

# Automation of Interactive Telecom Application using Expect/TCL Programming

Rashmi Singh, Prasanna Ravindran  
Rashmi.Singh@aricent.com, prasanna.ravindran@aricent.com  
Aricent Communication Softwares, Plot 17, Sector 18, Electronics City,  
Gurgaon, Haryana, INDIA  
<http://www.aricent.com>  
**Gurgaon**

## Abstract

Test Automation is an emerging field in the Software Testing nowadays to perform Regression Testing and functional testing. Test Automation has its own significance in various applications both Interactive and Non-Interactive. Test automation of Telecom Applications, which are mostly interactive and are having RTOS in nature, is an uphill task since it behaves in more complex manner in the live network. In this paper we propose how expect can be used to automate the complex interactive embedded telecom application in an efficient manner. The System Under Study (SUS) is a real telecom network element, which operates on a real time operating system. The Automation is done using TCL/Expect commands to check its functionality and features. TCL/Expect is used for interacting with the application, querying the system, analyzing the responses and reporting the results. The automation architecture involves interaction with simulators, interface between simulator and the SUS and reporting on the Test Management tool. This paper is intended for Testing Managers and Testing Engineers to understand the concept and challenges of Interactive application automation.

**Key words: Test Automation, Expect Programming, Test Setup, Automation Strategy, Implementation, Benefits & Issues.**

## 1. Introduction

As the organization grows so does the complexity of the products handled and tested. This calls for more focus on the testing, before a product is shipped to the customer. As a result the testing process takes a significant chunk of the overall project time. The more we invest in testing the more we reap in term of quality and stability of the product. Testing the application on different releases consumes lot of time particularly in regression. There comes the picture of automation, which helps not only in regression but also in functional testing. Automation helps us in reducing the time of execution and the manual testing efforts involved for doing the regression testing. Automation is performed based on the type of application. Lot of tools are available for GUI automation whereas the interactive applications has lot of limitations and assumptions. In this paper, we proposed an idea of how to automate an interactive telecom application.

## 2. Test Automation

Test automation is the use of software to control the execution of tests, the comparison of actual outcomes to predicted outcomes, the setting up of test preconditions, and other test control and test reporting functions. Commonly, test automation involves automating a manual process already in place that uses a formalized testing process. Test Automation eases the regression testing and functional testing. Testers are being asked to test more and more code in less and less time. Test automation is one way to do this, as manual testing is time consuming. As and when different versions of software are released, the new features will have to be tested manually time and again. With the help of automation, the regression testing is performed without taking much time thereby the quality of testing the new features will improve. Test automation is a boon to the organization in terms of execution time and manual efforts.

### 3. Expect for Interactive Application

Expect is an efficient tool for handling the interactive applications. Expect is Unix Automation and testing Tool developed by DonLibes. The tool can be used effectively for automating interactive applications such as for interactive applications such as telnet, ftp, passwd, fsck, rlogin, tip, ssh, and others. Expect has regular expression pattern matching and general program capabilities, allowing simple scripts to intelligently control programs such as telnet, ftp, and ssh, all of which lack a programming language, macros, or any other program mechanism. The result is that Expect scripts provide old tools with significantly new

power, flexibility, and reliability. An Expect script can spawn a shell, look up environmental variables, perform some Unix commands to retrieve more information, and then enter into the product's command-line interface armed with the necessary information to achieve the user's goal. After looking up information inside the product's command-line interface, the script can make an intelligent decision about what action to take, if any.

Every time an *Expect* operation is completed, the results are stored in a local variable called *\$expect\_out*. This allows the script to both harvest information to feedback to the user, and it also allows conditional behavior of what to *send* next based on the circumstances.

Expect is basically an extension of TCL with commands for handling the interactive environment. Expect combined with TCL provides an efficient way to test the applications. Expect is a powerful scripting language to Automate Command Line Applications in Telecom domain. Expect has regular expression pattern matching and general program capabilities, allowing simple scripts to intelligently control the interactive programs. Using Expect, the interactive output is captured and the output is manipulated using TCL to test it efficiently.

#### 3.1 Sample Program in Expect

The sample program of telnet a server is given below:

```
spawn telnet $remote_server
expect "username:"
# Send the username, and then wait for a
password prompt.
send "$username\r"
```

```

expect "password:"
# Send the password, and then wait for a shell
prompt.
send "$password\r"
expect "%"
# Send the prebuilt command, and then wait for
another shell prompt.
send "$command\r"
expect "%"
# Capture the results of the command into a
variable. This can be displayed, or written to
disk.
set results $expect_out(buffer)
# Exit the telnet session, and wait for a special
end-of-file character.
send "exit\r"

```

The output of the commands gets stored in a temporary buffer and it can be retrieved through any variable. Operating on the variable through any standard language helps us in further manipulation. Commonly, TCL is the language, which is used for operating on these variables.

## 4. Automation of SUS

### 4.1 Test Setup

The Test bed setup is given below:

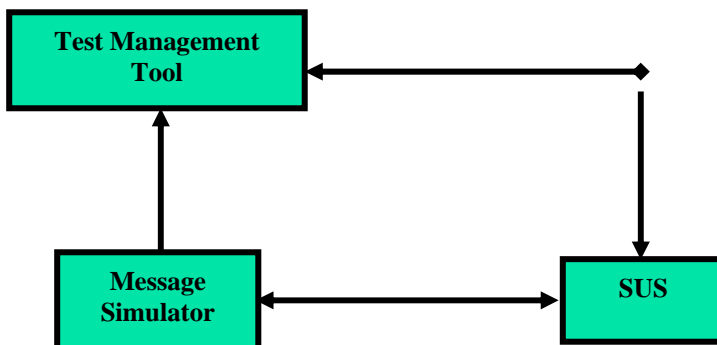


Figure1: Test Bed Setup

The Test Bed comprises of the SUS, a message simulator and a test controller for managing the execution. The test controller used is the one, which manages the execution as well as provides an interface for the SUS and the message simulator. The message simulator sends the message to the SUS. The SUS on receiving the messages responds back based on its intended operation. The traces are logged and they can be verified using TCL. On top of these components, the test controller, which is basically a test management tool, controls the entire execution and provides the results in a user-friendly manner.

## 5.Automation Strategy

To meet the challenges of testing large number of cases and carrying out repeated regression tests, testing framework is developed that made complete automation of testing possible. The framework involves multiple interfaces and the key requirement for automation was synchronization and coordination between these interfaces.

The framework is developed based on the modularity level and the scripts are made in a more modular level. Any changes can be taken care easily by modifying only the functions. The coding guidelines are set for all the automation engineers thereby ensuring the quality of code also. This can easily merge any changes in the test plan with the code.

## 5.1 Tools used for automation

The following tools were used for this automation:

Message Simulator, a tool that simulates the peer nodes and is used for sending messages between these nodes.

Test Management Tool is used for execution and management of different interfaces. It is also used for presenting the results in a user-friendly manner.

*TCL/Expect* – The basic building block interactive application automation lies with *TCL/Expect*. The scripts were written in *TCL/Expect* and executed via Test Management tool.

## 5.2.Implementation

The individual scripts for test cases are written in such a way that it does the initial configuration of SUS and sends the messages to the network element and gets the responses. At the end of every test case, cleanup activity is performed which deletes the entire initial configuration made by the script. The scripts are complete in themselves, handling the flow as per the interface protocols, and contributing towards the main verdict. The test management tool was designed for coordination. It achieves the task of synchronization of the scripts, to produce the end-to-end scenario. Besides the synchronization, it also enables the batch mode testing, which is basically execution of a batch of test cases. The test management tool triggering the action and collecting the responses from individual scripts achieves batch mode. Upon final execution of all the scripts, it gives a verdict for the test case based on its responses. It also consolidates the results at multiple points of observations. This reduced the testing turn-a-

round time drastically and helped the tester to have more time for identifying more scenarios for testing which further enhanced the test coverage.

## 6.Benefits & Issues

The automation of the SUS basically reduces our testing efforts as it adds more value in regression testing of the application. Time efforts are reduced in comparison to the manual testing efforts. The automation can be deployed on the client side also to reduce the testing efforts.

The automation is always needed for a kind of product, which comes up with various releases and new features along with the existing ones. Automation reduces the time drastically and it provides a value add to the organization also. Automation of interactive applications needs expertise and the understanding of the framework. Getting the response from the SUS, fixing up the machine delay and determining the behavior are the key issues faced in automation

## 7.Conclusion

The Automation of the interactive telecom applications are very challenging and it needs lot of efforts and expertise. Because of Automation, Customer's confidence is increased as more time can be spent on understanding the System and Quality Testing can be achieved.

## References:

1. Libes, Don (1995). *Exploring Expect: A Tcl-Based Tool for Automating Interactive Programs*. O'Reilly & Associates, Inc
2. <http://en.wikipedia.org/wiki/Expect>
3. <http://expect.nist.gov/>
4. <http://wiki.tcl.tk/201>

## ***Bibliography***

***Rashmi singh & Prasanna Ravindran*** are associated with **Aricent Communication Softwares**, Gurgaon for the past 1.5 year. Both are having a good expertise in application testing and automation. They possessed a good knowledge on SS7, TCL, Expect, Rational Robot, Rational Functional Tester and Test management Tools. They are actively involved in the automation activities of the various applications in Aricent.