CHAPTER FOURTEEN

Tracking Software Progress

Elizabeth (Betsy) Clark

How can we avoid the "90 percent done" syndrome in software development? Whether through wishful thinking, general optimism, or a desire to avoid confronting difficult situations, many projects do not communicate schedule overruns until late in the project when corrections are much more difficult and the consequences much more severe. In my consulting experience over the past twenty years, I have found that the most common motivation for implementing a measurement program is to track progress. My clients have typically fallen into one of the following categories:

- An acquisition or outsourcing organization wanting increased visibility into vendor progress following a bad experience with a vendor who never got the job done or was very late.

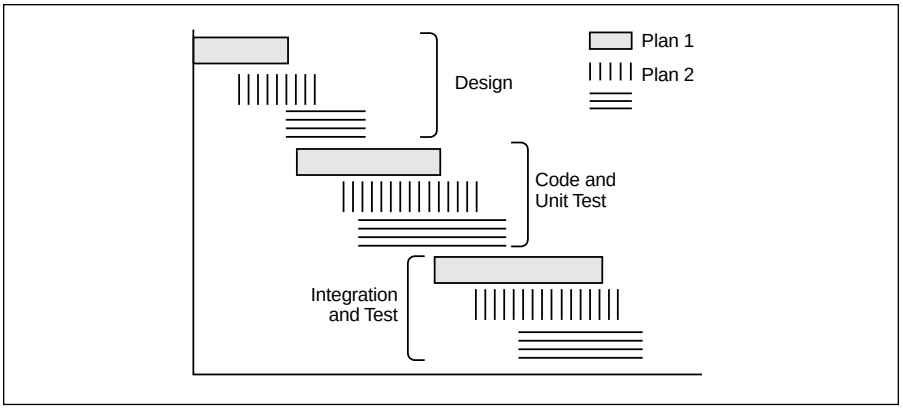

Figure 14-1: Typical Gantt chart reflecting overrun schedules and repeated replanning

- A contracting organization experiencing problems in meeting schedules. For them, this situation is no picnic either. By the time they have missed repeated deadlines, their credibility is lacking and project members are burned out.
- Executive management wanting a quick, easy-to-understand view of status for a large number of projects.

A typical project in trouble shows an activity or Gantt chart that looks something like Figure 14-1.

This pattern is sometimes referred to as a "slinky chart" because the end date just keeps inching to the right. The amount of time scheduled for integration and test typically gets shorter with each new plan. For projects delivering in multiple increments, we may find that functionality moves to later increments, producing what has been called a "bow wave" effect.

Gantt charts provide an easy-to-understand visual representation of the temporal sequencing of activities and certainly have their proper place as a management tool. But they do not give us information about how much of a given activity or product is completed at any specific time. For example, Figure 14-2 shows a Gantt chart for a hypothetical project on March 15. That date marks the halfway point in elapsed time for coding, but that is all the information we have. We know nothing about how much coding was actually done by March 15.

How can we effectively track progress? This chapter will describe the characteristics of an effective progress measure. It will also give examples of real-life progress measures that have been used with varying degrees of success, and the pros and cons of each. In each consulting engagement, my objective is to build on

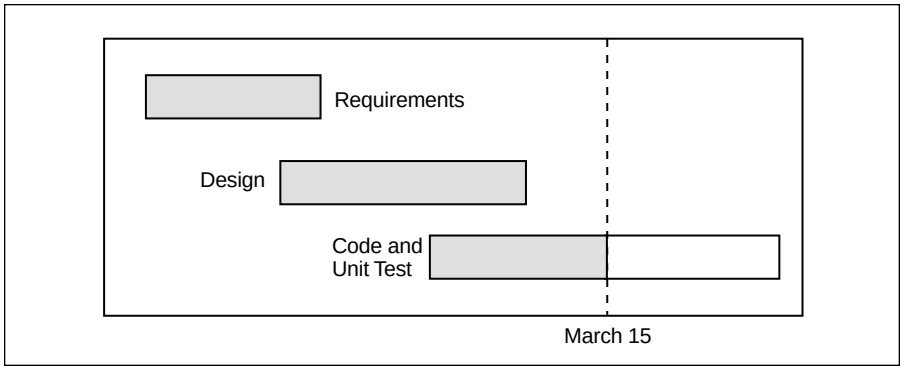

Figure 14-2: Gantt chart for a hypothetical project

what the client currently has available, making those resources more useful. One of the key messages I try to convey is that effective measurement does not, by itself, fix anything, solve anything, or do anything. Measurement often raises more questions than it answers, but these are better questions than people were asking before. And that is a big step forward. The effective use of any progress measure requires an honest desire to know the real status of the project and a willingness to take action to correct problems.

Tracking Progress: Criteria for Effective Measures

An effective measure for tracking progress should exhibit the following characteristics:

- **Objectivity:** The measurements should be based on criteria that are observable and verifiable. Measurements based on objective criteria are difficult to manipulate to make things look better than they are.
- **Near real time:** The measurements should reflect what is going on in the project now, not what happened a month ago. We want to be managing the present, not the past.
- **Multiple levels:** Multiple levels of data enable a manager to drill down and isolate problem areas or to roll up for high-level views. When a project is slipping its schedule, most managers want to know if the entire project is late or specific trouble spots need attention. Along with the drill-down capability, the measure can be rolled up to summary levels for upper management (oversight) review. Thus, an effective progress measure provides a vehicle for communicating project status to people outside the project.

- **Prediction:** The measure must support projections about future progress. Simply knowing that a project is behind schedule is not enough. We also need to predict when it will be completed. In general, past performance provides the best predictor of the future.

Two types of measures are commonly used for tracking progress.

1. **Activities Complete:** This measure compares actual progress against planned in terms of the number or percentage of completed activities.
2. **Work Units Complete:** This measure compares actual progress against planned in terms of the number or percentage of completed product units.

Each type of measure has strengths and weaknesses that are discussed in the following sections. Each type can be implemented poorly or effectively. Examples of each have been taken from real projects.

Fundamental to each type of measure is the comparison of planned to actual progress. The only way to know whether your project is ahead of or behind schedule is to look at it relative to an expectation. That expectation is your plan.

Activity-Based Measures of Progress

We can measure progress as a percentage of project activities that have been completed. We begin with an example of an easy-to-implement progress measure—one requiring only planned start and end dates for each major activity, along with periodic estimates of the percent of each activity complete.

Table 14-1 shows a monthly progress report for one software project. The project has defined a set of activities and assigned start and end dates. Each month, a percent complete estimate is provided, based on the project manager's assessment of how much was actually accomplished up to that point. This report is used to convey progress and current status to several levels of management, up to the corporate CIO. One problem with the report illustrated in Table 14-1 is that the CIO receives this report from more than a hundred individual projects every month. Not only are these reports not really helping the individual projects, but the CIO complains about not having time to read all of them and understand the status of each project.

Looking at Table 14-1, which is a report for one project on April 30, 2001, can we say that the project is ahead of or behind schedule? We might say it is behind because the Develop Design activity should be completed by now, but the

Table 14-1: Percent Complete Report for the End of April

Phase	Start Date	Planned End Date	Percent Complete
Document Business Requirements	1/15/2001	3/15/2001	100
Document Technical Requirements	1/15/2001	3/15/2001	100
Develop Design	3/1/2001	4/30/2001	66
Code and Unit Test	4/15/2001	6/30/2001	25
System Test	6/15/2001	8/15/2001	5
Training	8/01/2001	10/31/2001	0
Data Conversion	7/01/2001	9/30/2001	0
Installation	9/1/2001	11/30/2001	0

project manager is estimating that it is only two-thirds done. On the other hand, the project seems to be ahead on the Code and Unit Test and System Test activities. We could improve the information value of this report, while using the same information, by turning it into a graphical representation comparing planned versus actual percent complete. The CIO (or anyone else) could then tell at a

Table 14-2: Weights for Each Activity Derived from Scheduled Duration

Activity	Weight
Document Business Requirements	10%
Document Technical Requirements	10%
Develop Design	10%
Code and Unit Test	13%
System Test	10%
Training	15%
Data Conversion	15%
Installation	15%

glance how well the project is progressing. In addition, we would like to see the trend over time, not just for a single month.

Our first step is to represent the plan graphically. We did this by making a straightforward assumption that each activity contributes a percentage weighting to the total according to the length of time planned for that activity. For example, eight weeks have been planned for Document Business Requirements. If we add up the total number of weeks summed across all activities, we get 78, and 8 of 78 gives us 10 percent. Thus this activity counts as 10 percent of the total project. The weights for each activity are shown in Table 14-2.

To create the plan line, we can take a regular interval—in this case, monthly—and multiply the percent of each activity that should be completed at the end of each month. Thus, the Document Business Requirements activity should be 25 percent complete at the end of January (2 weeks divided by 8 weeks). It should be 75 percent complete at the end of February (6 weeks divided by 8 weeks), and so on. Based on the dates shown in Table 14-1, we can fill in Table 14-3.

The final step in creating the plan line is to multiply corresponding entries from Tables 14-2 and 14-3. For example, we derive the data point for Document Business Requirements for January by multiplying 10% by 25% to give us 2.5%. The completed weights are shown in Table 14-4.

Table 14-3: Planned Percent Complete for Each Activity by Month

	Jan 31	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov
Bus. Rqts.	25%	75%	100%	100%	100%	100%	100%	100%	100%	100%	100%
Tech. Rqts.	25%	75%	100%	100%	100%	100%	100%	100%	100%	100%	100%
Des.	0%	0%	50%	100%	100%	100%	100%	100%	100%	100%	100%
Code	0%	0%	0%	20%	60%	100%	100%	100%	100%	100%	100%
Sys. Test	0%	0%	0%	0%	0%	25%	75%	100%	100%	100%	100%
Train.	0%	0%	0%	0%	0%	0%	0%	33%	66%	100%	100%
Conv.	0%	0%	0%	0%	0%	0%	33%	67%	100%	100%	100%
Inst.	0%	0%	0%	0%	0%	0%	0%	0%	33%	67%	100%

Table 14-4: Weights for Each Activity Used in Creating a Plan Line

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov
Bus. Rqts.	2.5%	7.5%	10%	10%	10%	10%	10%	10%	10%	10%	10%
Tech. Rqts.	2.5%	7.5%	10%	10%	10%	10%	10%	10%	10%	10%	10%
Des.	0%	0%	5%	10%	10%	10%	10%	10%	10%	10%	10%
Code	0%	0%	0%	3%	8%	13%	13%	13%	13%	13%	13%
Sys. Test	0%	0%	0%	0%	0%	2.5%	7.5%	10%	10%	10%	10%
Train.	0%	0%	0%	0%	0%	0%	0%	5%	10%	15%	15%
Conv.	0%	0%	0%	0%	0%	0%	5%	10%	15%	15%	15%
Inst.	0%	0%	0%	0%	0%	0%	0%	0%	5%	10%	15%

Our next step is to gather up actuals from the past four months, not just the most recent month. We have gone back to the reports from each month to create the entries shown in Table 14-5.

By multiplying the activity weights from Table 14-2 by the actual percentage complete from Table 14-5, we can graphically present actual progress as shown in Figure 14-3.

Remember that the original complaint about this report came from the CIO: the textual report (Table 14-1) for each project was too confusing, and more than a hundred of these were produced for the entire IT organization.

One benefit of the graphical report shown in Figure 14-3 is that it shows planned and actual, so that in an instant, anyone can tell whether a project is ahead or behind. Because these values are expressed in percentages for each project, they can be rolled up for a division or for the entire IT organization. The CIO can then selectively drill down to the level of individual projects.

As an indicator of project status, this simple percent complete measure has several strengths, especially as compared to the original text report from Table 14-1. For example, the data provides visibility into project progress, and the project did not require a mature process to generate the plan or actual lines. Also, Figure 14-3 lets us make predictions about the future by extrapolating from the actual line.

Table 14-5: Percent Completed Reported by Month

Phase	Planned Start Date	Planned End Date	Percent Complete Jan	Feb	Mar	Apr
Document Business Requirements	1/15/2001	3/15/2001	10	25	50	100
Document Technical Requirements	1/15/2001	3/15/2001	15	33	75	100
Develop Design	3/1/2001	4/30/2001	5	10	20	66
Code and Unit Test	4/15/2001	6/30/2001	0	0	0	25
System Test	6/15/2001	8/15/2001	0	0	0	5
Training	8/01/2001	10/31/2001	0	0	0	0
Data Conversion	7/01/2001	9/30/2001	0	0	0	0
Installation	9/1/2001	11/30/2001	0	0	0	0

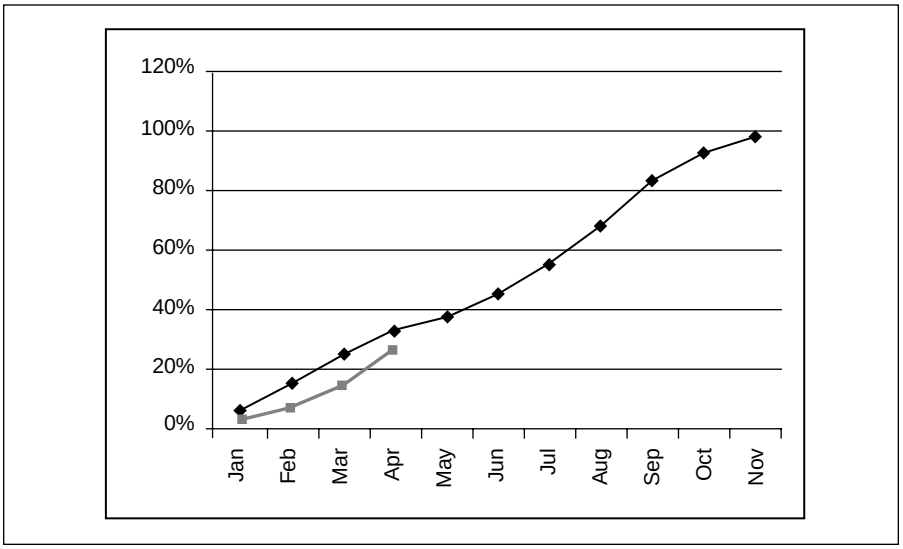

Figure 14-3: Graphical representation of planned activities complete

This percent complete measure has weaknesses in several areas:

- The measure is based on subjective judgment. Judgments tend to be overly optimistic, with the result that major schedule slips may not be apparent until late in the project.
- Reporting progress by major activity is too high a level for identifying problem areas. The next example shows a level of detail that supports an effective drill-down capability within a single project.
- Monthly reporting is too infrequent. Effective progress reporting occurs weekly.

One note of caution regarding progress measures based on subjective judgment: Be sure you do not punish someone for providing honest data. People should be in trouble only if they are not reporting things accurately. Conversely, there should be no shame from being behind, reporting honestly, and indicating what is needed to correct the problem. This requires a drastic change in culture for many organizations. Getting the numbers is not the hard part of progress measurement. Using the numbers effectively and constructively requires strong leadership.

Detailed Activity-Based Measure

Percent-complete measures can be based on a much lower level of granularity. Table 14-5 shows an example based on a very detailed breakdown of activities and rolled up to the project level. (Figure 14-4 is a simplification from an actual project but the general idea has been captured.)

For each of the units to be designed, an estimated number of design hours were generated (based on a combination of parametric cost-modeling and engineering judgment). For Subsystem 1, Unit A will require an estimated 120 hours, while Unit B will require an estimated 88 hours. Each unit is assigned a weighting based on these estimates. For example, the 120 hours estimated for Unit A represents 120/1071 hours or 11.20 percent of the estimated design effort.

Each design activity (for example, requirements trace or use cases) is assigned a weight based on a combination of historical data and engineering judgment. Thus, tracing the requirements counts for 15 percent, documenting use cases counts for 25 percent, and so on. We can multiply the two sets of weights to derive detailed weights for each unit by activity combination. For example, for the activity

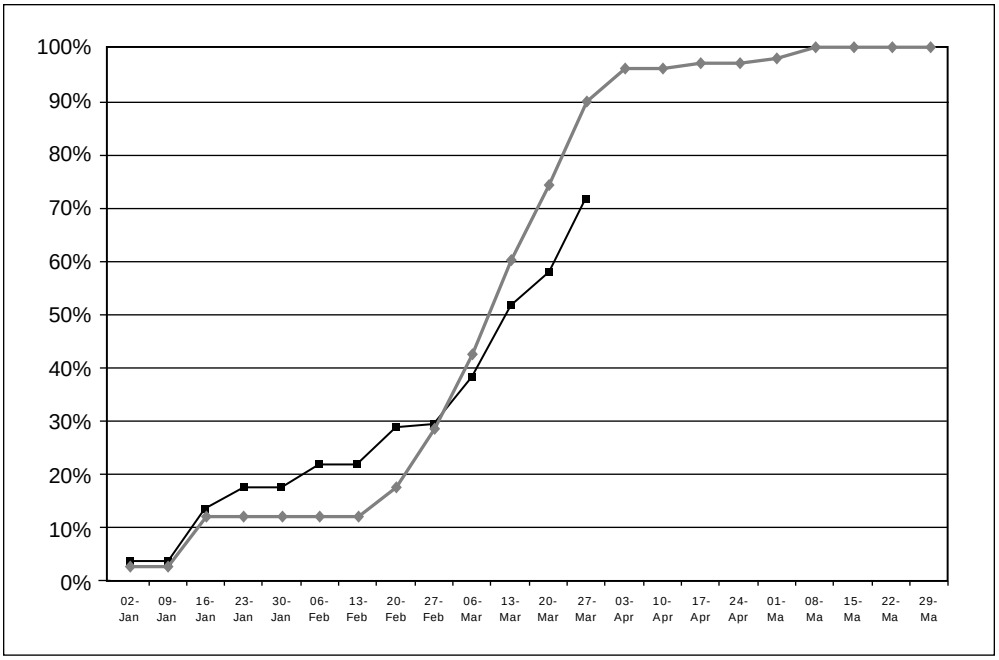

Figure 14-4: Graphical representation of percent complete based on data in Table 14-5

of requirements trace, we multiply 11.20 percent times 15 percent to arrive at a weighting of 1.68 percent for Unit A. This means that a credit of 1.68 can be claimed only after the requirement trace for Unit A from Subsystem 1 is completed, documented, and signed off (by SQA or whatever exit criteria are in place for calling a design activity complete). Each of the other design activities—use cases, object model, and peer review—is associated with planned and actual completion dates and weightings. For the sake of saving space, these dates and weightings are not filled in.

You can use this same method to track code progress or integration and test progress. The percentages can be summed on a weekly basis to show overall progress. By looking at the detailed data in the spreadsheets, it is possible to see the progress of individual tasks for individual units.

This is the approach underlying the concept of Earned Value, a topic that is beyond the scope of this chapter. An excellent reference is *Earned Value Project Management* by Quentin W. Fleming and Joel M. Hoppelman.

The strengths of this detailed level of tracking progress include the following:

- It is more objective than the less detailed example above because concrete exit criteria were associated with each of the design activities.
- Weekly reporting made this much closer to real time than the monthly reporting of the first example.
- The detailed reporting provided a clear drill-down capability for the project manager, who could look at progress from the perspective of the entire project or for individual subsystems and units.

The weakness in this example was the need for a detailed level of planning (certainly not a weakness in itself, but an immature process will not support this level of planning and tracking).

Product-Based Measures of Progress

The second type of measure—work unit progress—looks at progress from the perspective of the work product rather than the intermediate activities. Instead of counting activities completed, we count units of product that pass objective completion criteria. The key word here is "objective." Work unit progress can be measured at any phase in the development (units designed, units coded, units completing unit test, units integrated, tests successfully executed, problem reports closed). Units can refer to any work entity that is meaningful, including design components, function points, lines of code, screens, reports, and so on.

The basic concept is to start with an estimate of the total number of units, a planned start and end date, and a plan line that represents the number of units completed at various points. The completion of work units often follows an S-shaped curve similar to that shown in Figure 14-5. Progress appears slow at first, but the rate increases and then tapers off as the final and most difficult units are completed.

Figure 14-6 shows a measure of planned versus actual progress based on this type of measure. Again this example is from a real project.

The work unit progress measure shown in Figure 14-6 has the following strengths:

- It is objective to the extent that it is based on concrete exit criteria for counting a unit as coded.

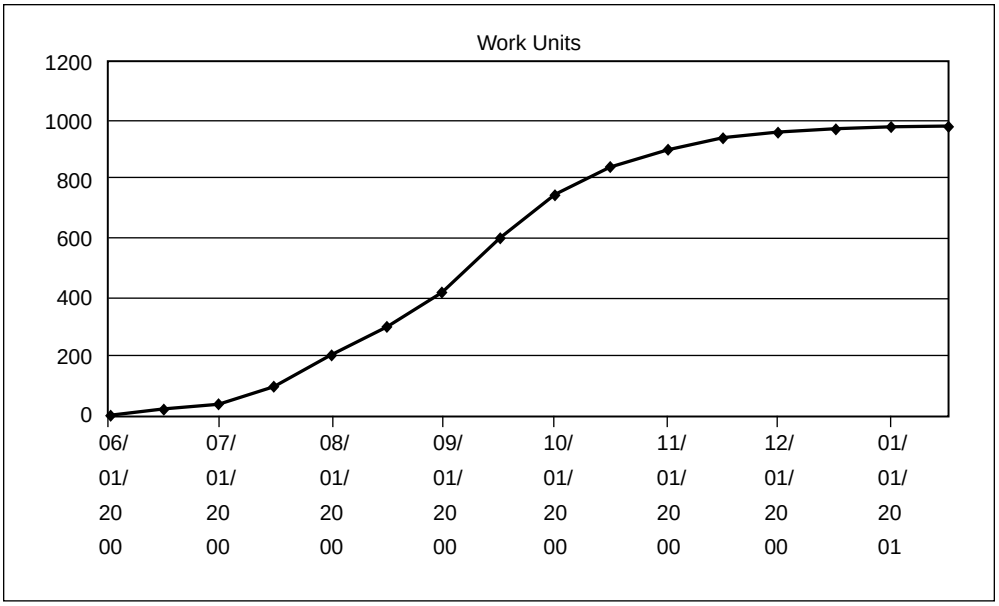

Figure 14-5: Typical S-shaped curve

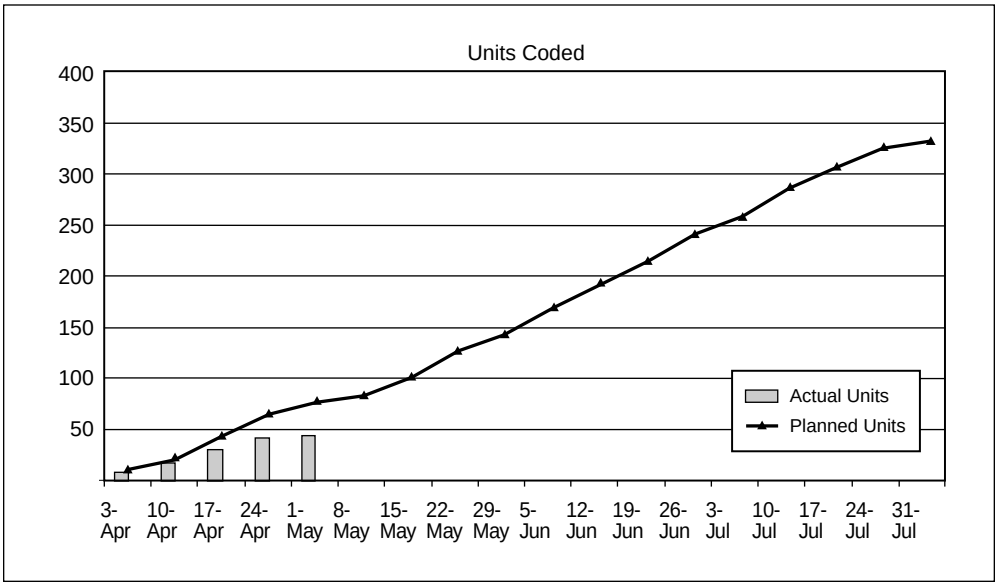

Figure 14-6: Work unit progress

- It can support roll-up and drill-down. We can look at coding progress for the entire project, or we can drill down and look at individual subsystems.
- The biggest strength: Work unit progress can be very useful, even in the absence of a detailed planning process. It provides visibility into the rate of progress required to meet the planned start and endates for any given activity. In that sense, it can be a useful counterpart to a Gantt chart by allowing us to answer the question "How many units do we need to have completed at any given time?"

Comparing Activity-Based and Product-Based Measures of Progress

The primary advantage of the percent-complete measure is that credit is given for work that is partially done. The primary disadvantage is that a project can claim credit for a lot of progress but have no completed units. Many units can be almost done before anything is totally finished. This could represent a red flag that people are taking credit for progress without completing the product. At the very least, it invites questions.

The advantage of the work-unit progress measure is that it shows the progress of the actual product as it evolves toward completion. The disadvantage is that no value is earned by the project until the whole unit is completed (no partial credit). Another disadvantage is that units tend to be weighted equally when there may be large differences in difficulty. If a project completes all the easy units first, the project may appear to be further along than it is in reality.

Recommendation

The best measure for any given project is the one that comes closest to meeting the criteria outlined at the beginning of this chapter: one that is most objective, most timely, most detailed (with a roll-up capability), and most predictive.

For either type of measure (percent complete or work unit), identifying the project's work units and/or activities and their completion criteria is important. In my consulting practice, this identification is typically done during a one-day workshop with the project members. When multiple contractors are involved, you may need to have separate workshops for each, especially if the development activities or work units differ (as, for example, with object-oriented versus functional decompositions). Follow-up workshops are held to work through any problems in implementing and reporting the measures.

No matter which measure you choose, be sure to keep the original baseline and then the latest plan. If comparisons are made against the latest plan only, everything looks on schedule. (I've seen projects do frequent replans so that actuals always match the "plan.")

The ideal case is to use both types of measures. Together, they should give a consistent view of progress. If they do not, that encourages questions that can provide important information about the true status of the project.

Biography

Elizabeth (Betsy) Clark has been involved in the practical application of measurement for predicting, controlling, and improving software process and product quality since 1979. She is the president of Software Metrics, Inc. a consulting company she co-founded in 1983. Dr. Clark is a primary contributor to *Practical Software Measurement: A Guide to Objective Program Insight.* She is a certified PSM instructor and has conducted numerous PSM training classes and workshops within the United States and Australia. Dr. Clark was also a principal contributor to the Software Engineering Institute's (SEI) core measures. She is an IFPUG Certified Function Point Specialist.

Through her affiliation with the Institute for Defense Analyses, Dr. Clark has extensive experience in performing independent cost analyses for government clients. Her experience covers a range of weapons platforms as well as large information systems. Dr. Clark is currently working with Dr. Barry Boehm and Chris Abts to develop and calibrate a cost-estimation model for COTS-intensive systems (COCOTS) under sponsorship of the Federal Aviation Administration.

Dr. Clark received her B.A. from Stanford University and her Ph.D. from the University of California, Berkeley. She is an avid equestrian, having earned her bronze, silver, and gold medals from the United States Dressage Federation. She can be reached at (703) 754-0115. Her e-mail address is Betsy@Software-Metrics.com.