BIO

PAPER

W3

November 6, 2002
1:00 PM

TEST AUTOMATION: REDUCING TIME TO MARKET

Jim Dougherty and Keith Haber
LexisNexis

International Conference On
Software Testing Analysis & Review
November 4-8, 2002
Anaheim, CA USA

Jim Dougherty

Jim Dougherty is a Consulting Software Test Analyst in the Global Electronics Product Development division at LexisNexis in Miamisburg, OH. He has been with LexisNexis in a test capacity for the past six years.

Prior to joining LexisNexis, his testing background was as a member of the Air force Communications Command's Operational Test & Evaluation Center at Wright-Patterson AFB, OH. He was a Team Engineer on a Wideband Microwave evaluation team and later was assigned as the Team Chief of a Base Telecommunications evaluation team evaluating airbase telecommunications systems worldwide.

During his time at LexisNexis Jim has been the test lead for many critical projects.
He has been the Automation Team lead for the past two years.

Jim is an American Society for Quality Certified Software Quality Engineer.

Keith Haber

Keith Haber is a Software Test Analyst in the Global Electronic Product Development division at LexisNexis in Miamisburg, Ohio. He has been a member of the GEPD Product Certification group since 1998, and has worked exclusively in the field of test automation for over three years.

Keith is a junior at Wright State University pursuing a Bachelor of Science degree in Computer Science.

Automation
Reducing Time to Market

Jim Dougherty
LexisNexis™

International Conference on

Software Testing, Analysis & Review

November 4-8, 2002

The Beginning

The test department began it's introduction to automation approximately four years ago when two automation tools were compared in an evaluation and a decision was made to select what appeared to be the tool that offered the most value to an as yet undefined process.

The test department at that time had a population of approximately 40-45 test analysts, none of whom had ever been exposed to an automation tool. The introduction to automation continued by presenting each of the test analysts a copy of the software and the documentation offered by the provider.

The test analysts were instructed to begin learning and using the product by using the documentation, the on-line tutorials available, and by taking small segments of test plans and experimenting with creating automation for these segments.

The result was the generation of large numbers of capture-replay created automation. The initial response to these test scripts was excitement, the scripts worked, the automated product permitted improved regression testing of the applications under test, and the test analyst could perform tasks that were not amenable to automation while the test scripts were running.

More and more test scripts were created using the capture-replay capability of the product. They numbered in the hundreds for a single group of products alone. They did however permit an expanded scope of testing and appeared to offer even greater promise.

During this period, market considerations began to impact the appearance and performance of the applications under test. As the user interface for a product or group of products underwent change, massive failures of the capture-replay scripts began.

The test analysts were forced essentially to stop testing until these scripts were repaired, or as was pretty much the case, the test cases were again automated using capture-replay.

The initial excitement generated by the apparent successful use of automation was now tempered by the impact of the necessity to repair or replace hundreds of automated test cases. The management team and the test leads began meeting to address the situation.

Addressing the Issues

The focus of the discussions initially was what to do about the problem and how quickly could it be done. Additionally, how could this type of problem be prevented from re-occurring? The initial approach was to investigate the training programs offered by the tool developer. What were the costs involved, where did the cost go in the budget? Who should be trained? What would the expectations be after the training was completed.

These and several peripheral issues were discussed and within a relatively short period approximately fifteen members of the test staff were provided with a four day training session that consisted of two days in the basic use of the tool and two days introducing the students to scripting using the tool's proprietary language.

This of course took time, it was done off-site at two different locations and the cost was not trivial. The resources coming back from the courses were better informed of the tool's capabilities, with methods to access those capabilities, and how to consider and begin to implement something other than capture-replay automation of their test cases.

What became apparent rather quickly that as the resources began to attempt to use what they had learned at the training courses, as they began to search for better ways to automate, was that automation takes time when you are expert in that task. When you are learning how to practically apply classroom lessons and perform your normal testing duties, not only does one of the tasks suffer, it became apparent that both tasks suffered.

There were exceptions within the group, team members who immediately grasped the concept and the techniques for creating methods, the creation of variables, the idea of passing arguments to a variable, having central files that identified the application objects, and these resources were quite comfortable doing so.

These few resources were adopted by the test group as an ad hoc source of expertise. As time permitted they helped team members with problems, offered suggestions as to methods, and were generally keeping the automation effort afloat. They were loosely called an automation team but had no specific responsibilities or goals; they were basically a “finger in the dike.”

During this time, products were expanding, new products were added, and many products experienced significant functionality and user interface changes. An inventory of manual test cases during the late summer of 2000 indicated there were approximately 9,500-10,000 manual test cases as opposed to several hundred automated scripts that had an extremely short useful life.

Decisions were made concerning the automation team and the automation process.

What Was Done

The automation team was assigned a team lead whose directive was to deal with the deficiencies in the automation process. The team would take on all automation and free the test staff to create documentation and perform their tests without the onerous, to the test staff; burden of creating usable automated test cases. There was also the extremely large number of manual tests cases, priorities to be assigned and implemented and a method to accomplish all this.

A Flagship product with a large number of tests cases was selected to begin the effort. A contractor was asked to supply a Project Manager and ten skilled developers who however had no skills with the automation tool's language. The automation team members provided an automation tool user's guide and hands on training and in approximately four weeks the team was able to develop approximately 500 automated test cases. This effort alone reduced the execution time for these tests from 40 person hours to ten hours of running time. It became clear that automation, properly done, is effective. There was still a large body of manual test cases waiting for automation.

During this effort the team was building the outline of a formal automation process. It was imperative to the success of the team and the automation effort that a clearly defined path exist for the test staff to present a request for automation, to be advised where test plans did not lend themselves to automation, to be provided feedback and suggestions where changes could be made that would then permit the test case to be automated. The requester would be provided with a sizing, that is to say how long it would take to complete the automation. A priority was assigned so that the requester knew exactly where they were in the queue and when they could expect to have access to the automated product.

The first part of this task was the creation of the Automation Process document. This document describes in its entirety the process, the flow of the work, the responsibilities assigned to the automation team members and those of the testing staff. What the deliverables are for each of these groups. Who would do maintenance when it was required, and, if maintenance was assigned to the test staff, it outlined the provisions for staff training so that the staff could complete tasks of this nature.

The team recognized that standards would be essential to the success of the automation effort. Using software development best practices, a standards document was created and reviewed and then implemented. All code created would be bound by these standards.

Change control for all developed code was implemented; all code was stored in Microsoft Visual Source Safe. All pertinent documentation was also maintained under change control.

A library of stand-alone functions began to grow, the ability to use and re-use these methods or functions was supported by the fact that every member of the team was adhering to the standards, that the scripts and master test plans created might differ by application and functionality but the manner in which they were created was identical.

The sizing standards and tool were refined as experience indicated that through the use of the stand-alone functions and following a common development method the time required to create the automated product could be reduced.

Tools to produce plan files and element methods were created and put into use, further increasing the team member's efficiency.

The work in progress and completed work was tracked using an Excel spreadsheet. By the close of the year 2001 there were approximately 8,300 automated test cases in the inventory.

Improving Automation

Although significant progress had been made and measured, there was a strong desire to improve what was being done. None of what the team accomplished came cheaply. Automation is expensive. Several analysis efforts did indicate however that for one of the Flagship products 2610 tests were run during a certification effort. With 0 percent of those cases automated this was a 9-day/11-resource effort. With 12 percent of 3274 test cases, certification required 6 days/6 resources. At a point where 55 percent of 3216 test cases were automated, the certification effort required 4 days/6 resources.

The cost of testing during 2001 was significantly reduced. But automation was still an expensive line in the budget.

Investigation into several articles about data driven automation revealed that if applicable, this method could not only significantly reduce the size of the scripts created during automation but would provide more robust scripts requiring less maintenance and thus reducing the cost of rework.

Data Driver.inc was created. This was the tool used to process .CSV files in a generic manner dispatching the instructions to function/method calls. It was determined by analysis that approximately 79 percent of the test plans presented for automation would lend themselves to this method of automation. Templates were created to use during the creation of the test plans and the automation process. A document defining the creation of a test plan for data driven automation was provided to the test staff.

A sample data driven test plan was created and demonstrated to the management team for their consideration. In addition to the demonstration of this method, a cost analysis comparing the existing cost of producing automation using functional decomposition to the projected cost of producing data driven automation was provided to the management team. The cost projections indicated at least a 40 percent reduction in the cost of automation and a 50 percent reduction in maintenance costs.

The next marketing effort was to present this method to the test staff for consideration. They would be impacted, their test designs would change and their planning would be different. There would be benefits specific to the test analyst, since the test data resided in the .CSV files, they had control over how much of a test would be run, which data would be used, and now had the ability to add or delete data as desired without the assistance of the automation team.

Several projects were selected for betas, one of which ran the same test with ten thousand lines of data. The results were convincing and positive. Data driven automation is the preferred method of automation whenever it is practical.

The cost reductions are no longer projections; it has been possible to reduce the size of the automation team by eliminating contractor resources and re-assigning team members to test staffs. This reduced the team's overhead and added highly skilled test resources to several test teams.

Over the ten-month period from September 2001 to July 2002 the maintenance percentage for the team was 17 percent. A tribute to the smaller scripts and the methods employed when creating automation.

In addition to these changes in the method of automation, changes to components of the process were implemented. A Project Status DB was created which replaced the Excel spreadsheet as a tracking tool. The DB can instantly provide current team assignments; projects in progress, projects expected, projects completed, and offers the ability for the user to do Ad Hoc queries.

A Script Result Reporter DB was created that accepts the automated test results from the automation tool. The tool writes to the DB and those results are available to the user in the form of an Excel spreadsheet or a Word document.

Sizing and Priority algorithms and tools were improved. A project estimation tool was built to use during the Test Assessment phase of planning.

It is projected that by the end of this calendar year, three members of the former nine-member automation team will remain as a center of expertise and that the majority of the maintenance will be shifted to the test staff. The first training session has been conducted for a group of test analysts in accepted methods of script repair. A standards document was provided as a guide.

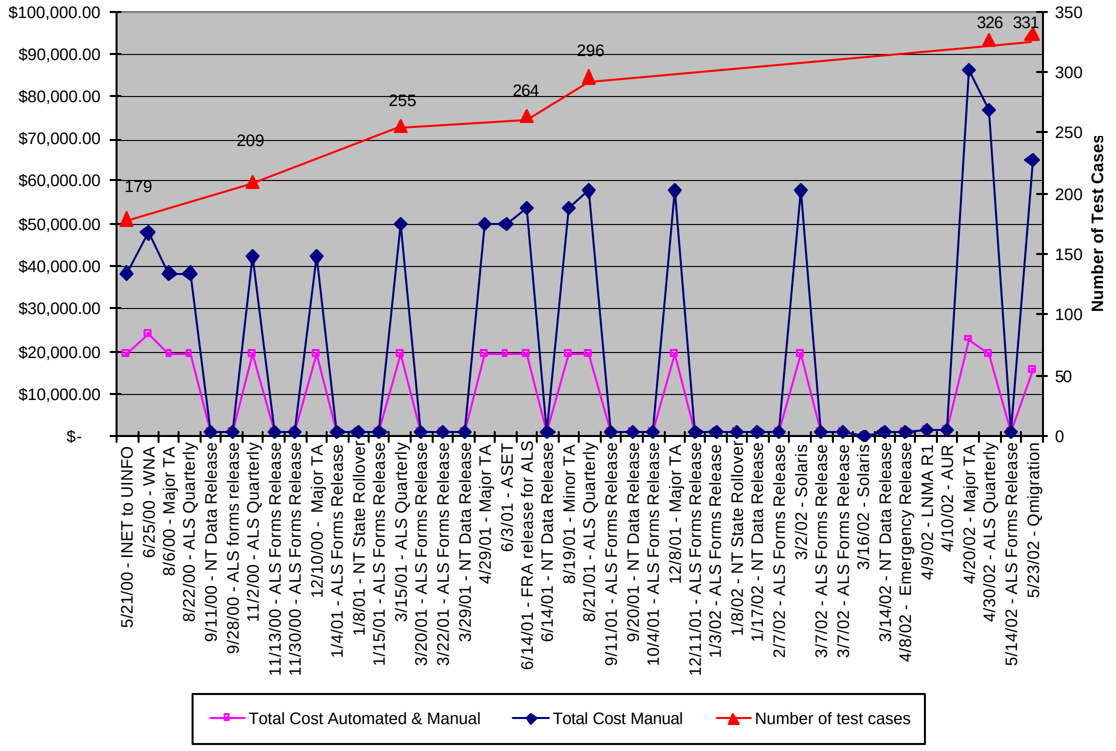
Measuring the Results

Several analysis efforts over the past eighteen months have revealed significant contributions towards improving the test process and moving the product to market more quickly.

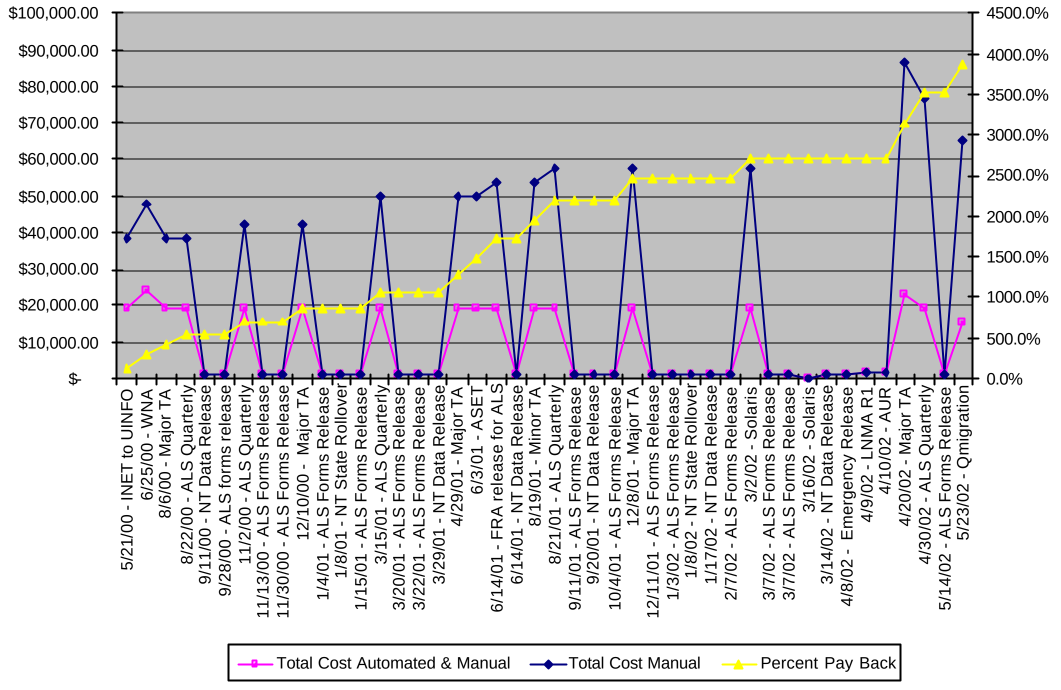
Other efforts have analyzed the cost of automation as compared to the benefits derived and those results were positive. As the process grew and was refined, the efficiency grew and the size of the staff was reduced, the time required to complete the task was reduced. And of course, costs were reduced.

These charts are a depiction of the cost of testing of one group of products over a span of a year, it displays the cost of testing with and without automation, shows increased test scope, and identifies the perceived payback percentages during the test process.

Functional Test Cost For ALS Projects
(Automated & Manual vs. Manual Only)
May 21, 2000 through May 23, 2002)



Percent Pay Back For Automating ALS Functional Tests
(May 21, 2000 through May 23, 2002)


$100,000.00
$90,000.00
$80,000.00
$70,000.00
$60,000.00
$50,000.00
$40,000.00
$30,000.00
$20,000.00
$10,000.00
$-
5/21/00 - INET to UINFO
6/25/00 - WNA
8/6/00 - Major TA
8/22/00 - ALS Quarterly
9/11/00 - NT Data Release
9/28/00 - ALS forms release
11/2/00 - ALS Quarterly
11/13/00 - ALS Forms Release
11/30/00 - ALS Forms Release
12/10/00 - Major TA
1/4/01 - ALS Forms Release
1/8/01 - NT State Rollover
1/15/01 - ALS Forms Release
3/15/01 - ALS Quarterly
3/20/01 - ALS Forms Release
3/22/01 - ALS Forms Release
3/29/01 - NT Data Release
4/29/01 - Major TA
6/3/01 - ASET
6/14/01 - FRA release for ALS
6/14/01 - NT Data Release
8/19/01 - Minor TA
8/21/01 - ALS Quarterly
9/11/01 - ALS Forms Release
9/20/01 - NT Data Release
10/4/01 - ALS Forms Release
12/8/01 - Major TA
12/11/01 - ALS Forms Release
1/3/02 - ALS Forms Release
1/8/02 - NT State Rollover
1/17/02 - NT Data Release
2/7/02 - ALS Forms Release
3/2/02 - Solaris
3/7/02 - ALS Forms Release
3/7/02 - ALS Forms Release
3/16/02 - Solaris
3/14/02 - NT Data Release
4/8/02 - Emergency Release
4/9/02 - LNMA R1
4/10/02 - AUR
4/20/02 - Major TA
4/30/02 - ALS Quarterly
5/14/02 - ALS Forms Release
5/23/02 - Qmigration
Total Cost Automated & Manual
Total Cost Manual
Percent Pay Back