels and flexibly reused keywords, which is independent from platforms and applications, and also highly scalable. Firstly, this paper studies and implements a new framework which is suitable for automated testing of embedded communication devices, it also integrates Ch script language into RobotFramework by using the highly scalable feature of RobotFramework. The integrated test framework has the advantages of both Ch script and RobotFramework.

Keywords: Ch; RobotFramework; test automation; KeyWord; embedded communication equipment

1 引言

在软件开发过程中, 随着软件规模越来越大, 业务逻辑越来越复杂, 软件测试框架的好坏, 软件测试效率的高低对提升整个软件开发的质量至关重要。

现在的通信设备软件一般由系统语言 C/C++ 开发, 而 Ch 是一种 C/C++ 脚本语言, 所以在测试通信软件时, Ch 脚本有它的优势, 它可以和 C/C++ 软件无缝衔接, 不需要在通信软件上封装一层 CLI(Commod Line Interface), 有些通信设备无 CLI, 若采用其它测试脚本则工作量会大很多, 也更容易出错。

RobotFramework 是一种自动化测试管理系统, 运用 RobotFramework, 用户可以将复杂的测试软件逻辑分解, 分级整合到这个系统里面, 利用这个系统可以

使测试用例结构简明, 容易复用。

基于此, 本文利用介绍 Ch 脚本的特点创建一种面向嵌入式通信设备自动化测试框架, 并将测试框架上实现的 Test Case(测试用例)整合进 RobotFramework, 以便拥有其的优点。

2 Ch脚本及RobotFramework测试系统介绍

2.1 Ch脚本^[1]

Ch 它是一种跨平台的解释型脚本语言, 其支持 1990 年制定的 ISO C90 C 标准, 也支持 C++ 的大部分特性。Ch 支持了比 C 编译器更多的 C99 特性。Ch 是一个高级语言和低级语言的桥梁, 可以很方便的操作内存, 具有脚本语言的无需编译调试, 可嵌入的优点,

^① 收稿时间:2010-06-06;收到修改稿时间:2010-07-07

它弥补 C 语言在应用方面的不足,在数字计算,图形绘制,嵌入式脚本等方面都有很灵活的应用。

因为 Ch 具有无需编译调试,可无缝嵌入软件,具有 C/C++特征这些特点,且通信设备软件一般采用 C/C++开发,所以 Ch 特别适合做通信设备的自动化测试。

2.2 RobotFramework 系统^[2]

RobotFramework 基于 Python 开发,可扩展的关键字驱动的自动化测试系统。测试用例位于 HTML 或者 TSV 文件,使用在测试库中实现的关键词来在测试中运行程序。因为 RobotFramework 是灵活和可扩展的,所以它很适用于测试具有多种接口的复杂软件:用户接口,命令行,web service,编程接口等。图 1 是 RobotFramework 系统的层次结构。

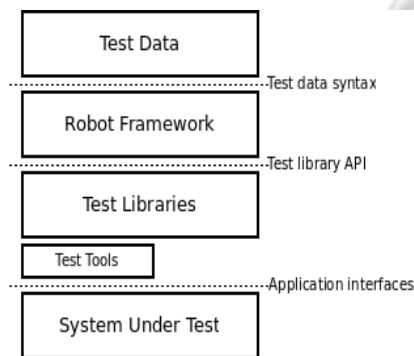


图 1 RobotFramework 系统结构

第 1 层,测试数据 (Test Data),它按照 RobotFramework 的语法规则被整合在这个系统中,是一个或多个具体的测试行为 (Test Cases)。

第 2 层,RobotFramework 提供的是一个用户易操作的平台,它与具体的测试程序无关,它起到了管理的作用。

第 3 层,测试库 (Test Libraries) 中提供关键字,这些关键字既可以是已经整合在 RobotFramework 里面的现有库,也可以是用户根据需求自我扩展的库。

最后两层的就是被测软件提供的信息,用户可以通过一些工具来测试被测试软件的属性信息,或者直接通过被测软件的函数接口来测试。

3 架构基于 Ch 脚本的自动化测试系统

本研究在此部分利用 Ch 脚本和通信设备软件无缝衔接的特点,以 Ch 脚本为语言,构建了一个适合嵌

入式通信设备的自动化测试新框架。为了使新框架具有 RobotFramework 测试系统的好处,本章最后把这个框架整合进 RobotFramework。

3.1 新框架术语

(1) 网元,即通信设备。

(2) 网元通信 API,它指的是和设备通信的一些通用基础函数。

(3) 设备属性,一个网元通信设备软件以几百上千个设备属性为基础,一个小功能对应一个设备属性,例如网元设备的温度即是一个功能,它就有对应的一个设备属性。

(4) Struct,设备属性对应的 C 结构体,它是这个设备属性在和设备通信时数据的载体。

(5) 通用 API,一些通用全局的函数,如连接网元的函数 NeConnect(),遇到错误打印错误退出程序的函数 FATAL(),以及遇到错误打印错误不退出程序的函数 ERR()等。

(6) 原子操作 API,对设备属性封装的最基本的操作(获取和设置属性)。它根据相应的属性结构体(即对应的 struct)及已有的通用 API 和网元通信 API 编写的。

(7) 原子操作组合 API,它通过对原子操作 API 进行组合,萃取出常用的功能 API。

(8) Test Case,一个及以上的原子操作 API 或原子操作组合 API 组合成功能单一的简单测试用例。

3.2 新框架和通信设备无缝衔接流程^[3]

通信设备代码大同小异,软件一般由 C/C++编写,设备由许多功能构成,具体功能对应一个属性,如获取设备温度是一个属性。每个属性以对应的具体 C 语言的 struct 结构体为载体^[4]。某些属性的 struct 结构体相当复杂,嵌套多层,而且有些结构体是变长,数据长度由获取的实际值决定。若用类似 Python, Tcl 等其他脚本语言表述各个属性 struct,由于设备属性很多,且有些很复杂,工作量将会相当大。一般通信公司在设备软件属性结构体 struct 上面封装了一层 CLI(command-line interface)^[5],因此 Python, Tcl, Perl 等脚本语言就可以利用 CLI 对通信设备进行设置和测试。但是很多公司没有封装 CLI,且封装这层 CLI 的工作量很大,容易很多错误。本测试框架能避免这种缺陷。Ch 脚本可以直接利用通信设备软件内部的各个属性 struct 作为载体,无需转化表达^[6]。

上述流程如图 2 所示:

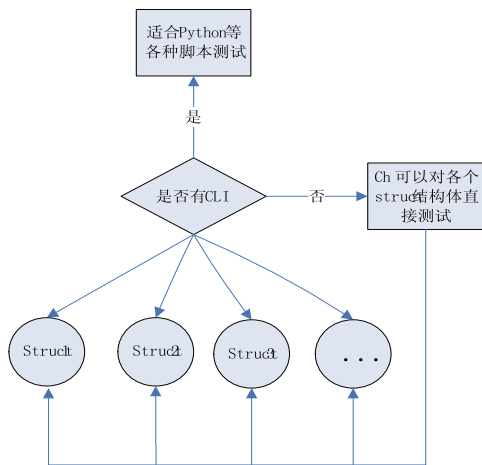


图 2 新框架和通信设备无缝衔接流程

3.3 Ch 测试框架总体设计^[7,8]

本文已经实现了这个基于 Ch 脚本的面向嵌入式通信设备的自动化测试框架, 测试程序员只要根据具体设备和业务做部分改动, 调用框架上已有的函数就可以写自动化测试用例。

如图 3 是整个 Ch 测试框架结构:

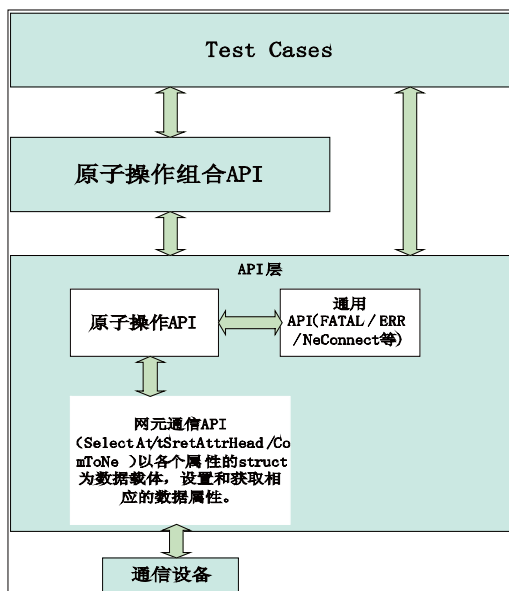


图 3 Ch 自动化测试框架总体设计

如图 3 所示, 本框架实现了网元通信 API,通用 API, 利用网元通信 API 可以与设备进行数据通信,

利用通用 API, 本框架可以对网元进行连接, 对错误信息进行打印等一些操作。测试程序员利用这两者编写原子操作 API, 原子操作组合 API, 再经过人工合理的组合成 Test Cases。在此框架下, 测试程序员的主要精力在编写原子 API 上, 下面是一个获取网元温度的原子 API 的伪代码:

GetTemperature()//获取设备温度

```

{
    NeConnect(10.8.159.163)//连接网元 IP
    //三个网元通信 API
    SelectAttr(TemperatureID); //根据属性 ID 选择属性, 这里是温度 ID
    SetAttrHead(arg1, arg2,...); //相应属性头的一些要求
    ComToNe(); //通过 SOCKET 和网元进行数据交换
    FATAL(); //若对网元操作失败则打印错误, 并退出程序
}
    
```

测试程序员只需利用这简单的几句代码, 就可以对网元众多属性的 struct 进行操作, 它通过属性 ID 来区分设备属性, 例如获取网元温度的操作, 经过这三个网元通信 API 的正确填写并执行后, 用户就可以在相应的网元温度的 struct 变量中, 获取到真实的网元温度数据。

3.4 Ch 测试框架与 RobotFramework 的融合

为了利用 RobotFramework 测试系统的优势, 本研究把图 3 中的自动化测试新框架整合进 RobotFramework。

3.4.1 两者融合的好处

Ch 脚本框架产生的众多原子操作 API, 以及原子操作组合 API, 它们需要测试程序员组织成复杂的测试用例, 这个工作量很大, 原子操作不容易积累和复用。RobotFramework 以关键字为单位, 它是个良好的关键字驱动的系统, 具有良好的复用机制, 所以本研究 Ch 脚本产生的原子操作 API 或者原子操作组合 API 整合入 RobotFramework 的关键字库里面, 这样 Ch 自动化框架就拥有 RobotFramework 的所有优点。在 RobotFramework 下, 本研究将每个原子操作 API 或者原子操作组合 API 看做作为一个关键字, 这样本研究利用了 RobotFramework 易管理, 易复用, 易扩展的优点,

原子操作 API 和原子组合 API 作为关键字在 RobotFramework 下很容易积累,并能非常灵活地组合成不同的 Test Case,减少了人工合成 Test Case 的工作量。

3.4.2 融合流程

框架结构如图 4,要把 Ch 框架融合到 RobotFramework 中,只要在 Ch 中创建相应的原子 API(原子操作 API 或者原子操作组合 API),然后把把这些 Ch 原子 API 用 Python 语言集合到在 RobotFramework 中的新的库文件 LIB1(假设 LIB1 是 Ch 测试框架专用)中。即把众多 Ch 原子 API 变成相应库中的关键字(KeyWord),框架中有多个库,一部分是框架集成的,一部分是用户自建的。每个库中有一个或者多个关键字,每个 Test Case 是多个关键字的集合,一个 Test Case 可能调用了多个关键字,一个关键字可能被多个 Test Case 调用。库中的关键字和 Test Case 是多对多的关系。

Test Case 是一个完全可以实现的最小测试用例,用户可以使用 RobotFramework 中 pybot 命令来执行测试用例,一个 Test Case 也可以当做一个关键字,重新和一些关键字组合生成新的 Test Case。

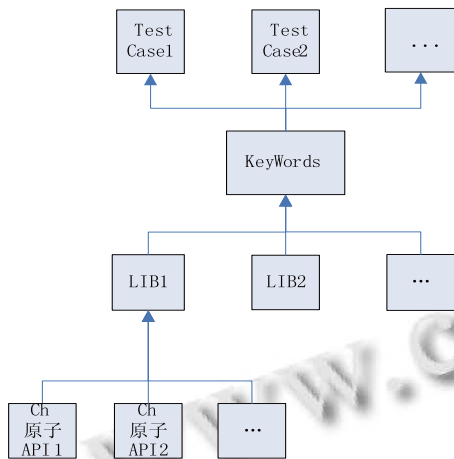


图 4 Ch 框架下的原子 API 与 RobotFramework 融合流程

3.4.3 原子 API 与 RobotFramework 融合实例

用户可以用 python 或者 java 来扩展自己的库,下面是一个用 Python 写的库文件(MyLibrary),其里面包含了一个关于网络连接的关键字(conn),其中 Connection 是原有框架已完成的原子 API。

实现伪码:

```

class MyLibrary:
def conn(host, int(port)):
Ch Connection(host, int(port))
    
```

在 Robotframework 中添加库文件如表 1 所示:

表 1 RobotFramework 库文件

Setting	Value	Keyword	Arg1	Arg2
Library	MyLibrary	conn	10.0.0.1	8080

4 结语

为了和嵌入式通信设备软件无缝衔接,本文介绍了一个基于 Ch 脚本的自动化测试框架,并把这个框架中的基本原子操作 API 整合到 RobotFramework 的库里,整合后的 Ch 测试框架同时拥有了 Ch 脚本和 RobotFramework 的优点,该框架已经在某通信公司具体的嵌入式通信设备中进行了应用,应用结果表明它具有如下优点:(1)和通信设备软件无缝衔接;(2)易学的表格语法;(3)从已有关键字可以创建可重用的高级关键字;(4)易读的 HTML 格式的报告和日志;(5)独立于平台和应用;(6)可以用 Python 或者 Java 创建自己的测试库;(7)提供命令行接口和基于 XML 的输出以方便集成到现有系统。

参考文献

- 1 Ch: An Embeddable C/C++ Interpreter. [2010.1.3-2010.5.8]. <http://www.softintegration.com>.
- 2 Robot Framework User Guide.[2010.1.3-2010.5.8].<http://code.google.com/p/robotframework>
- 3 International standard:Programming languages-C, ISO/IEC, Geneva, Switzerland,1999.
- 4 SoftIntegration GraphicalLibrary. [2010.1.3-2010.5.8].<http://www.softintegration.com/products/silib/graphlib/>
- 5 Cheng HH. C for Engineers and Scientists: An Interpretive Approach. New York: McGraw-Hill, Inc. March 2009.
- 6 Cheng HH.Handling of Complex Numbers in the Ch Programming Language. Scientific Programming, 1993,2 (3): 76—106.
- 7 Cheng HH. Extending C with Arrays of Variable Length. Computer Standards and Interfaces, 1995,17:375—406.
- 8 Zhu Y, Chen B, Cheng HH. An object-based software package for interactive control system design and analysis. ASME Trans. Journal of Computing and Information Science in Engineering, 2003,3(4):366—371.