



TAC Automation T-API and Tools

Overview

Test & Automation Consulting LLC (TAC) understands that test automation is a software development process. Current automation tools do not supply all of the resources necessary to support a robust programming environment. Therefore, in order to best support our clients, we have created and maintain a number of code libraries and tools that add functionality to the commercially available automation tools.

Each TAC-supported automation tool has a set of native code functions written specifically for it. Mercury-Interactive WinRunner™ has a library written in TSL, Compuware QARun™ has a library written in QA Basic, and Rational Robot™ has a library written in SQA Basic. At this point, TAC has no native code libraries for Segue Silk™ or Rational Visual Test™, although they could be ported quickly once we have a client who needs them.

In addition, a common set of functions supplied by custom written DLLs is available to each tool. These functions consist of routines that are useful when automating, but would be impractical or impossible to write in the native tool languages. These libraries can be used by any tool that has the capability of importing functions from DLLs (all current major automation tools currently have this capability.)

TAC also supplies a number of stand-alone tools which support automation, a pair of external logs, a full set of script templates, and a design methodology to tie all of these artifacts together in a way that can ensure the highest return on investment (ROI) for a client's testing dollars. All of the above listed artifacts, processes, and templates are known collectively as the **Test – Automation Programming Interface (T-API)**. T-API is also sometimes referred to as the **TAC Infrastructure**.

An automator who uses T-API and the TAC tool set has an immediate technical edge over other automators who either must use the tools as designed or write their own routines to support their work. This document will describe the TAC infrastructure and explain why we believe that Test & Automation Consulting's T-API provides the edge in test automation.

Native Code Libraries

Wherever possible, functionality to extend an automation tool is written in its own native code. This facilitates the most seamless integration with each tool and ensures the fastest execution when run.

As each tool supplies different technical capabilities, however, the TAC T-API native code libraries for each tool are slightly different. In those cases where a tool does not supply sufficient capability to support a necessary function, the functionality has been



Innovative solutions to maximize your testing investment.

supplied via custom DLL services. For example, WinRunner does not support a good set of basic string handling functions to build on; therefore, most of the string functions that T-API adds are from the TAC DLLs. QARun and Robot both support very robust string handling capabilities; therefore, an automator using those tools may decide to use the native tool functions rather than the supplied DLL string functions.

In the documents listed hereunder, the available T-API native code functions for each tool are listed. As the TAC infrastructure has been developed to support ongoing automation projects, it is not abnormal for some functionality to be available for one tool but not yet for others. It should be understood that all functionality would be made available in each supported tool as needed. Test & Automation Consulting LLC understands that it is essential to ensure that the TAC infrastructure be comprehensive for any tool a client may own.

Finally, it should be noted that all functionality has been documented. Some of the functions may not seem to be useful at first glance to the automator. Some functions are supplied to support other infrastructure functionality. In other cases, functions may have been created to solve a very specific problem in a particular environment and may not be useful in other situations. Like any other code library, the TAC T-API has been written to support automation in many different environments. It is expected that the automator will use only those functions that are immediately useful to their work.

See Documents:

- TAC Compuware QARun Native Code T-API Libraries.doc**
- TAC Mercury-Interactive WinRunner Native Code T-API Libraries.doc**
- TAC Rational Robot Native Code T-API Libraries.doc (TBD)**

Custom DLL Libraries

Occasionally, test automation tools do not supply the required capability to perform some desirable development tasks. Other times, the complexity of the code may be reduced or safety increased by isolating the code in a dynamic link library (DLL.)

T-API includes a number of DLL modules that contain code that can be executed in any of the commercially available tools. These are written in Delphi and supported by TAC. See the document:

- TAC Custom DLL Libraries.doc**

Script Templates

One of the powers that TAC T-API bestows on an automator is the ability to easily build scripts that are very robust: able to execute the desired architecture without extra coding. The mechanism we use to support this level of functionality is the **Template Script**. By pre-building the architecture into the script, TAC has given its automators a very real



Innovative solutions to maximize your testing investment.

competitive advantage. There are different templates to support different execution models. Each tool has its own templates, which can be found in the following documents:

- TAC Compuware QARun Script Templates.doc**
- TAC Mercury WinRunner Script Templates.doc**
- TAC Rational Robot Script Templates.doc (TBD)**

Custom Automation Tools

The TAC T-API includes several stand-alone executables that perform specific tasks in the automation environment. These are custom written by TAC to support the automator during the creation, execution, or post-run review of a test suite. These tools are described in the document:

- TAC Custom Automation Tools.doc.**

External Logs

Good logging can raise the ROI of an automated test suite by facilitating the troubleshooting of failed test cases. Unfortunately, the internal logs used by most automation tools are lacking many of the features a test team needs. To help a test team understand what happened during a test run, TAC supplies two separate external logs. These external logs are described in the document:

- TAC External Log Design.Doc.**

Processes and Techniques

Successful test automation consists of many different processes working together. The TAC processes and techniques that have been developed for test automation are documented in the following document:

- TAC Processes and Techniques.Doc**

Models

How do we model the environment in which we are testing? How does the record / playback (R/P) method model testing, and why doesn't it work? What is the overall model that TAC uses for its automation? All of these questions and more are answered in the document:

- TAC Automation Models.doc**



Innovative solutions to maximize your testing investment.

Architecture Theory

What different architectural models are available to the automator? The following document looks at architectural models that have been used in automation from a theoretical point of view.

TAC Automation Architecture Theory.doc

TAC Architecture

So what architecture does Test & Automation Consulting LLC implement for our clients? How do we take all of the theory and models, and create a working automation framework that is robust and maintainable? The following document contains the architectural and high-level details of the TAC Infrastructure.

TAC Architecture and High Level Design.doc

Managing Complexity

The key to successful automation is the ability to manage the complexity that is inherent in all development projects. The following document discusses the problems of automation complexity, and explains the how the TAC T-API helps to manage it.

TAC Managing Automation Complexity