Help for the project manager

Robert M. Poston,
Software Standards Editor

The other day I saw a project manager pick up the latest draft of IEEE P1058, a developing standard for software project management plans. The standard defines format and content for project plans. In about three minutes the manager laid the copy aside and told me he would create a plan on his own. Why did he reject the standard?

The words and organization of information in the plan described in the standard were not what he expected to see. Terms like “work package,” “work product,” and “project agreement” were new to him. The standard proposed an outline that emphasized the process of managing and the process of doing technical work as much or more than cost and schedule information. And besides, this was just one more boring standard. He had a real project to worry about.

This “shoot from the hip” approach made me think, “How can a busy project manager quickly make a sound decision to use or not to use this standard?” Here’s an organized way to make that decision.

Why a standard at all? Every software project is different. The product being developed can range in size from minuscule to enormous. The technology used may be old and familiar or brand-new and known only to a few. Project teams are sometimes small and members have a history of working together. On the other hand, professionals may come together for the first time on a project. With such diversity, why try to standardize management plans at all? Why make them all look alike?

The reason so many people have worked so hard to develop this standard is that project managers have expressed a great need to compare information from project to project. All cost- and schedule-estimation predictors, for example, require the same kind of information gathered from several projects. Evaluations of software development techniques and tools are best when based on project-to-project comparisons.

When information is organized in one standard format for all projects, comparisons are much easier to make.

Checklist for completeness. We see, then, that there is good reason for a project-management plan standard. But how about this particular standard? Will this IEEE standard help me?

Most project managers want certain specific questions addressed in project plans:

1. Who is responsible?

The standard addresses this question by telling us we must define organizational structure and interfaces as well as individual responsibilities.

2. What explicit, trackable activities and tasks must be performed?

The standard tells us all tracked activities and the tasks that make up those activities must be stated, and each one must be defined in a description called a work package.

3. What work products must be produced?

The standard covers this question in two ways: It requires a list of all items that will be delivered to the customer (project deliverables), and it requires all items produced in activities or tasks (work products) to be identified in workpackage descriptions.

4. When must work products be completed?

The standard does require schedule information, but it doesn’t tell us what scheduling technique to use. It gives project managers the freedom to choose an appropriate method for their projects.

I use an old management rule of thumb: If a project has more intertask dependencies than I can easily memorize (seven), I use PERT. Otherwise, I use a bar or Gantt chart.

We must realize, however, that when project managers can choose their own scheduling techniques, project-to-project information will become harder to compare.

5. What will it cost to build the work products?

STANDARDS BULLETIN

Error-prevention series scheduled

A new miniseries, “Preventing the Most Probable Errors,” will begin next issue in Software Standards. It will be presented in seven parts that will help you prevent the most probable errors in:

- planning a software project (July 1987),
- specifying software requirements (September 1987),
- designing software (November 1987),
- programming software (January 1988),
- testing software (March 1988),
- assuring software quality (May 1988), and
- managing software configurations (July 1988).

Error prevention is replacing early error detection in many segments of our industry. Standards help prevent errors. We can’t tackle all errors in software, of course. But, we’ll show you methods to prevent the ones that occur most often.

Do you have a software error that raises its ugly head again and again in your shop? How do you deal with it? I’d really like to hear from you.
The standard calls for traditional cost information to be defined at the work-package level. There's nothing new here. The project manager is simply asked to identify resources such as people at particular skill levels or computer systems of certain configurations and to state the amount of time those specific resources will be used.

Experienced project managers will recognize that the standard incorporates the old upward-summation calculation method: Calculate the costs of an activity by first adding all the costs of the tasks that make up that activity. Then find the project costs by summing activity costs.

6. How should a task be performed?
The standard requires that techniques to perform both management and technical tasks be identified in the project-management plan.

"How am I supposed to do this job?" is one of the first questions a person will ask when undertaking a new job. If the project-management plan spells out specific techniques, there is no question about how to do the job, and everyone pulls in the same direction.

7. According to what standard(s) should each work product be produced?
The project-management plan does call for identification of work-product standards. But, we encounter a pointer-to-pointer-to-pointer problem here.

An engineer working for a project manager will get an assignment to complete a certain task defined in a work-package description. The description tells the engineer to produce a work product called a gizmo. "What does a gizmo look like?" the engineer will ask.

To discover the answer, the engineer must seek out at least three references in three different places. He'll have to trail from the work-package description to the references section of the project-management plan to the standard for the work product.

In the days when technical information was recorded only on paper, this kind of searching had to be done. Today, the project manager can see that this information is available on-line in the engineer's computer system. The engineer should have to go no further than his computer screen to find out what a gizmo looks like.

Because software tools are evolving so rapidly, standards will continue to become outdated to some degree, especially in state-of-the-art environments.

8. How will individuals on the project know if they have produced a high-quality work product?
The standard says that metrics or measurements of the work products produced should be included in the project-management plan. Or, alternatively, metrics may be documented in the project's software quality-assurance plan.

Of course, only when people know how their work product will be measured can they judge the quality of their work product. So having metrics specified in the project-management or quality-assurance plan is extremely important.

However, there is a big problem with this project-management plan standard relative to metrics. It doesn't tell us what metrics to collect or what form to collect them in.

Suppose the gizmo project manager decided to collect failure-count metrics by life-cycle activity. Next year, the whiz-bang project manager wants to collect failure metrics by job function. And the what's-it project manager down the hall told the project team to measure complexity per software module.

All three project managers satisfied the metrics requirement in the IEEE standard. Yet, none of those metrics can measure anything from project to project. The gizmo project collected failure measurements partitioned by time (life-cycle phases). The whiz-bang project used the job functions (instead of time) for partitioning, and the what's-it project went off in an entirely different direction.

Managers must compensate for the standard in this area. They must add detail about standard measurements for work products and coordinate their metrics efforts company-wide. Otherwise, forget about project-to-project metrics comparisons.

9. What tools can replace human labor in each project activity?
The standard requires that tools used in technical activities be identified. But what about tools for project management, quality assurance, and configuration management?

Which PERT system should be used? Which standards checker should be used? Which change-management system should be used?

Again, project managers will want to augment the standard here with their own knowledge and experience.

10. Where can the project manager send team members to get answers to these questions?
The standard does not address this question. Nevertheless, it is a very important question to the project manager. Cost, schedule, and team morale can be greatly affected by untrained people.

Consider, for example, the designer who is baffled by a new software design tool. Will that designer take time on the job to learn to use the tool, or go to a class? Either way, this learning must be accounted for.

While it is fairly simple to identify a suitable tool or technique for a project, it is really difficult to get all members of the project team to apply the tool or technique efficiently. Many project managers will want to consider the effect of training, or not training, when writing their project-management plans.

What the checklist shows. Although not perfect, we can see that PI058 does cover a lot of the necessary bases in a project-management plan.

From the checklist, we can see where the standard is weak, so we can compensate in those areas. Particularly, we should be aware of deficiencies in the metrics area.

Perhaps a little reorganizing might help (see Figure 1). The standard is very flexible. It lets project managers reor-
ganize at will. It allows PERT and Gantt graphics to be included or referenced.

Other standards. As far as I can see, the project manager has four choices when it comes to project-management plan standards: Create one independently, use a company-created one, follow the military standard, or use this new IEEE standard.

Getting people to buy into a standard is difficult. A quietly conceived standard is usually rejected. In the IEEE we have learned that the more inputs we have and the more fights we endure, the better the standard and the higher the probability of it being accepted. An independent effort is only for the very strong of heart.

Using corporate standards is easier than writing your own. Lucky companies have had enough time and resources to develop some effective standards. Generally, though, standards are at the bottom of corporate priority lists and never get the attention they deserve in development or maintenance.

There is a military standard, MIL-STD 2167, for a software-development plan that may help military managers. Outside the military, I believe IEEE P1058 is the only consensus-based, industry-level project management plan standard available. I recommend that my commercial customers use this new standard, at least as a starting point. It is flexible enough to allow in-house tailoring that should make it acceptable in most cases.

More information. If you'd like more information about P1058, here are three contacts on the working group:

- Richard Thayer (chairman), Lockheed STC, 2100 E. St. Elmo Rd., Bldg. 30E, Austin, TX 78744; (512) 448-9702.
- Richard Fairley (cochairman), Wang Institute, Tyng Rd., Tyngsboro 01879; (617) 649-9731.
- Gary Whitten (secretary), AT&T Information Systems, 11900 N. Pecos, Denver, CO 80234; (303) 538-4909.

P1058 has been sent out for formal balloting once and received a very positive return. In early March, Thayer reported that the few negative comments that had been returned were resolved. We should see an approved standard this summer.

Standards effort status report

Current efforts. The following software engineering projects are under way:
- P982.1, A Standard for Measures to Produce Reliable Software.
- P982.2, A Guide to the Use of Standard Measures to Produce Reliable Software.
- P1028, A Standard for Software Reviews and Audits.
- P1044, A Standard for Classification of Software Errors, Faults, and Failures.
- P1045, A Standard for Software Productivity Metrics.
- P1061, A Standard for Software Quality Metrics.
- P1062, A Recommended Practice for Software Verification.
- ANSI/IEEE Std 829-1983, A Standard for Software Test Documentation. (This established standard is being reopened for review and update.)

Contact John Horch, chairman of the Computer Society Software Engineering Standards Subcommittee, for further information about any of these activities. He can be reached at Teledyne Brown Engineering, 300 Sparkman Dr., Huntsville, AL 35807; (205) 532-1100.

New standards. The IEEE Standards Board approved three software engineering standards at its December meeting: A Guide to the Use of Ada as a PDL (990), A Standard for a Software Engineering Standards Taxonomy (1002), and A Standard for Software Verification and Validation Plans (1012).

The board approved A Recommended Practice for Software Design Descriptions (1016) at its March meeting.

Now awaiting final approval are A Guide to Software Configuration Management (1042) and A Standard for Software Project Management Plans (1042). Watch the Standards Bulletin for more information on these standards as it becomes available.

- Richard Thayer (chairman), Lockheed STC, 2100 E. St. Elmo Rd., Bldg. 30E, Austin, TX 78744; (512) 448-9702.
- Richard Fairley (cochairman), Wang Institute, Tyng Rd., Tyngsboro 01879; (617) 649-9731.
- Gary Whitten (secretary), AT&T Information Systems, 11900 N. Pecos, Denver, CO 80234; (303) 538-4909.

Posix progress

The P1003.2 shell and utilities working group on the Posix portable Unix system interface definition has agreed on a complete set of definitions that will be incorporated in the standard's working draft, reported group chairman Hal Jespersen.

It also tentatively approved the contents of the command environments from a list of 141 common commands. The commands were categorized as follows: included execution, 58; excluded execution, 18; undecided execution, 34; included development, 17; excluded development, 3; undecided development, 11.

AT&T, the US Army, and X/Open each proposed command groups to the P1003.2 working group. The P1003.2 group adopted X/Open's approach of dividing commands into mandatory and optional software development groups.

In response to pressure to accommodate international character sets, date formats, collating sequences, and the like, a proposal on regular expressions will be forwarded to the /usr/group technical committee on internationalization for its recommendations.

More work is needed to ensure that the Posix standard conform to the US National Computer Security Center's trusted-system security standards, reported Jim Isaak, chairman of the P1003.1 real-time working group.