Calculating your Test Overheads, Part 1

Ross Collard
rcollard@attglobal.net

Do you remember receiving your very first paycheck? That initial happy feeling which turned to confusion when you saw that your expected $100 payment had shrunk to $28 after oodles of deductions? Welcome to the world of overheads. When does a two-hour testing task take two days? When a manager, who is not very close to the work and is informal in his or her analysis of situations, casually observes only the visible part of the work. The conspicuously visible part is a tester tapping on the keyboard, running the test cases, and muttering occasionally in deep thought, excitement or bewilderment. (Testers can get passionate about their work.)

The test execution takes two hours, but is the tip of the iceberg. The overhead for all the support activities often is much larger than the visible part. These support activities include everything from setting up test equipment to writing problem reports. (It is better not to label these activities as Aoverhead@ because they are B or should be B a natural, integral part of the test process.) One secret of realistic estimating is to identify all of the contributions needed to complete an activity. This point seems obvious, but it is often overlooked.

Calculating Your Direct Test Overhead (Allow 20 to 30 minutes for this exercise.)
Facing reality, and analyzing where our time is going today, is an important step in realistic estimating. The purpose of this exercise is to calculate how long you and by extension your team are really spending on testing activities, compared to the obviously visible part of testing.

Consider a typical system test project in which you have been involved recently. This project should have contained at least one task where you were assigned the job of manually testing a feature or a set of related features, and you tested them primarily by yourself. Instead, if a team of three people jointly tested the features, is harder to analyze the other people's time than your own. In your time accounting, do not include automated testing tasks or non-feature testing tasks (e.g., load testing or configuration testing). For every two hours that you spent at the keyboard, manually running the feature test cases and doing nothing else, approximately how long did you spend on the other tasks listed here?

What if You don't have Testing Experience?
So far in this exercise, I have assumed you have considerable hands-on testing experience, in other words, you are personally quite familiar with the work content on a test project. A solid base of real-world experience is a powerful tool for an estimator. But not everybody in our audience will necessarily have that background. You may be a new test professional but still be expected to produce estimates, or an experienced tester who has moved into a new and different work environment, or an experienced project manager or business professional who's never personally spent much time testing.

If you do not have direct personal testing experience, I would like you to do this exercise from a

different perspective. Instead of remembering your own experience and recording how time is being spent on your testing projects, write down the amount of overhead which you think *should be allowed* in each category. Your common sense and general background knowledge are enough to give you a sense of what's reasonable. Later, compare your answers with the testers in your organization. Your points of agreement and disagreement will be illuminating.

What if every Project is Different?
Some people live in rapidly evolving and even chaotic worlds. There is little continuity of how their projects are done B the only constant is that software development, testing and maintenance methodologies are always changing. It is difficult to know how and where the patterns of past behaviors will apply to the future. If you are in this situation, the exercise is still doable but harder. For each overhead item listed below, provide a range of three numbers: your most likely, most pessimistic (highest) and most optimistic (lowest) guesses. For those overhead factors which you consciously or unconsciously feel are fairly stable, the spread among the three numbers will be low.

When the spread is wide, indicating considerable uncertainty, you can take the average of the three numbers and mark it up by a generous (25% to 50%) factor of safety, to compensate for the risk introduced by the uncertainty. This is not fudging the numbers, this is prudent risk management. (If the boss buys this justification, tell him about the need to restore the testers' bug hunting powers with a rest trip to Hawaii.) Seriously, though, adding a generous factor of safety to an item which has high uncertainty is a good practice.

Customization Pre-Requisite
The tasks listed here will not exactly fit how you do things. You may need to modify the list of tasks to match your approach before you collect and enter the data.

Testing Support Task	**Time Consumed** for each two hours of hands-on testing (Approximate hours)
A. Test Preparation	
Participate in a meeting with the team leader to coordinate your activities, or with the test team to coordinate everyone's activities, and to receive your next work assignment: to test the set of features	_____
Review the functional specs. or an equivalent description of the relevant feature(s) B this includes verbally quizzing the developers or the users about the feature	_____
Resolve questions about the specs. and the expected behavior of the feature and the system	_____

Review the pertinent parts of the test plan _____

Learn about any changes to the system functionality and test plan since you last accessed them, and their impact on how to test the feature _____

Review problem reports which have been fixed and are ready for testing _____

Explore the feature(s) on-line, or a prototype of the feature _____

Speak with the software engineers to resolve questions about the feature implementation _____

Develop the test requirements, if these do not already exist _____
(Or: review and clarify the test requirements, if they already exist)

Search for existing test cases which can be adapted and reused _____

Adapt the existing test cases as needed for this use _____

Develop additional new test cases (where no existing ones can be adapted) _____

Develop the test data needed by the test cases, if these do not already exist _____
(Or: review and if necessary update the test data files, if they already exist)

Gather the equipment and software needed for the testing _____

Set-up and check-out the test facilities _____

Integrate or build the version of the system to be tested _____

Establish version control or configuration management control over the new system version _____

Load the system version we are testing into the test environment _____

Load the data files and databases needed for this test _____

Establish version control or configuration management control over the loaded test environment _____

Review and become familiar with the testware, test cases and procedures _____

Confirm everything is configured correctly, run a smoke test and verify that you can proceed with the testing _____

Total for Test Preparation: _____ hours

B. Test Execution and Follow-up

Execute the test cases manually 2.0 hours

Evaluate the test results _____

Re-run test cases to replicate and confirm the system's behavior _____

Trouble-shoot, debug and fix the testware _____

Confer with the software engineers about the symptoms found (i.e., the test case failures) and possible bugs _____

Document the test results log _____

Write the problem report(s) _____
Enter data into the bug tracking system _____

Monitor the problem resolution and the availability of the fix to be re-tested _____

Re-test to confirm the fix works _____

Regression test to find unintended sides effects of the fix _____

Document and archive the test results _____

Restore the test environment to a known pristine condition ready for the next test suite _____

Break down the test environment and return hardware used (cables, computers, etc.) _____
Confer with the boss on the project status and direction _____

Update the test plan and test case libraries, if necessary _____

Take a well deserved break _____

Total for Test Execution and Follow-up: _____ hours

Unplanned Interruptions during the Work _____

Total including All Activities : _____ hours

One Test Team's Answer to this Exercise

When they took the time to analyze where their time being spent, a test team with a large multinational bank found that they spent approximately two days of effort in all, for each visible two hours of test execution. The team's time allocations, for each two hours of manual test execution, were as follows:

Testing Task	**Time Consumed** (Hours)
A. Test Preparation	
Participate in a test team meeting to coordinate the team's activities and to receive your next work assignment	0.75
Review the functional specs. or equivalent description of the feature or system	0.5
Resolve questions about the specs. and the expected behavior	0.5
Review the pertinent parts of the test plan and test procedures	0.50
Learn about the changes to the system functionality and test plan since you last accessed them	0.25
Review problem reports which have been fixed and are ready for testing	0
Explore the feature(s) on-line, or a prototype of the feature	0
Speak with the software engineers to resolve questions about the feature implementation	0
Develop the test requirements, if these do not already exist	0.5
(Or: review and clarify the test requirements, if they already exist)	
Search for existing test cases which can be adapted and reused	0.5
Adapt the test cases for the new use	1.0
Develop, document and review additional new test cases (where no existing ones can be adapted to meet the need)	1.50
Develop the test data needed by the test cases, if these do not already exist	0.5
(Or: review and if necessary update the test data, if they already exist)	
Gather the equipment and software needed for the testing	0.00
Set-up and check-out the test facilities	0.25

Integrate or build the version of the system we are testing	0.00	
Establish version control or configuration management control over the new system version	0.00	
Load the system version we are testing into the test environment	0.25	
Load the data files and databases needed for this test	0.25	
Establish version control or configuration management control over the loaded test environment	0.25	
Review and become familiar with the testware, test cases and procedures		
Confirm everything is configured correctly, run a smoke test and verify that you can proceed with the testing	0.17	
Total for Test Preparation		7.67 hours

B. Test Execution and Follow-up

Execute the test cases manually	2.00	
Evaluate the test results	1.00	
Re-run test cases to replicate and confirm the system's behavior	0.33	
Trouble-shoot, debug and fix the testware	0.20	
Confer with the software engineers about the symptoms found (i.e., the test case failures) and possible bugs	0.33	
Document the test results log	0.07	
Write the problem report(s)	0.50	
Enter data into the bug tracking system	0.10	
Monitor the problem resolution and the availability of the fix we are testing	0.10	
Re-test to confirm the fix works	0.33	
Regression test to find unintended sides effects of the fix	0.20	
Document and archive the test results	0.20	
Restore the test environment to a known pristine condition ready for the next test suite	0.00	
Break down the test environment and return hardware used (cables, computers, etc.)	0.00	
Confer with the boss on the project status and direction	0.33	
Update the test plan and test case libraries, if necessary	0.17	
Take a well deserved break	1.50	
Total for Test Execution and Follow-up		7.36 hours

Unplanned Interruptions (*)	2.0	
Total Time:		***17.03 hours***

(*) Meetings not directly related to the task at hand, phone calls, consultations on other projects, etc.

Limitations of this Analysis
Unless we conduct a prohibitively expensive effort to collect and validate the data, we should not assume that numbers like the above ones are very accurate.

These numbers are rough averages because each task varies, sometimes taking no time and sometimes much more than the numbers shown.

In addition, testers have their own styles and do things in different ways, regardless of what the written test procedure is. People don't even use the same names for the same things, so achieving consistency in the timing data is problematic.

There's also the question of the care that was taken in the measurements. Imagine that the duration of each task was under-reported by 5 minutes (0.08 hour). It doesn't seem significant. However, with about 35 tasks on this list, the aggregate under-reporting is approximately 2.8 hours or 15% of the total estimate.

Conclusions from this Analysis
Since the bank managers have the impression that the testing took two hours (the visible part), that's the amount of time that they are inclined to estimate for future similar test tasks. (This test team does not have much input to the decision makers on the project budget and schedule decisions.)

How effective do you think a test team is which spends 17.03 hours to get two hours' worth of hands-on testing? This does not appear agile: 88.25% of their time is not being spent directly testing. What do you think the ideal ratio should be? Should 100% of their time be spent testing (zero overhead)? That means there is no time to plan, just mindless banging on the keyboard, and no time to evaluate and document the test results.

On the surface, test process improvement seems like a good idea for the test team at the bank, to help them become more efficient.

The impression of inefficiency is misleading, though. The test team could be highly efficient already or quite inefficient; this data does not tell us. Our conclusion of inefficiency is based on the way I presented the data: Atwo days is needed to complete two hours' work@.

Let's say I introduced the same data with the statement: ALook how much the testers can do in only two days@. I could bias the suggestible to conclude that the team is industrious.

Until the improvement is realized, though, the bank managers should use the reality-based ratio of 17.03-to-2.0 (8-to-1) if they want to accurately estimate.

Some project planners disagree B they think that allowing 88.25% of the testers' time to be spent on Aoverhead@ is condoning inefficiency, and that a goal of higher productivity should be established by setting aggressive estimates.

Perhaps targeting the overhead ratio at one-to-one will motivate the testers to work smarter and harder, but probably also lead to project schedule and budget overruns or lead to poorer-quality delivered systems.

Testing Practices

What ratio is typical of the total test effort to the hands-on test execution? In a survey which I conducted of 111 testers, their overhead ratios, in a situation where the hands-on testing took two hours, were as follows:

Ratio of Total Test Effort to Hands-on Test Execution	**Percentage of Testers**
2-to-1, or less	0%
2-to-1 to 4-to-1	6%
4-to-1 to 6-to-1	21%
6-to-1 to 8-to-1	37%
8-to-1 to 12-to-1	16%
12-to-1 to 16-to-1	10%
16-to-1 to 20-to-1	5%
20-to-1 or more	5%

Follow-Up Exercises

Repeat the process of calculating your test overhead for each of these circumstances which happen on your projects:

(1) A test of a new system which is done manually.
(2) A test of a new system which is automated, not manual.
(3) A test of a change to an existing system which is done manually.
(4) A test of a change to an existing system which is automated, with an existing regression test facility and automated test case libraries.
(5) A test of a system's performance characteristics and ability to hasndle load, which does not include functional testing.

Comparing Ratios with the Managers

Perhaps the most important exercise is to compare people's perceptions with reality. Do your managers know what your overhead ratios are? If you are unsure, ask them to give you their sense of the ratios. If they are significantly different from your own numbers from this exercise, the differences may be worth a follow-up discussion.

Identifying Inefficiencies

We can also use the work sheets above to ask: AWhere did all the time go?@ Often, surprisingly large amounts of time are frittered away of Alittle@ inefficiencies. Analyzing how the test team's time was actually spend is an important step in streamlining the test process.

When I helped a test team in a financial services organization analyze how their time was being spent, we were amazed to find that thousands of hours per year were being spent trying to find existing test cases, figure out what they did, and assess whether they could be used on new projects. Once the inefficiency was apparent, the solution was obvious B to improve the test data management and build an organized, indexed repository of test cases. The need was not apparent without the time analysis, though. Some testers assumed that the existing inefficient methods were the ways things had to be. Others were so busy getting the work done that they had not noticed where their time was being spent.