BIO

PAPER

W12

November 6, 2002
3:15 PM

"EXCEL-ERATING" TEST STATUS REPORTING

Earl Burba and Jim Hazen
SysTest Labs

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

Earl Burba

Earl has over 20 years experience in software development and testing. Earl is a Senior Project Manager with SysTest Labs in Denver, Colorado.

Earl has worked in various positions within industry. Earl's experience includes flight systems, telephony, database, business and real-time embedded systems. Earl is an award winning author, and holds patents on software testing tools and methodologies.

Jim Hazen

Jim has 15 years experience in software development and testing. Jim is a Test Engineer with SysTest Labs in Denver, Colorado. Jim is a Certified Software Test Engineer through the Quality Assurance Institute.

Jim has worked in various capacities as a test lead / project manager, automation developer, and department manager. He has worked with a wide variety of applications on DOS, OS/2, Windows, and Web environments.

‘Excel’erating Test Status Reporting

Jim Hazen & Earl Burba, SysTest Labs
303.575.6881 jhazen@systest.com eburba@systest.com

Extended Abstract:

In this presentation, we will demonstrate a method to implement a test status reporting tool and strategy that uses Microsoft Excel. The presentation will address the use of Excel as a tool to develop, manage, and report the testing status for a testing effort during a project. We will introduce the Excel features required for developing a test status and reporting tool. We will also demonstrate when and how to build worksheets for test cases, summary reports, and status reports. Additionally, we will discuss the benefits of effective test status reporting and how Excel can support effective reporting. Attendees will come away with a working model for using Excel for test status reporting.

1.0 Introduction

As testers we are often asked how far along the testing effort is and when it will be done. These can be very difficult questions to answer, and usually the most nerve racking, especially when the testing effort for a project is just starting up, or close to being finished. A process is needed to help gather information and effectively report on the status of testing. The problem is that a lot of companies cannot afford a complex commercial tool due to financial reasons, or time constraints to evaluate and implement the tool.

A solution is available using commercial spreadsheet products, specifically Microsoft Excel. Using the logic and formula functions along with a combination of linked worksheets, an easy-to-implement and usable test status reporting tool can be built. This paper will present the following:

- Microsoft Excel and the features / functions used
- When and how to implement the tool / system
- Test Status reporting system architecture
- Test Case worksheet layout and formulas
- Test Status worksheet layout and formulas
- Report generation
- Benefits of using the tool for test status reporting
- Time needed to build and maintain the system
- Lessons learned using the tool

1.1 Microsoft Excel features and functions

Microsoft Excel is a very robust spreadsheet application. Its numerous features and functions allow the user to build simple to complex calculations and query formulas. This allows a user to gather and analyze data from numerous sources. These functions

and features (i.e., COUNTIF, IF and linking) are the building blocks for the test reporting system.

The function used most often is **COUNTIF**. This function counts the number of cells within a range that meet a given criterion. The format is: COUNTIF(D13:D38,"x"); where D13:D38 is the range and "x" is the criterion to count. In the Test Case worksheet, which we will discuss in detail later, the COUNTIF function is used to count the number of test steps that have a specific status (PASS, FAIL, N/A). This total in turn feeds into another calculation to determine if the Test Case itself is complete or incomplete, and if it passed or failed. This information is then fed into another worksheet / workbook that calculates and reports on the test status progress.

Another function used extensively is the **IF** logic construct. IF is used to conduct conditional tests on values and formulas. The **IF** function returns one value if a condition you specify evaluates to TRUE and another value if it evaluates to FALSE. The format is: IF(logical_test,value_if_true,value_if_false). An example of the statement is: IF(D11 + E11 + F11 = 0, "Not Started", IF(A11 = D11 + F11,"Passed", IF(A11 <> D11 + E11 + F11, "Not Complete", IF(E11 > 0,"Failed")))). This compound IF statement is used in the Test Case worksheet to determine the status of the test (Passed, Failed, Not Started, or Not Complete). Again this information feeds into the report analysis / generation worksheet.

The method used to tie the worksheets together is '**linking**'. The formula that is created uses the link reference to the data to be collected. This formula contains a reference to the workbook (.xls file), the worksheet, and the cell. An example of the statement is: [TestCaseTemplate.xls]IE5.x!$A$11; where "[TestCaseTemplate.xls]" is the workbook file, "IE5.x" is the worksheet name, and "$A$11" is the cell to reference. The exclamation point (!) separates the worksheet name from the cell range referred to. Excel can automatically build this link for you in just a few steps. (Look up "Create a formula to calculate data on another worksheet or workbook" in Help for the procedure.)

Finally, the typical mathematical operators (*, +, -, /) along with the other functions (SUM, MIN, MAX, etc.) are used to calculate the various statistics for reports.

Basically, the whole system works using only a few Excel functions. Sounds pretty simple and straightforward, right? It is, and that is the beauty of the system.

2.0 When and how to implement the system

Ideally the implementation of this tool methodology should occur as early in the project lifecycle as possible. The template files can be created without all of the testing task information (requirements and functionality to test, and associated test cases). If the templates are built early on, it helps in the tracking and reporting of the test case creation. This is a bonus for both the test team and project management as it helps identify the scope of the effort.

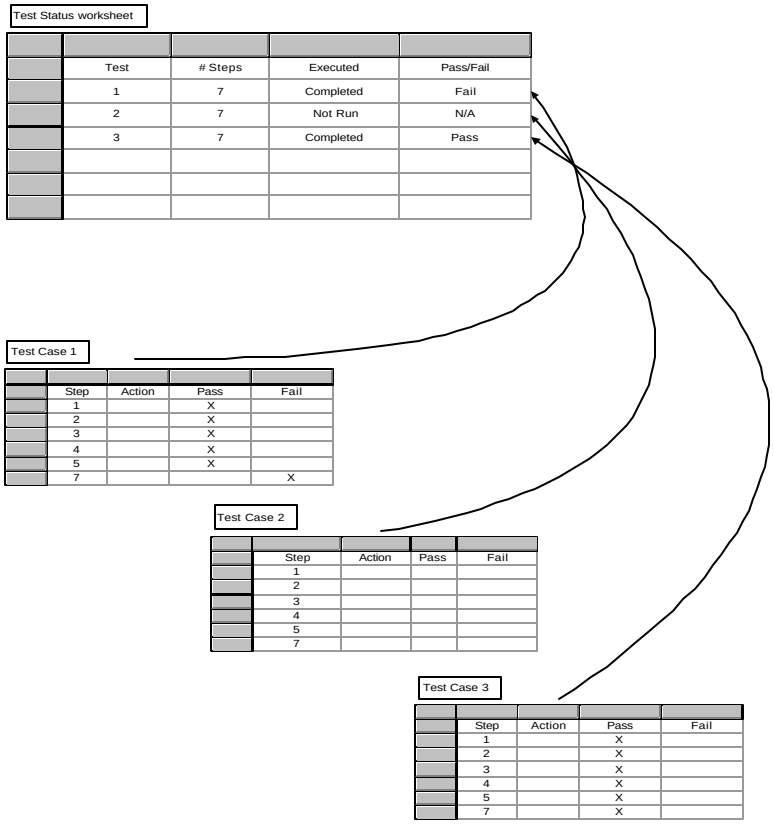
In our experience the system is implemented following the completion of the test plan.

The implementation of the system is very straightforward.

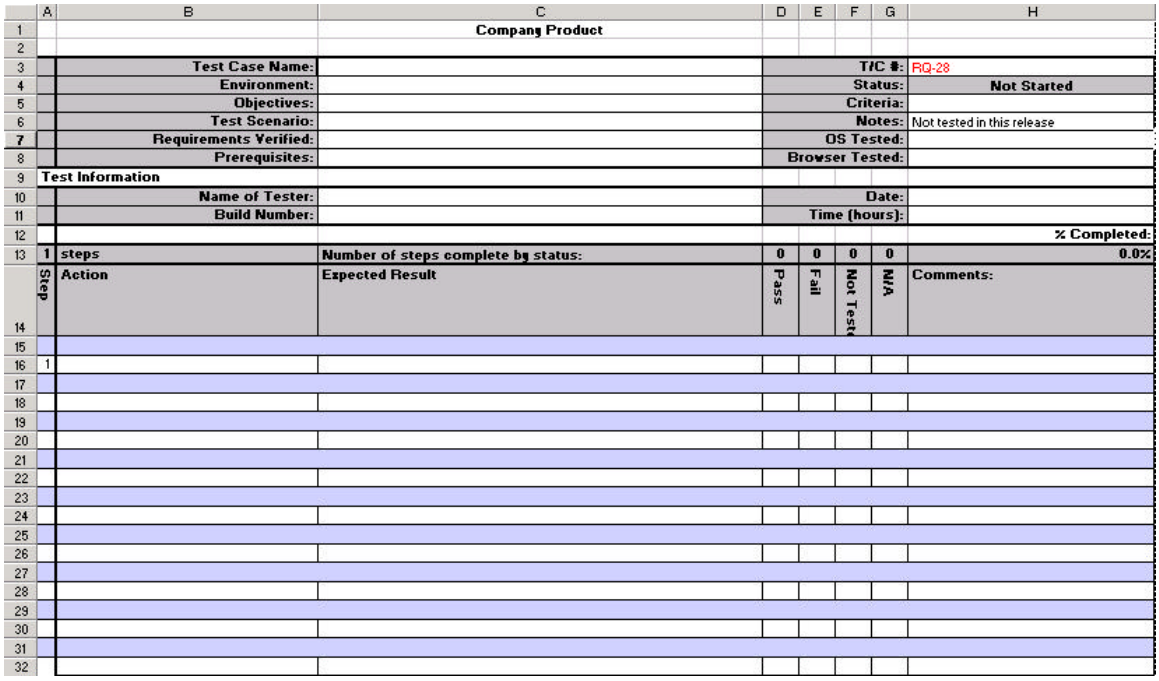
1. Create a repository on the network. This is just a directory/folder that everyone has access to. All interested parties will need access to the information.
2. Build the Test Status Tracking System for your project from the template files.
   a) The first template to be modified is the Test Case template. The template should be populated with all of the desired test cases, without modifying the formulas and functions.
   b) The next will be the Test Status template. This template should be modified to track the test cases appropriately as they are completed.
3. Link the worksheet templates together. The test cases and their data will need to be associated to the status worksheet.
4. Finally, begin building actual test cases and update/add them into the test status worksheet. Again, this is a very straightforward approach.

2.1 Test Status System Architecture

The architecture of the system is simple. Each Test Case worksheet is linked into the Test Status worksheet. The data that is recorded and calculated in the test case is pulled into the test status worksheet where it is again crunched for project level statistics reporting. Once the links are established, data can be collected and reported in any desired manner.



2.2 Test Case Worksheet

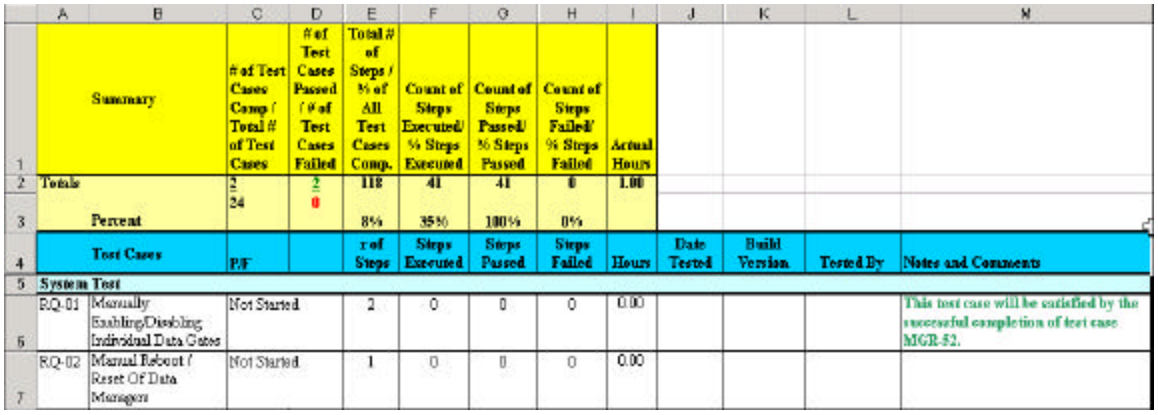


	A	B	C	D	E	F	G	H
1			Company Product					
2								
3		Test Case Name:					T/C #:	RQ-28
4		Environment:					Status:	Not Started
5		Objectives:					Criteria:	
6		Test Scenario:					Notes:	Not tested in this release
7		Requirements Verified:					OS Tested:	
8		Prerequisites:					Browser Tested:	
9	Test Information							
10		Name of Tester:					Date:	
11		Build Number:					Time (hours):	
12								% Completed:
13	1	steps	Number of steps complete by status:	0	0	0	0	0.0%
14	Step	Action	Expected Result	Pass	Fail	Not Teste	N/A	Comments:
15								
16	1							
17								
18								
19								
20								
21								
22								
23								
24								
25								
26								
27								
28								
29								
30								
31								
32								

The test case worksheet itself is basic in its format and content. Specific fields are calculated (Status, % Complete, and Num. of steps completed by status) and are pulled into the test status worksheet for further analysis and reporting. Other static fields (Test Case Name, T/C #, number of Steps, Name of Tester, Build Number, etc.) can also be pulled into the test status worksheet if desired.

The calculated field formulas are:
Status: =IF(D13+E13+G13=0,"Not Started",IF(A13=D13+G13,"Passed",IF(A13<>D13+E13+G13,"Not Complete",IF(E13>0,"Failed"))))
# of Steps : = =COUNTA(A16:A113)
Number of steps complete by status: =COUNTIF(D16:D113,"x") for Pass, =COUNTIF(E16:E113,"x") for Fail, =COUNTIF(F16:F113,"x") for Not Tested and =COUNTIF(G16:G113,"x") for N/A
% Complete: =(D13+E13+G13)/A13

2.3 Test Status worksheet



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Summary		# of Test Cases Comp / Total # of Test Cases	# of Test Cases Passed / # of Test Cases Failed	Total # of Steps / % of All Test Cases Comp.	Count of Steps Executed/ % Steps Executed	Count of Steps Passed/ % Steps Passed	Count of Steps Failed/ % Steps Failed	Actual Hours				
2	Totals		2	2	118	41	41	0	1.00				
3		Percent	24	0	8%	35%	100%	0%					
4		Test Cases	P/F		r of Steps	Steps Executed	Steps Passed	Steps Failed	Hours	Date Tested	Build Version	Tested By	Notes and Comments
5	System Test												
6	RQ-01	Manually Enabling/Disabling Individual Data Gates	Not Started		2	0	0	0	0.00				This test case will be satisfied by the successful completion of test case MGR-52.
7	RQ-02	Manual Reboot / Reset Of Data Managers	Not Started		1	0	0	0	0.00				

The test status worksheet has numerous calculated fields (# of Test Cases Comp. / Total # of Test Cases, # of Test Cases Passed / # of Test Cases Failed, Total # of Steps, Count of Steps Passed, Count of Steps Failed, % Test Cases, % Steps Passed, % Steps Failed, and P/F) and others that contain information pulled from the test case worksheet (Number of Steps, Steps Executed, Steps Passed, Steps Failed, Date Tested, Bld Ver.). Additional information can be pulled in from the test case worksheet (Tested By). This information can include whatever is important to status reporting.

The calculated fields formulas are:

# of Test Cases Comp: =COUNTIF(C6:C37,"Passed")+COUNTIF(C6:C37,"Failed")
Total # of Test Cases: =COUNTA(C6:C37)
# of Test Cases Passed: =COUNTIF(C6:C37,"Passed")
# of Test Cases Failed: =COUNTIF(C6:C37,"Failed")
Total # of Steps: =SUM(E6:E37)
Count of Steps Executed: =G2+H2
Count of Steps Passed: =SUM(G6:G37)
Count of Steps Failed: =SUM(H5:H37)
Actual Hours: =SUM(I6:I37)
Percent of All Test Cases Comp.: =C2/C3
Percent of Steps Executed: =F2/E2
Percent of Steps Passed: =IF(F2 > 0,G2/F2,0)
Percent of Steps Failed: =IF(F2 > 0,H2/F2,0)
Steps Executed: =SUM(G6:H6)

The test case linked fields formulas are:

Passed/Failed: =’[TestCaseTemplate.xls]RQ-01'!$H$4
Number of Steps: ='[TestCaseTemplate.xls]RQ-01'!$A$13
Steps Passed: ='[TestCaseTemplate.xls]RQ-01'!$D$13
Steps Failed: ='[TestCaseTemplate.xls]RQ-01'!$E$13
Hours: ='[TestCaseTemplate.xls] RQ-01'!$H$11
Date Tested: =IF('[TestCaseTemplate.xls]RQ-01'!$H$10 <> “”,'[TestCaseTemplate.xls]RQ-01'!$H$10, "")
Build Version: =IF('[TestCaseTemplate.xls] RQ-01'!$C$11 <> “”,'[TestCaseTemplate.xls]RQ-01'!$C$11, "")

In each of the calculations (local to the sheet or the link to the test case worksheet), the cell ranges (i.e., C7:C8) can vary in number (range) depending on how the spreadsheets are built. Again, the “[TestCaseTemplate.xls]” is the workbook file, “RQ-01” is the worksheet name, and “$A$11” is the cell to reference. Notice how the IF statement is

used to either get data from the test case worksheet (as in Build Version) or calculate a value based on other cells in the status worksheet.

Presented here is the core functionality, and it can be expanded upon based on need.

This information can also be pulled into another worksheet. In the case of compatibility testing, another worksheet can show which platforms (Software version, Operating System, Browser version/type, etc.) have been tested and their status. To do this, duplicate the test case worksheets for each platform (i.e., browser version and type) in that test case workbook. Then, in the test status workbook, create summary worksheets for each platform. Finally, create a summary worksheet that pulls its information from the platform worksheets. Following is an example in which each column is the platform being tested.

Test Case Status of Primary Platforms		
	# of Test Cases Comp. / Total # of Test Cases	# of Test Cases Passed / # of Test Cases Failed
Totals	**2** **2**	**2** **0**
Percent	**100%**	**100%**
Completion	**IE5.5**	**NS4.7**
Test1	**Passed**	**Passed**

From here any type of report (tabular or chart) can be generated.

2.4 Test Planning

	A	B	C	D	E	F	G	H	I	J	K	L
1	Proc.	Test Description	Requirement	# of Steps	Est. Hours	Actual Hours	Planned Date	Actual Date	P/F	Tester	Notes	
38	SC-01	On-Screen Help/Page Help							Skipped		SKIP - Customer request	
39	SC-02	Error Status Reporting/Display		24	4.5	0.50	6/6/2002	6/5/2002	Passed	Jesse Peterson		
40	SC-03	Manage Modules							Skipped		Functionality tested in MGR-S2	
41	SC-04	Save & Restore		8	1.5	1.00	6/7/2002	6/7/2002	Failed	Rex Reed		
42	SC-05	Multiple Users		7	1.5	0.5	6/6/2002	6/10/2002	Passed	Rex Reed		
43	SC-06	Initialize Arrays							Skipped		SKIP - Customer request	
44	SC-07	Infoshield Testing		2	0.5		6/17/2002		Not Started		Infoshield functionality not present in this release per Release notes ERRATA Manage > LUN Management > Partition Menu > Infoshield: Not implemented This feature enabled in Firmware	
45	SC-08	Status Bars Check							Skipped		SKIP - Customer request	
46	SC-09	Fibre Channel Port Status							Skipped		SKIP - Customer request	

After the test cases have been created and the scope of effort more clearly understood, it is important to plan the testing effort. This is usually constrained by start and end dates and staffing resources (testers). It is important to assign resources reasonably based on experience and abilities. The planning spreadsheet is used to assign dates and resources to each test case previously written. This spreadsheet is the basis for metrics and tracking the test effort progress.

Following are the linked values and/or formulas:
Proc.: ='[Scenario Test Cases_san.xls]SC-02'!$H$3
Test Description: ='[Scenario Test Cases_san.xls]SC-02'!$C$3
# of Steps : ='[Scenario Test Cases_san.xls]SC-02'!$A$13
Est. Hours: =IF(D39 <> 0,IF((D39/6)-ROUNDDOWN(D39/6,0) < 0.5,0.5,1) + ROUNDDOWN(D39/6,0),0)
Actual Hours: =IF('[Scenario Test Cases_san.xls]SC-02'!$H$11 <> "", '[Scenario Test Cases_san.xls]SC-02'!$H$11, "")
Planned Date: Entered by test manager or project manager
Actual Date: =IF('[Scenario Test Cases_san.xls]SC-02'!$H$10 <> "",'[Scenario Test Cases_san.xls]SC-02'!$H$10,"")
P/F: =Scenarios!C7
Tester: =IF('[Scenario Test Cases_san.xls]SC-02'!$C$10 <> "",'[Scenario Test Cases_san.xls]SC-02'!$C$10,"")
Notes: =IF('[Scenario Test Cases_san.xls]SC-02'!$H$6 <> "",'[Scenario Test Cases_san.xls]SC-02'!$H$6,"")

2.5 Master Test Status Tracking

	A	B	C	D	E	F	G	H	I	J	K	L
1												
2		*Customer System Testing* *June 1, 2002 - June 26, 2002*	**Total Number of Test Cases**	**Number of Test Cases Passed**	**Number of Test Cases Failed**	**Number of Steps**	**Actual Hours**					
3												
4		**Requirements TCs**	24	2	0	118	1.0					
5		**Scenarios**	17	11	5	163	27.0					
6		**Monitor**	21	15	6	233	13.4					
7		**Manage**	50	35	15	279	44.9					
8		**Total**	**112**	**63**	**26**	**793**	**86.2**					
9												
10		**System Test**										
11		**Daily Totals**	3-Jun		4-Jun		5-Jun		6-Jun		7-Jun	
12			Passed	Failed	Passed	Failed	Passed	Failed	Passed	Failed	Passed	Failed
14		**Requirements TCs**										
15		**Scenarios**							1		1	2
16		**Monitor**					1	1	1	3	1	3
17		**Manage**						4		4	4	4
18		**Total**					**1**	**5**	**2**	**7**	**6**	**9**
19												
20		**System Test**										
21		**Daily Totals**	10-Jun		11-Jun		12-Jun		13-Jun		14-Jun	
22			Passed	Failed	Passed	Failed	Passed	Failed	Passed	Failed	Passed	Failed
24		**Requirements TCs**	0	0								
25		**Scenarios**	2	2	3	3	5	3	8	4	8	5
26		**Monitor**	8	3	14	4	15	6	15	6	15	6
27		**Manage**	9	4	22	2	26	6	30	10	33	11
28		**Total**	**19**	**9**	**39**	**9**	**46**	**15**	**53**	**20**	**56**	**22**
29												
30		**System Test**										
31		**Daily Totals**	17-Jun		18-Jun		19-Jun		20-Jun		21-Jun	
32			Passed	Failed	Passed	Failed	Passed	Failed	Passed	Failed	Passed	Failed
34		**Requirements TCs**	2	0								
35		**Scenarios**	10	5								
36		**Monitor**	15	6								
37		**Manage**	35	15								
38		**Total**	**62**	**26**								

During the course of testing, daily statuses are normally reported to track the progress of the testing effort. The spreadsheet above is an example of how those metrics can be gathered. The top cells (B1:G8) are the cumulative numbers of test cases executed. These are automatically populated through linking to other spreadsheets.

The linked formulas for these cells are:
Total Number of Test Cases: =Requirements!$C$3
Number of Test Cases Passed: =Requirements!$D$2
Number of Test Cases Passed: =Requirements!$D$3
Number of Steps: =Requirements!$E$2
Actual Hours: =Requirements!$I$2

The **Totals** are calculated for each column: i.e., =SUM(C4:C7)

The daily totals are compiled by entering the numbers from the top cells at the end of each testing day. These are copied and pasted by the test manager or project manager. This cannot be automated since the cumulative totals change each day.

2.6 Test Metrics

	A	B	C	D	E	F	G	H	I	J	K
1											
2		**a - Actual # Cases**	**90**	66%	a/f			**Week 1**			
3		**b - # of cases Executed**	89	99%	b/a			Planned	Actual	Cumulative Planned	Cumulative Actual
4		**c - # of cases Passed**	63	70%	c/a	Mon	6/3/2002	0	1	0	1
5		**d - # of cases Failed**	26	29%	d/a	Tues	6/4/2002	0	1	0	2
6		**e - # of cases Pending**	0	0%	e/a	Wed	6/5/2002	0	1	0	3
7						Thurs	6/6/2002	8	4	8	7
8		**f - # of cases Planned**	**137**			Fri	6/7/2002	15	9	23	16
9		**g - # of cases Not Executed**	1	1%	g/a		Total	23	16		
10		**h - # of cases Skipped**	47	34%	h/f						
11		**i- # of cases Added**	0	0%	i/f			**Week 2**			
12								Planned	Actual	Cumulative Planned	Cumulative Actual
13						Mon	6/10/2002	12	12	35	28
14						Tues	6/11/2002	10	22	45	50
15						Wed	6/12/2002	12	13	57	63
16						Thurs	6/13/2002	12	10	69	73
17						Fri	6/14/2002	8	5	77	78
18							Total	54	62		
19											
20								**Week 3**			
21								Planned	Actual	Cumulative Planned	Cumulative Actual
22						Mon	6/17/2002	8	9	85	87
23						Tues	6/18/2002	5	1	90	88
24						Wed	6/19/2002	0	0	90	88
25						Thurs	6/20/2002	0	0	90	88
26						Fri	6/21/2002	0	0	90	88
27							Total	13	10		
28											
29							**Grand Total**	**90**	**88**		

Another task of the test manager or project manager is to report certain metrics to the customer, client or interested parties. These can be customized to the project's needs and are based on the testing efforts and planned dates for testing efforts.

The formulas for the metrics shown are:
a – Actual # Cases: =C8 - C10 + C11
b - # of cases Executed: =C4 + C5 + C6
c - # of cases Passed: =COUNTIF('Test Plan'!I:I, "Passed")
d - # of cases Failed: =COUNTIF('Test Plan'!I:I, "Failed")
e - # of cases Pending: =COUNTIF('Test Plan'!I:I, "Not Complete")
f - # of cases Planned: =SUM(C10+H9+H18+H27)
g - # of cases Not Executed: =COUNTIF('Test Plan'!I:I, "Not Started")
h # of cases Skipped: =COUNTIF('Test Plan'!I:I, "Skipped")
i- # of cases Added: =COUNTIF('Test Plan'!I:I, "Added")

The percentages in the next column are calculated using the following formulas:
Percentage of Actual Cases: =IF(C8 > 0,C2/C8,0%)
Percentage of cases Executed: =IF(C2 > 0,C3/C2,"0%")
Percentage of cases Passed: =IF(C2>0,C4/C2,"0%")
Percentage of cases Failed: =IF(C2>0,C5/C2,"0%")
Percentage of cases Pending: =IF(C2>0,C6/C2,"0%")
Percentage of cases Not Executed: =IF(C2>0,C9/C2,"0%")
Percentage of cases Skipped: =C10/C8
Percentage of cases Added: =C11/C8

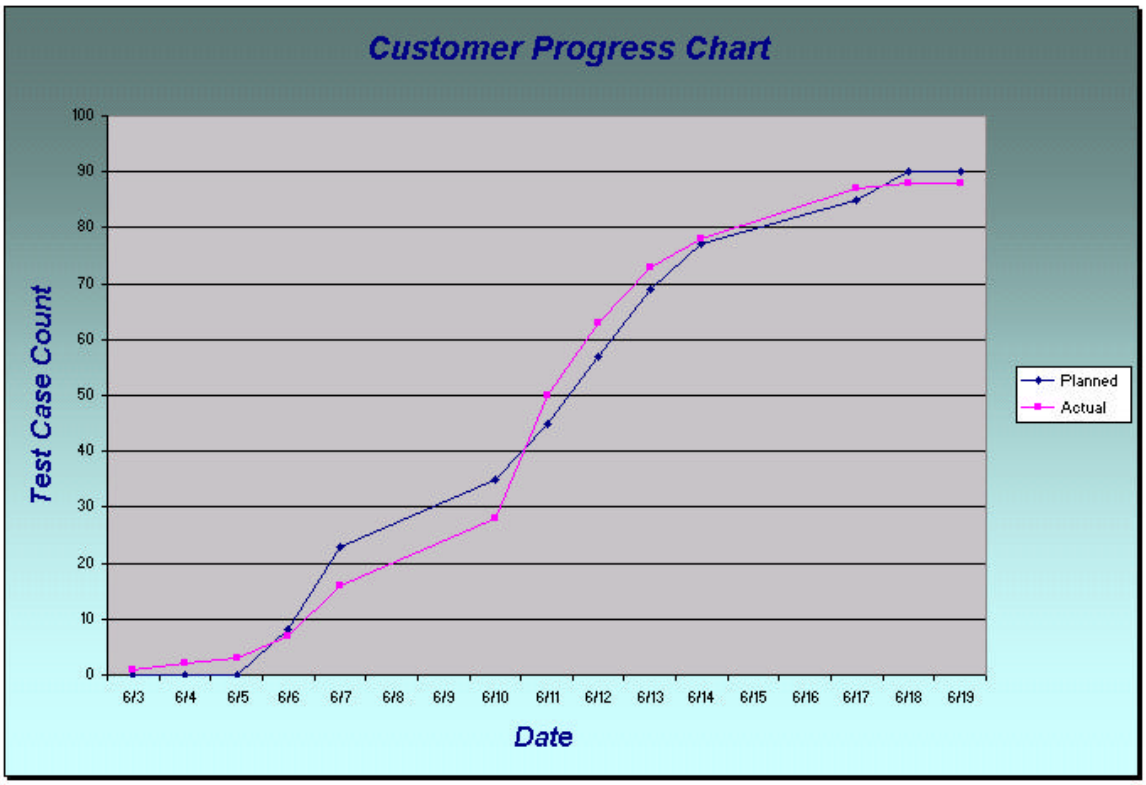
The second part of this spreadsheet is the Planned vs. Actual test case execution metric. This metric reports how close the project is to being on its planned target. If the numbers of test cases being executed are not within tolerance bands, management should be notified immediately. Also, this set of metrics can be used for future planning as historical evidence (i.e., how many modules of this type can potentially be tested per day). The dates in the formulas for the Planned and Actual columns are entered based on the testing period and must be entered appropriately for the task. It should be noted that these numbers will be used in the customized graphical charts that the client, customer and/or management sees (Section 3.0 Report Generation).

The formulas for Week 1 day 6/7/2002 of our example are as follows:
Planned: =COUNTIF('Test Plan'!G:G, "6/7")
Actual: =COUNTIF('Test Plan'!H:H, "6/7")
Cumulative Planned: =SUM(J7+H8)
Cumulative Actual: =SUM(K7+I8)

3.0 Report Generation

The test summary spreadsheets themselves are used as the standard reports. Most people (management and other groups outside of Test/QA) want to know how many tests passed and failed, how many tests have been executed, and how many are left to execute. They may also want to know who executed them or who is assigned to execute them. This last item helps in keeping tests assigned to staff and ensuring that the tests are being executed in a timely fashion. There is nothing worse than at the end of a project realizing an important test was not executed because it fell through the cracks, or that an important platform was not tested.

The reports can also be converted into charts for a visual view. Bar, Line, and Pie charts can be used to show status. Line charts show progress over time. This can include tests executed per week, or tests passed vs. failed per week, or number of test cases written vs. total to be written (this last one is done early in the project lifecycle and is beneficial in that it shows readiness of the Test group). Pie charts can be used to show status for coverage. This can include percent test cases executed vs. not executed, or percent platforms tested vs. not tested. Either of these can then be combined with defect status reports to show effectiveness of testing (number of new defects found per week that are attributable to test case execution, meaning that any ad-hoc testing defects are left out and just the ones found by the test cases are considered). Or it can be used to show system stability (more defects are closed than found in a week for the amount of test cases executed). The example line chart shown below depicts test case execution status.



4.0 Benefits of using the tool

Although there is some initial preliminary setup required and maintenance is needed to allow for changes in testing scope, many benefits can be realized by implementing this type of method.

- This method allows higher accuracy of reporting testing status. Because the tool is automated and the format is standardized, the chance for miscalculation is minimized, with no direct interaction from tester.
- It provides earlier warning if the project / test effort is in trouble or off-track.
- Individual or project progress can be tracked, allowing for better estimates of completion dates, percentages complete, rates of failure etc.
- Results can easily be incorporated into daily and project reports in other documents. This saves time and effort on the part of the test manager and other interested parties.
- The data is in a centralized repository and is more accessible.
- It saves time and money. Because Excel is part of Microsoft Office, it is readily available, and the cost of the system is in the time spent to build it. No other software is needed. There isn't any ramp up time to learn or develop a third party tool.

5.0 Summary

Using Excel as an alternative to a third party tool is a viable solution for companies who do not have the time or the budget to implement other tools. Utilizing its built-in functions and formulas, and a little bit of inspiration, a customized and flexible test status reporting system can be built. With the information presented here, anyone can jumpstart the implementation process.

We have found, out of necessity, that proper test management and status reporting are vital to the success of a project. By using the test status system we have saved ourselves time and money, increased our reporting accuracy, and kept our testing projects under control.

Author(s) Biography:

Jim Hazen - Jim has 15 years experience in software development and testing. Jim is a Test Engineer with SysTest Labs in Denver, Colorado. Jim is a Certified Software Test Engineer through the Quality Assurance Institute.

Jim has worked in various capacities as a test lead / project manager, automation developer, and department manager. He has worked with a wide variety of applications on DOS, OS/2, Windows, and Web environments.

Earl Burba – Earl has over 20 years experience in software development and testing. Earl is a Senior Project Manager with SysTest Labs in Denver, Colorado.

Earl has worked in various positions within industry. Earl's experience includes flight systems, telephony, database, business and real-time embedded systems. Earl is an award winning author, and holds patents on software testing tools and methodologies.