# Using Earned Value
# Part 2: Tracking Software Projects

## Abstract

We've all experienced it too often. The first 90% of the project takes 90% of the time, and the last 10% of the project takes 90% of the time again. How can 90% of our projects be 90% complete 90% of the time? This happens because we use planning and tracking methods that allow far too much subjectivity.

Earned Value is a project planning and tracking method that removes much of the subjectivity from the process.

- In the first article, we discussed the principles behind Earned Value planning and tracking, and learned how to create an Earned Value plan for a software project.
- In this second article (of two), we will learn how to track status against that plan, and how to handle problems like accounting for unplanned tasks.

## Tracking Against an Earned Value Plan

Tracking your progress should be a continuous activity. That is, as you are doing your work, you should also be collecting your data about your progress. Waiting until once per day or once per week to do this has two main disadvantages:

- Your data will be less accurate. This will be especially true of your record of effort you spent. With less accurate data, your ability to learn from your estimating mistakes is severely constrained, and the value of the data you collect is compromised.
- The effort to record your data seems to be more onerous. If you have an entire week's worth of data to record all at once, you will spend a lot of time reconstructing the week in your mind, recording all of the numbers, and measuring the work products.

So, you should try to record your data as you finish each task. Noting the effort you spent will be easy and accurate, and measuring the size of a single thing will take only a minute or two. If you forget to record something, then record the data as soon as you think of it.

(To facilitate understanding the following discussion, please refer to the example in Appendix A.)

### *Data to record*

These are the data that you will need to record:

- For each task you work on, record the effort you have spent on that task to date. If your work on a task is spread over days or weeks, continue accumulating the effort for that task until you have finished working on it.
- For each task you complete, record:
    - The week in which you completed it, and
    - The actual size of the product.
- For each week, record the total hours of productive time on task.

### *Status against Earned Value Plan*

The status against your earned value plan can most easily be seen on a graph that shows your Earned Value plan and your Actual value earned by week. (Refer to the Planned vs. Earned Value chart in Appendix A.) This chart shows several things:

- The yellow line is your original Earned Value plan.
- The red line is the actual value earned by week. It started out slightly ahead of plan, but has since fallen significantly behind plan. Although significant value was earned in the most recent week, the actual value is still far beind plan.
- The green line is a simple projection that assumes we will continue earning value at the same average rate we have been earning it so far. It shows us reaching full value for the project three weeks late.

This chart clearly shows us that our project is in jeopardy of finishing several weeks late. However, it can not tell us why this is so, nor can it give us insight into what we can do about it. To gain this sort of insight, we will need to look at some other data that we have collected.

### *Status against the Effort Plan*

Another important element of our status can be seen on a graph that shows your available effort plan and your actual effort spent by week. (Refer to the Plan vs. Actual Hours chart in Appendix A.) This chart also shows us several things:

- The red line shows our effort plan.
- The black line shows our actual effort spent.

This chart shows that we are falling slightly behind our effort plan each week. This could be an important piece of information about our status, but it, too must be analyzed in the light of the other data we have collected.

### *Understanding our Status*

Using the two charts we discussed above, we can identify nine distinct situations we could be in, as shown in the matrix below. We will discuss the important ones.

| | | **Earned Value** | | |
|---|---|---|---|---|
| | | Ahead of Plan | On Plan | Behind Plan |
| **Effort Available** | Ahead of Plan | 1 | 2 | 3 |
| | On Plan | 4 | 5 | 6 |
| | Behind Plan | 7 | 8 | 9 |

In the six cases where Earned Value is on or ahead of plan, we may be tempted to celebrate and not look any further at our data. But doing so could be a mistake. There could be reasons why your project appears to be in good shape now, but will fall apart later. For example:

- If your actual effort is ahead of plan, that can cause you to earn value faster than you anticipated. But, will you be able to keep up that pace? Is there some special situation that has allowed you to put in more hours? And if so, will that condition continue?
- If your actual effort is behind plan, you may be skipping or short-changing important steps in your process. For example, if you skipped reviews, or did only cursory reviews, then your testing time may balloon far beyond what you planned, because you will be

finding and removing many defects that you would have more economically removed in the reviews.

In the more likely case that your earned value is behind plan, you should have enough data to figure out why and what to do about it. For example:

- If your actual effort is ahead of plan, you are in the precarious position of working harder than planned, but not making the progress you expected. This indicates that the job is much harder than you expected. You should compare the actual object sizes to the planned sizes. You are likely to be building a bigger product than you expected. If not, then you simply underestimated the effort required for some or all of the tasks.
- If your actual effort is behind plan, then that may be the cause of the earned value problem.

In the example in Appendix A, both the earned value and the actual hours are behind plan. But in comparing the charts, we can see that actual hours are only slightly behind plan, while earned value is much worse. So, although fewer productive hours than we planned are probably contributing to our problems, other factors are likely to be at work as well.

Upon examining our task list we can see our problem. Although our effort estimates for the design work on Module A were good (actually, slightly high), our effort estimates for the coding tasks all appear to have been low. This could be a simple estimation error, but it could also be an indication that we spent too little time in design, and were still working things out while coding. How can we tell? The number and types of problems found in the code review should give us a good indication. If we found more design issues that we would expect, then deign problems may be the culprit.

Another troubling issue is the serious underestimation of the time to write the test cases. The number of cases came out very close to plan, so that wasn't the cause. You may want to look at the detail in the test cases to be sure that you are not over-doing it. It would also be a good idea to review the test cases to see if there is a problem with the way they were written.

Regardless of what is the cause of the problems, we must pay attention to the fact that as soon as we complete Module A, we will be developing Module B. Any error we had in our estimate for Module A may have been repeated in Module B. Re-visiting our estimates for Module B may be a good way to mitigate our schedule risk.

As you can see, understanding your status involves a lot of detective work. But because of the detail in our plan and the data we are collecting, we have plenty of evidence available to us. This is far superior to our traditional position of realizing we are in trouble, but having little understanding of why, and even less of an idea about what we can do about it.

## *Accounting for unplanned tasks*

As we said in the "Earning Value" section in the first article, when you do a task that you did not plan, you earn no value against your plan. This seems to be unfair, so we will take some time here to explore our options in this situation.

Planning errors are inevitable. When we make a plan, we include everything we can think of; but we know that reality will be different from our plan. Sometimes we simply estimate the wrong amount of effort for a task. But at other times, we include tasks we ultimately need not do, or we omit tasks that turn out to be required to complete the job.

In accounting for unplanned tasks (or for tasks we need not do), we must keep in mind the purpose of tracking our progress against our plan. Our plan represents the details behind the commitments we have made. We said that we would do a certain job in a specific timeframe, and our plan is our roadmap to fulfilling that commitment. Therefore, our plan is useful for two purposes: 1) guiding our work, and 2) understanding our status.

Our natural reaction when we realize that we have omitted a task (or that we need not do a task) is to change our plan to reflect our new understanding of the job. We see that our original plan was flawed, so we want to fix the plan. The problem with just fixing the plan is that every "fix" affects the end-point of the plan. That is, adding or deleting tasks changes our schedule and costs. The net result is that our plan no longer corresponds with our commitment; and it is no longer useful for guiding us in fulfilling our commitment.

On the other hand, by *not* changing our plan to reflect reality, we make it impossible to accurately account for the work we have done. We have the choice of logging our time and effort against a task other than what we are working on, or we skip logging it all together. Either way, the data we collect will be much less useful for guiding our future plan building, and we will make little progress in improving out ability to plan.

There is only one way to handle the situation so that we lose the accuracy of neither our plan nor our actual data. That is to add any unplanned tasks to our plan, but with zero planned effort (and hence, zero planned value). Doing this will have no effect at all on the effort, cost or schedule of our plan. But having these tasks in our plan allows us to record actual data against them so we have an accurate picture of what really went on during the project.

Planned tasks that need not be done are handled in a similar way. The plan for those tasks is left as it was originally made, and the actual data is entered against them. What actual data do we enter? Actual effort against the task is logged as zero, and the completion date is logged as the date when we realized that the task need not be done. So, we earn the planned value of the task at the point when we realize that it need not be done.

By accounting for our planning errors in this way, we can still see our status against our (flawed) plan and understand where we are relative to the commitments we made. As long as we can still use the plan for its two purposes (guiding our work and understanding our status), there is no need to go beyond this method in dealing with planning errors.

### *When to Re-plan*

People tend to go to one extreme or the other when it comes to re-planning. Either they make a plan and never change it, or they change the plan each time they find a flaw. Doing either of these things renders the plan unsuitable for the two purposes described in the previous section. It seems obvious that a plan that is allowed to become obsolete is useless. But why is this true of continuously changing the plan?

A plan that is continuously changing provides no perspective for assessing how you are likely to perform against the future tasks. Your performance to date on the project is difficult to understand, because the plan you see is different from the plan at the time you did the tasks. On top of this, as we said above, by changing the plan, you are likely changing the end-point so that it no longer corresponds to your commitments for the project. For these reasons, you should avoid re-planning.

The only time re-planning is called for is when your current plan is no longer useful for guiding your work and understanding your status. This will happen if the scope or nature of the project changes in mid-stream, or if your initial plan turns out to be seriously flawed. In either case, you will need to re-plan *and* renegotiate your project commitments. When you have completed re-planning and renegotiating your commitments, you will once again have a plan that is useful for guiding your work and understanding your status.

When you re-plan, begin your new plan at the current point in the project. Many people make the mistake of re-planning from the beginning of the original plan's timeframe, using actual data for the planned size and effort of tasks they have already completed. The problem with doing this is that it can make your projected value look better than it should because of the number of weeks when your "plan" is artificially the same as your actual data. This artificial situation could mask any estimating errors in your new plan until it is too late to recover from them. A plan should always look forward. Including history in the plan will reduce its usefulness.

### *Make a better plan next time*

Besides the two main uses of a plan (guiding your work and understanding your status), there is a third important use: to improve your planning accuracy. There are a number of ways in which you can use your plan at the end of your project (or when you must re-plan) to improve your next plan.

First, compare your estimates of size and effort for your tasks to the actual data. Is there a consistent bias in these estimates? Are there any specific estimating errors that are note-worthy? Are you good at estimating certain kinds of things, but not others? In short, learn your strengths and weaknesses; capitalize on your strengths, and look for ways to improve on your weaknesses.

Next, make note of your unplanned tasks. (These will be obvious because you will have entered them with planned effort of zero.) Look at the types of tasks you omitted from the plan and take specific actions to avoid forgetting them in the future. You may find it useful to create a checklist of all of the types of tasks you must remember to plan for, and use that checklist to avoid such errors in the future.

By the same token, make note of the tasks you planned but did not have to do. (These will be obvious because you will have entered zero for the actual effort.) Again, look at the types of tasks that these are, and try to understand why you made those errors. You may decide that an entry in your checklist could help you to avoid this sort of error as well.

Then, look at your planned vs. actual time on task per week. Are there any consistent errors in these estimates? Or were there special situations you overlooked in estimating certain weeks? Consider why you made these errors and what you may be able to do to avoid repeating them.

Finally, consider building a database of your actual size and effort data to use in estimating future projects. After collecting this data on even only one project, you will have an invaluable resource for making future plans. Now, instead of guessing at the size of what you will build, you can find similar items in your database and see the range of sizes they ended up being. Also, instead of guessing about the effort to do a task, you can draw explicit correlation between size and effort, and be able to make an informed effort estimate based on your size estimate.

So, we find that the data we collect while doing the project can help us even more on future projects. It can help us to improve our planning accuracy and avoid planning errors.

## *Conclusion*

Earned Value planning and tracking is a valuable method for improving our ability to accurately plan and track our projects.

- By providing a set of specific guidelines and set methods, it allows even the novice user to produce more accurate plans than before.
- By eliminateing subjectivity, it provides a reliable method for understanding project status.
- By maintaining a record of both your plan and your actual performance, it allows you to understand your planning errors and improve your ability to plan future projects.
- By recording actual data, it provides a basis for making more accurate estimates the next time.

Many individuals and organizations have adopted Earned Value and realized significant benefits. (The Personal Software Process teaches programmers how to use Earned Value to plan and track their personal work. And the Team Software Process makes effective use of Earned Value to help teams to achieved order-of-magnitude improvements in their project planning accuracy.) If your project plans are not as useful as you would like, then Earned Value should be of interest to you.

## Appendix A – Example Earned Value Plan

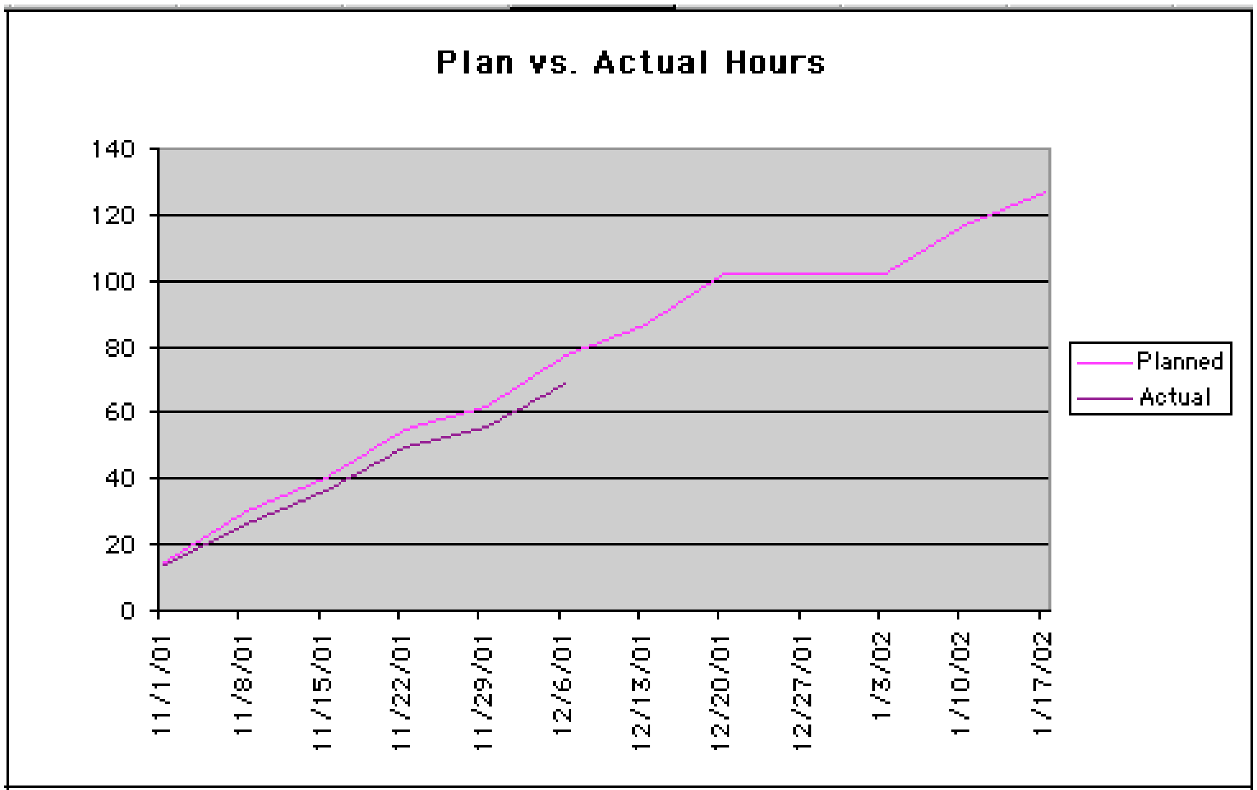
### *Task List*

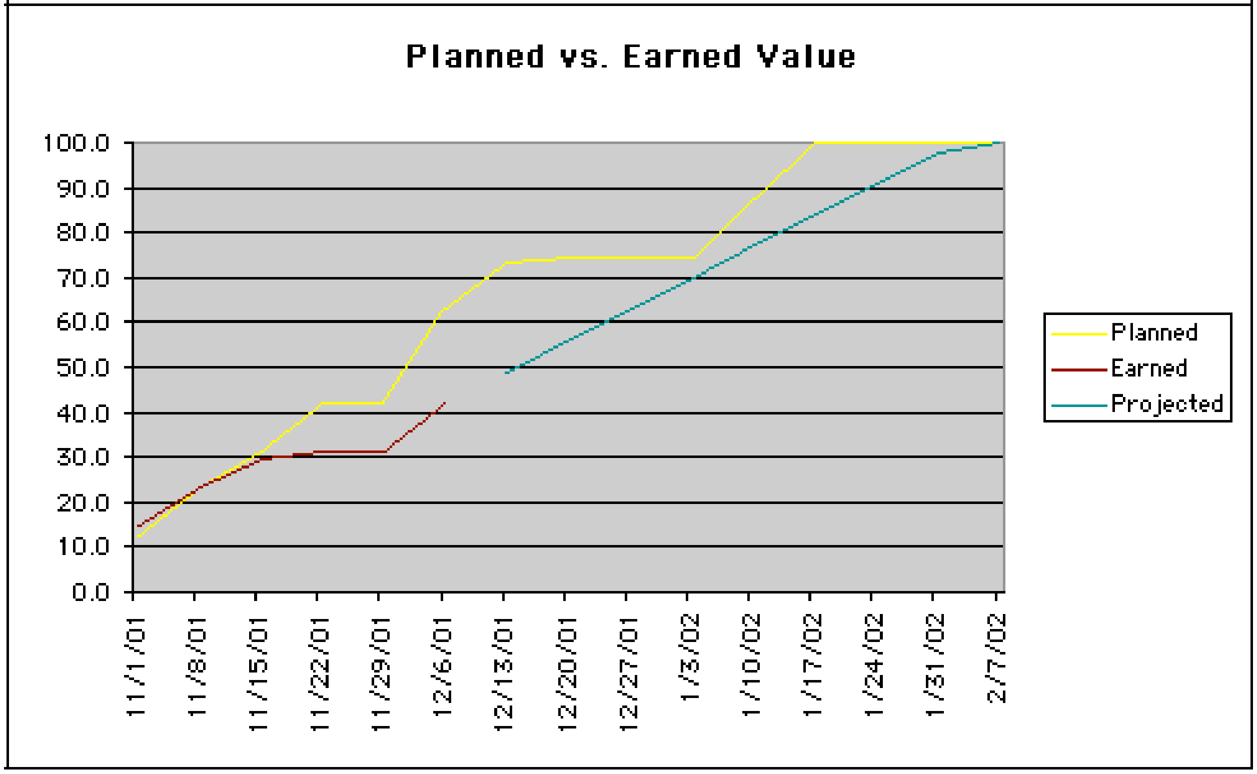
| | | Estimated | | | | | Actual | | |
|---|---|---|---|---|---|---|---|---|---|
| **WBS#** | **Task Description** | **Size** | **Effort Hrs** | **Cum Effort** | **Plan Value** | **Week** | **Effort** | **Size** | **Week** |
| **1** | **Module A** | **1000 LOC** | | | | | | **874** | |
| **1.1** | **Detailed Design** | | | | | | | | |
| 1.1.1 | Write Detialed Design | | 10 | 10 | 8.4 | 1-Nov | 8.25 | | 1-Nov |
| 1.1.2 | Detailed Design Review | | 5 | 15 | 4.2 | 1-Nov | 4.75 | | 1-Nov |
| 1.1.3 | Finalize Detailed Design | | 2.5 | 17.5 | 2.1 | 8-Nov | 1 | | 1-Nov |
| **1.2** | **Code** | | | | | | | | |
| 1.2.1 | Write Code | | 10 | 27.5 | 8.4 | 8-Nov | 12 | | 8-Nov |
| 1.2.2 | Code Review | | 5 | 32.5 | 4.2 | 15-Nov | 5.25 | | 15-Nov |
| 1.2.3 | Finalize Code | | 2.5 | 35 | 2.1 | 15-Nov | 3.5 | | 15-Nov |
| 1.2.4 | Compile | | 2.5 | 37.5 | 2.1 | 15-Nov | 3.5 | | 22-Nov |
| **1.3** | **Unit Test** | **50 cases** | | | | | | **52 Cases** | |
| 1.3.1 | Write Test Cases | | 12.5 | 50 | 10.5 | 22-Nov | 20.5 | | 6-Dec |
| 1.3.2 | Run Test Cases | | 12.5 | 62.5 | 10.5 | 6-Dec | 10 | | |
| **2** | **Module B** | **700 LOC** | | | | | | | |
| **2.1** | **Detailed Design** | | | | | | | | |
| 2.1.1 | Write Detialed Design | | 7 | 69.5 | 5.9 | 6-Dec | | | |
| 2.1.2 | Detailed Design Review | | 3.5 | 73 | 2.9 | 6-Dec | | | |
| 2.1.3 | Finalize Detailed Design | | 1.75 | 74.75 | 1.5 | 6-Dec | | | |
| **2.2** | **Code** | | | | | | | | |
| 2.2.1 | Write Code | | 7 | 81.75 | 5.9 | 13-Dec | | | |
| 2.2.2 | Code Review | | 3.5 | 85.25 | 2.9 | 13-Dec | | | |
| 2.2.3 | Finalize Code | | 1.75 | 87 | 1.5 | 13-Dec | | | |
| 2.2.4 | Compile | | 1.75 | 88.75 | 1.5 | 20-Dec | | | |
| **2.3** | **Unit Test** | **60 cases** | | | | | | | |
| 2.3.1 | Write Test Cases | | 15 | 103.75 | 12.6 | 10-Jan | | | |
| 2.3.2 | Run Test Cases | | 15 | 118.75 | 12.6 | 17-Jan | | | |

### *Available Effort*

| | Expected | | | Actual | | | Projected | Value |
|---|---|---|---|---|---|---|---|---|
| **Week of** | **Avail Hrs** | **Cum Hours** | **Planned Value** | **Effort Hrs** | **Cum Hours** | **Earned Value** | **Value** | **Per Wk** |
| 1-Nov | 15 | 15 | 12.6 | 14 | 14 | 14.7 | | 7.0 |
| 8-Nov | 15 | 30 | 23.1 | 12.5 | 26.5 | 23.1 | | |
| 15-Nov | 10 | 40 | 31.5 | 10 | 36.5 | 29.4 | | |
| 22-Nov | 15 | 55 | 42.0 | 13 | 49.5 | 31.5 | | |
| 29-Nov | 7 | 62 | 42.0 | 6 | 55.5 | 31.5 | | |
| 6-Dec | 15 | 77 | 62.9 | 13 | 68.5 | 42.0 | | |
| 13-Dec | 10 | 87 | 73.2 | | | | 49.0 | |
| 20-Dec | 15 | 102 | 74.7 | | | | 56.0 | |
| 27-Dec | 0 | 102 | 74.7 | | | | 63.0 | |
| 3-Jan | 0 | 102 | 74.7 | | | | 70.0 | |
| 10-Jan | 15 | 117 | 87.3 | | | | 77.0 | |
| 17-Jan | 10 | 127 | 100.0 | | | | 84.0 | |
| 24-Jan | | | 100.0 | | | | 91.0 | |
| 31-Jan | | | 100.0 | | | | 98.0 | |
| 7-Feb | | | 100.0 | | | | 100.0 | |

***Status Charts***


Plan vs. Actual Hours
11/1/01
11/8/01
11/15/01
11/22/01
11/29/01
12/6/01
12/13/01
12/20/01
12/27/01
1/3/02
1/10/02
1/17/02
Planned
Actual


Planned vs. Earned Value
11/1/01
11/8/01
11/15/01
11/22/01
11/29/01
12/6/01
12/13/01
12/20/01
12/27/01
1/3/02
1/10/02
1/17/02
1/24/02
1/31/02
2/7/02
Planned
Earned
Projected