31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 109

7

Software Planning

Up to this point, we have examined estimating, but the objective of the planning process is to produce a plan. This chapter describes how to use the results of the estimating process to produce a project plan. Starting with a discussion of plan requirements, this chapter covers project and period plans, making schedules, and tracking and reporting progress. The chapter also discusses planning accuracy, plan changes, and management reporting. Although the change-management and reporting issues are not very significant for the PSP exercises, they will be more important when you work on a TSP team. Because the PSP methods can be extremely helpful in handling these plan-management issues, they are covered in this chapter.

7.1 Plan Requirements

In producing a plan, the result must meet certain requirements. The five basic requirements for a plan are that it be accessible, clear, specific, precise, and accurate. These topics are discussed in the following paragraphs.

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 110

Is It Accessible?

Accessibility is best illustrated by the case of a company that was fighting a major U.S. government lawsuit. At one point, they were required to provide all of the historical documents that related to a particular topic. Because the company was large, this material was scattered among thousands of files. The cost of searching all those files would have been prohibitive, so the company gave the plaintiff all the files that could possibly contain any of the desired data. Even though these files undoubtedly contained material the company didn't want released, they judged this problem to be minor compared to the costs of a detailed document review. Although the plaintiff knew that what he wanted was somewhere in the many thousands of pages of files, he might as well not have had it. In practical terms, it was inaccessible.

To be accessible, a plan must provide the needed information so that you can find it; it must be in the proper format; and it must not be cluttered with extraneous material. Although having complete plans is important, voluminous plans are unwieldy. You need to know what is in the plan and where it is. You should be able to quickly find the original schedule and all subsequent revisions. The defect data should be clear, and the program size data must be available for every program version. To be most convenient, these data should be in a prescribed order and in a known, consistent, and nonredundant format.

Is It Clear?

I find it surprising that even experienced developers occasionally turn in PSP homework that is sloppy and incomplete. Some items are left blank, others are inconsistent, and a few are obviously incorrect. Sometimes, even incomprehensible notes are jotted in the margins. Such work is not data and should be rejected. The fundamental point this book teaches is the importance of quality data. If the data are not complete and unmistakably clear, they cannot be used with confidence. If they cannot be used with confidence, there is no point in gathering them at all.

Is It Specific?

A specific plan identifies what will be done, when, by whom, and at what costs. If these items are not clear, the plan is not specific. For the PSP plans you produce, the data in the Project Plan Summaries specifically answer these questions. You will finish the numbered program on the prescribed schedule and for the cost of your total estimated time.

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 111

Is It Precise?

Precision is a matter of relating the unit of measure to the total magnitude of the measurement. If, for example, you analyzed a project that took 14 programmer years, management would not be interested in units of minutes, hours, or probably even days. In fact, programmer weeks would probably be the finest level of detail they could usefully consider.

For a PSP job that takes five hours, however, units of days or weeks would be useless. Conversely, units of seconds would be far too detailed. Here you are probably interested in time measured in minutes.

To determine an appropriate level of precision, consider the error introduced by a difference of one in the smallest unit of measure. A project that is planned to take 14 programmer years would require 168 programmer months. An uncertainty of one month would contribute an error of at most 0.6%. In light of normal planning errors, this is small enough to be quite acceptable. For a five-hour project, an uncertainty of one minute would contribute a maximum error of about 0.33%. A unit of measure of one minute would thus introduce tolerable errors, whereas units of an hour or even a tenth of an hour would probably be too gross.

Is It Accurate?

Although the other four points are all important, accuracy is crucial. A principal concern of the planning process is producing plans with predictable accuracy. As you plan the PSP work, do not be too concerned about the errors in each small task plan as long as they appear to be random. That is, you want to have about as many overestimates as underestimates. These are called **unbiased estimates**. As you work on larger projects or participate on development teams, the small-scale errors will balance each other out and the combined total will be more accurate. The PROBE method guides you in producing unbiased, or balanced, plans.

7.2 Project and Period Plans

As developers, we all use both project plans and period plans. A **period plan** is for a calendar period—a day, week, month, or year. Period plans concern activities that are scheduled on a calendar basis. Businesses run on period plans. They pay salaries at regular intervals, periodically bill customers, and pay dividends every quarter. We also live in a periodic world. We eat and sleep at regular intervals and work on specified days of the week.

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 112

The project estimates and plans cover the effort and cost to develop products. The products may be tangibles like programs and books, or intangibles like designs and test plans. **Project plans** are driven by the needs of the job. Some tasks must precede others, and various deliverables can be produced only when the proper preparatory work has been done.

Although period plans and project plans are different, they must be related. Much of the difficulty that organizations have with software development stems from mismanagement of this period-project relationship. When making a plan, you should start with an estimate of work to be done. If the estimate is incomplete or inaccurate, the resulting plan will almost certainly be incomplete and inaccurate. However, even with an accurate and complete estimate, if you don't do a competent job of relating the project estimates to the calendar, the resulting plan could still be troublesome.

To make a project schedule, spread the estimated work for the job over the available time. This mapping of project tasks to the calendar becomes the project schedule. It is a key element of the project plan. Because your work is about products and your life is in periods, however, both period plans and project plans are important to you. You cannot produce a competent plan without properly producing and relating the two.

The relationship of the estimate and schedule is shown in Figure 7.1. This is the planning framework originally covered in Chapter 4 (see p. 63). It shows that

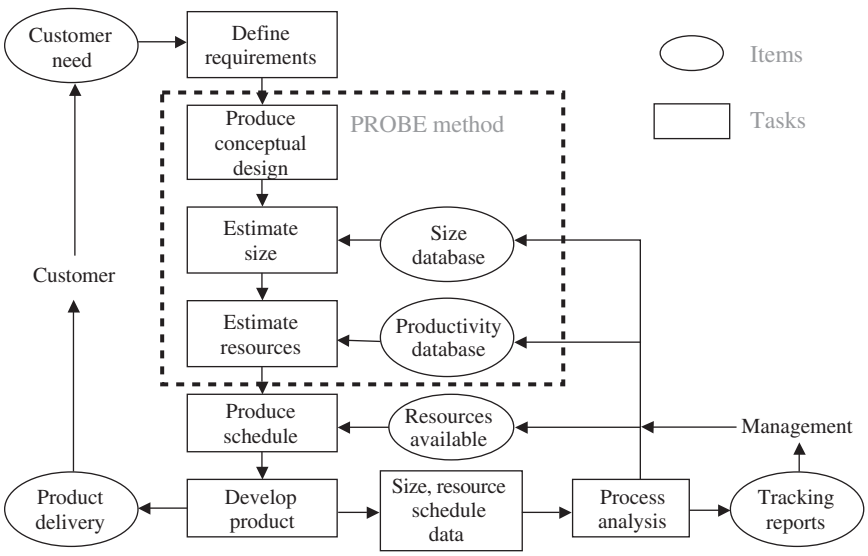

FIGURE 7.1 PROJECT PLANNING FRAMEWORK

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 113

the size and resource estimates are produced first and that these estimates are used to produce the schedule. However, the "Produce schedule" box has two inputs: the estimates and the available resources. The key question addressed now is, What resource information do you need and how do you combine it with the estimate to produce the schedule? That is the next subject in this chapter.

7.3 Producing the Schedule

To make a schedule, start by estimating the effort required to do the job. Then allocate these hours to the various project tasks. For the PSP exercises, spread the total hours over the phases using the To Date % from the projects completed to date. This assumes that the work for the new job will be distributed in much the same way it was for the prior jobs. It also assumes that each project phase requires only one task. For small projects like the PSP exercises, the projects are small enough that the phases can be considered as single tasks, and the work is similar enough from job to job that the To Date % provides reasonably accurate task estimates.

For larger team projects, there are normally many tasks per phase, and each project generally has a different To Date % time distribution. You must then estimate the time for each task. The PSP will help you do this if you have recorded the size and time for similar tasks. Then you can use the PROBE method for these tasks and use those estimates in your project planning. You will then be able to make accurate estimates, even for large projects. The key is to track time and size data for each kind of task and to use these data to make PROBE estimates whenever you have the data to do so. Even if you only have data on one task and even if these data are only for one team member, you can use the data until you have enough personal data to make a PROBE estimate.

Even if you have to use someone else's data, your estimate will be better than a guess. In one case, a large team was estimating the design work for the next project phase. They had completed the requirements work and concluded that there were about 600 classes to design in the next phase. The team was reluctant to guess at the time required to design a class because they would then multiply that guess by 600. One of the developers was ahead of schedule and had data on two classes she had already designed. The team used her data for the estimate. The result was within about 10% of the actual time the team spent.

Estimating Task Hours

To relate project estimates and period plans, you must decide how many hours you will spend each week working on project tasks. Then you can make the schedule.

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 114

On a TSP team, the hours spent on planned tasks are called **task hours**. Although you may work on many other project-related activities, if these activities are not specific tasks in your plan, that time is not task time. Task hours are part of the project plan, and normal working hours are part of the period plan. The rest of your working time is for those things you don't want to make the effort to plan, such as answering e-mail, attending meetings, helping coworkers, and so forth. To accurately plan this other work, you would have to measure and track all of these miscellaneous tasks. This is not usually worthwhile unless your work is billed by the hour and you need an accurate record.

Before estimating weekly task hours, consider the number of work hours in a year. A 52-week year of 40-hour weeks has 2,080 working hours. After accounting for vacations, holidays, sickness, and personal time, working hours are generally cut by 10% to 15%. In addition, in most engineering organizations, the developers spend 20 to 25 hours a week in meetings, handling e-mail, consulting on other projects, assisting marketing, or performing any of the other myriad tasks they generally must do. In a typical 40-hour workweek, it is not unusual for developers to put in only about 12 to 15 task hours. Many factors affect these hours, and they will vary substantially among individuals and even from week to week. Although most developers believe that they should aim for 20 or more task hours per week, few teams are able to achieve that many. Then, when their plans assume too many weekly task hours, their projects start out in trouble and they spend weeks struggling to recover. Until you have data on your own work, be conservative with your task-hour estimates.

Managing Task Time

The amount of time you spend on project tasks will fluctuate. First-time TSP team members generally start with a rate between 12 and 15 task hours per week. After a few weeks of work, they can often increase this task-hour rate to 15 to 17 hours, and some occasionally reach almost 20 task hours. Some experienced TSP teams can even reach 20 or more task hours in an average week, but they usually work more than 40 hours per week to do so. As with size and time estimating, the key is to gather data on your personal performance and to use these data in making estimates and plans.

With a realistic task-time plan, you can usually meet most of your schedule commitments. If your estimates are seriously off, when the inevitable schedule crisis comes, you can defer all routine work and devote 40 or more hours per week to the highest-priority tasks. For a brief period, you won't answer your mail, go to meetings, do paperwork, or even take coffee breaks. You will do nothing but work. If you plan for 15 to 20 weekly task hours, in a crunch you could easily double your task time. If you try to do this for any length of time, however, you will quickly burn out and the quality of your work will likely suffer.

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 115

In the long run, periodic crash efforts may help you to meet commitments, but they do not improve overall productivity. Although you will be very productive for short periods, when you get tired, your productivity will drop sharply. Just as in the fable of the tortoise and the hare, over the long term, if you make steady and orderly progress, you will generally outperform those who work in feverish spurts. Try to make realistic plans that leave a reasonable allowance for the normal events of your daily working life. This will provide a prudent cushion for the occasional last-minute crisis.

7.4 Making the Schedule

The PSP scheduling procedure is shown in Figure 7.2. This complex a procedure is not needed for the PSP assignments in this text, but you will use these methods with the TSP. Because most TSP tools will handle the scheduling and tracking calculations automatically, there is generally no need to do them by hand. To use these planning methods properly, however, you should understand how they work. The data in the following example are simplified versions of the data I gathered while writing the manuscript for the original PSP book (Humphrey 1995). The descriptions are keyed to the numbered items in Figure 7.2 and use the data shown in Tables 7.1 and 7.2.

1. **Estimate Resources:** Start with a resource estimate. For the PSP, this is the total estimated hours for the project.
2. **Estimate the Tasks:** For the PSP exercises, use the To Date % to allocate your time among the project phases. For larger projects, use data on prior similar tasks to estimate the time for the new tasks. Then enter the time for each task in the *Plan Hours* column on a Task Planning Template like that shown in Table 7.2.
3. **Estimate Weekly Task Hours:** Except for your first TSP project, use historical data to guide the task-hour estimate. To avoid getting overcommitted, base your estimates on personal data if you have it or on data from experienced TSP teams if you can get it.
4. **Calculate Available Hours:** If, like most developers, you spend at least some time on other projects, estimate the part of your task hours that you can commit to this project. Then enter these hours for each week, as shown in Table 7.1. As you do, consider both personal and other project commitments.
5. **Plan Task Dependencies:** Arrange the tasks in the rough order in which you expect to do them. This initial order should consider your current understanding of task dependencies. Don't worry too much about the precise task order because you will probably make many changes as the work progresses.

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 116

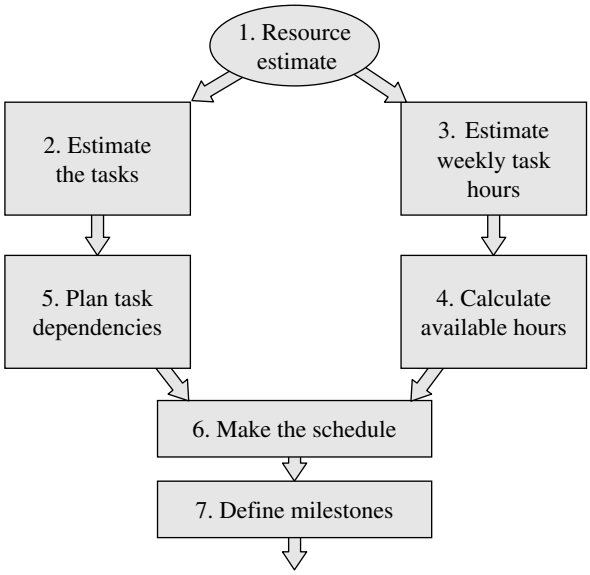

FIGURE 7.2 PSP SCHEDULE PLANNING

TABLE 7.1 SCHEDULE PLANNING TEMPLATE

Date	Week	Plan Hours	Cum. Hours	Actual Hours	Cum. Act. Hours	PV	Cum. PV	EV	Cum. EV
3/8	1	20	20			0.99	0.99		
3/15	2	20	40			0.53	1.52		
3/22	3	20	60			1.43	2.95		
3/29	4	10	70			0	2.95		
4/5	5	15	85			0	2.95		
4/12	6	15	100			2.94	5.89		
4/19	7	20	120			0	5.89		
4/26	8	15	135			0	5.89		
5/3	9	20	155			3.66	9.55		
5/10	10	15	170			0	9.55		
5/17	11	20	190			0	9.55		
5/24	12	20	210			2.94	12.49		
5/31	13	20	230			0	12.49		

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 117

TABLE 7.1 (continued)

Date	Week	Plan Hours	Cum. Hours	Actual Hours	Cum. Act. Hours	PV	Cum. PV	EV	Cum. EV
6/7	14	20	250			0	12.49		
6/14	15	20	270			4.37	16.86		
6/21	16	20	290			0	16.86		
6/28	17	20	310			0	16.86		
7/5	18	20	330			4.37	21.23		
7/12	19	20	350			0	21.23		
7/19	20	20	370			0	21.23		
7/26	21	20	390			3.66	24.89		
8/2	22	20	410			0	24.89		
8/9	23	20	430			0	24.89		
8/16	24	20	450			4.37	29.26		
8/23	25	20	470			0	29.26		
8/30	26	20	490			2.14	31.40		
9/6	27	20	510			0	31.40		
9/13	28	20	530			2.94	34.34		
9/20	29	20	550			0	34.34		
9/27	30	20	570			0	34.34		
10/4	31	20	590			4.37	38.71		
10/11	32	0	590			0	38.71		
10/18	33	0	590			0	38.71		
10/25	34	10	600			0	38.71		
11/1	35	20	620			0	38.71		
11/8	36	20	640			0	38.71		
11/15	37	20	660			4.37	43.08		
11/22	38	20	680			0	43.08		
11/29	39	20	700			0	43.08		
12/6	40	20	720			3.66	46.74		
12/13	41	20	740			0	46.74		
12/20	42	20	760			2.94	49.68		
12/27	43	0	760			0	49.68		
1/3	44	20	780			0	49.68		
1/10	45	20	800			0	49.68		
1/17	46	20	820			2.94	52.62		

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 118

TABLE 7.2 THE TASK PLANNING TEMPLATE

Phase	Task	Plan Hours	Cum. Plan Hours	PV	Cum. PV	Plan Week	Actual Hours	Actual Week	EV	Cum. EV
Plan	Book Plan	15.10	15.10	0.99	0.99	1				
Draft	Preface	8.14	23.24	0.53	1.52	2				
Draft	Chapter 1 Draft	21.70	44.94	1.43	2.95	3				
Draft	Chapter 2 Draft	44.75	89.69	2.94	5.89	6				
Draft	Chapter 3 Draft	55.60	145.29	3.66	9.55	9				
Draft	Chapter 4 Draft	44.75	190.04	2.94	12.49	12				
Draft	Chapter 5 Draft	66.45	256.49	4.37	16.86	15				
Draft	Chapter 6 Draft	66.45	322.94	4.37	21.23	18				
Draft	**Planning Drafts**	**0**	**322.94**	**0**	**21.23**	**18**				
Draft	Chapter 7 Draft	55.60	378.54	3.66	24.89	21				
Draft	Chapter 8 Draft	66.45	444.99	4.37	29.26	24				
Draft	Chapter 9 Draft	32.55	477.54	2.14	31.40	26				
Draft	**Quality Drafts**	**0**	**477.54**	**0**	**31.40**	**26**				
Draft	Chapter 10 Draft	44.75	522.29	2.94	34.34	28				
Draft	Chapter 11 Draft	66.45	588.74	4.37	38.71	31				
Draft	Chapter 12 Draft	66.45	655.19	4.37	43.08	37				
Draft	**Design Drafts**	**0**	**655.19**	**0**	**43.08**	**37**				
Draft	Chapter 13 Draft	55.60	710.79	3.66	46.74	40				
Draft	Chapter 14 Draft	44.75	755.54	2.94	49.68	42				
Draft	Chapter 15 Draft	44.75	800.29	2.94	52.62	46				
Draft	**All Drafts**	**0**	**800.29**	**0**	**52.62**	**46**				

For small jobs, ranking the tasks in order is relatively simple. However, for larger projects, it is often necessary to work together with your teammates to identify the task dependencies. If you find it helpful, you can also use a critical path scheduling system such as PERT (Zells 1990).

6. **Make the Schedule:** Starting with the first task, check the planned cumulative hours. In Table 7.2, the first task takes 15.10 hours. From Table 7.1, the plan shows 20 cumulative hours for week 1, so the first task will be finished in week 1. Enter "1" in the *Plan Week* column for that task. For the second task, the Preface, the cumulative hours are 23.24. The planned cumulative hours in

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 119

Table 7.1 first exceed this value in week 2, so the planned week for completing the second task is 2. Continue this process until you have entered the planned completion weeks for all the tasks.

7. **Define Milestones:** Finally, to facilitate project tracking and reporting, identify the key project milestones and list their planned completion dates. In Table 7.2, these are shown in bold as zero-hour tasks. Check the TSP support tool you will use to see what convention it uses to identify milestone tasks.

You now have a personal plan. For a team plan, you will also need to know the key task dependencies, task responsibilities, and principal checkpoints.

7.5 Earned Value

When planning a project with many tasks, you will need a way to track and report progress, particularly when you complete tasks in a different order than originally planned. If all the tasks take an equal amount of time or if there are only a few tasks, this might not be difficult, but projects typically have many tasks of differing types and sizes. The *Earned Value* (*EV*) measure provides a convenient way to address this tracking and reporting problem (Boehm 1981; Humphrey 1989). The EV method establishes a value for every task, regardless of its type. Whenever you complete a task, you earn this EV amount. To judge progress against the plan, merely compare the EV each week with the planned earned value, or PV, for that week. If the cumulative EV equals or exceeds cumulative PV, you are on or ahead of schedule.

To calculate PV, determine every task's estimated percentage of the total task hours for the entire project. That is that task's planned value. You earn the earned value when you complete the task. For example, for a project of 1,000 total task hours, a 15-hour task would have a planned value of 1.5. When you completed this task, you would earn 1.5 EV, which would then be added to the total project EV. Note that there is no partial credit for partially completed tasks. The earned value credit is given only when the task is completed. If the task is half done, it contributes nothing. For tasks that are so big that you want intermediate progress measures, break them into subtasks. Even when tasks take much more (or less) than the planned time, their EV is unchanged and is the planned value (PV) that was established when making the plan.

Earned value is particularly useful when there are frequent plan changes. It is also a surprisingly accurate way to measure project status and to estimate the project's completion date. In using EV, however, keep the following limitations in mind:

- The EV method assumes that the rate of task completion in the future will be roughly the same as what it was in the past. If this is not the case, EV-based projections will not be accurate.

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 120

- The EV method measures progress relative to the plan. If the plan is inaccurate, the EV projections will also likely be inaccurate. The EV method corrects planning errors when these errors are roughly the same for every project phase. That is, if you estimated project size as 10 KLOC and it turned out to be 25 KLOC, the EV projections could still be accurate. If, however, the development plan was accurate but the test plan was badly underestimated, the EV projections would be accurate for the development work but inaccurate for testing.
- The EV method assumes that the project's resources are uniform. If the staffing level increases, the EV projections will be pessimistic, and if the staff has recently been cut, the projections will be optimistic.

Generally, when plans are based on historical data, the EV method will provide accurate projections when the planned time distribution across the job phases is consistent with the historical To Date percentages. When the distribution by phase is unrealistic, the EV method will not provide accurate projections.

Depending on the project's size and expected duration, plan in sufficient detail to provide frequent status feedback. For example, for a 100-hour project stretching over five to six weeks, at least ten subtasks are required to provide one or two checkpoints a week. Although you will want to see regular EV progress, do not track yourself to death. My rule of thumb is that one checkpoint per week is not enough, and one per day is too many. I prefer two to four task completions a week. This requires that tasks be planned to take less than ten hours each.

7.6 An Earned Value Example

Tables 7.1 and 7.2 show simplified versions of the actual task and schedule plans I used for writing the draft of the first PSP book (Humphrey 1995). I had expected to spend 800.29 hours to write this draft and, based on my planned available time, estimated that I would finish in late January. As shown in Table 7.1, I planned a vacation in October (weeks 32 and 33), when I expected to accomplish nothing on the manuscript. I calculated that the draft would take 52.62% of the total book-writing time of 1,520.89 hours. Table 7.2 lists the planned completion dates for the draft chapters, together with the chapter PVs.

My early progress in writing the manuscript is shown in Tables 7.3 and 7.4. As is clear, I did not write the chapters in the planned order. I wrote the preface, then Chapter 11, and then Chapter 6. Next, I wrote Chapters 1, 2, and 3 in order. I then finished Appendix A before starting Chapter 4. Note several important messages here. First, even though this was my fourth book, my plan was not very accurate. I had originally planned to start the book with the material in Chapter 13. As writing progressed, however, I realized that this was not the right way to start

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 121

this book so I made a new design. Because I didn't add any new work, however, I could still use earned value to track progress against the original plan.

Later, I decided that the statistical material covered in many of the chapters should be treated more completely in a new Appendix A. Until I included this work in my plan, however, working on Appendix A did not earn EV credit. By now, getting EV credit had become so important to me that I kept delaying work on Appendix A until I changed the plan. Because I didn't have a TSP tool to do it for me, I had to manually recalculate the EV for all the tasks, as shown in Tables 7.3 and 7.4. The Appendix A tasks had the same planned work distribution by phase as the rest of the book, so the manuscript draft was still planned to take 52.62% of the job. Adding these tasks, however, did reduce the EV for all the work done so far. Though I later made more changes in chapter order and content, the EV system enabled me to track progress and accurately estimate when I would finish the job.

TABLE 7.3 EXAMPLE SCHEDULE PLANNING TEMPLATE

Date	Week	Plan Hours	Cum. Hours	Actual Hours	Cum. Act. Hours	PV	Cum. PV	EV	Cum. EV
3/8	1	20	20	12.60	12.6	0.93	0.93	0	0
3/15	2	20	40	33.47	46.07	0.50	1.43	0.93	0.93
3/22	3	20	60	25.70	71.77	1.32	2.75	0.50	1.43
3/29	4	10	70	16.58	88.35	0	2.75	0	1.43
4/5	5	15	85	22.05	110.40	0	2.75	4.09	5.52
4/12	6	15	100	32.93	143.33	2.75	5.50	0	5.52
4/19	7	20	120	32.08	175.42	0	5.50	4.09	9.61
4/26	8	15	135	25.80	201.22	0	5.50	0	9.61
5/3	9	20	155	34.58	235.80	3.42	8.92	0	9.61
5/10	10	15	170	31.65	267.45	0	8.92	4.07	13.68
5/17	11	20	190	29.65	297.10	0	8.92	0	13.68
5/24	12	20	210	31.95	329.05	2.75	11.67	0	13.68
5/31	13	20	230	46.68	375.73	0	11.67	6.84	20.52
6/7	14	20	250	31.88	407.61	0	11.67	2.75	23.27
6/14	15	20	270			4.09	15.76		
6/21	16	20	290			0	15.76		
6/28	17	20	310			0	15.76		
7/5	18	20	330			4.09	19.85		
7/12	19	20	350			0	19.85		
7/19	20	20	370			0	19.85		

(continued)

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 122

TABLE 7.3 (continued)

Date	Week	Plan Hours	Cum. Hours	Actual Hours	Cum. Act. Hours	PV	Cum. PV	EV	Cum. EV
7/26	21	20	390			3.42	23.27		
8/2	22	20	410			0	23.27		
8/9	23	20	430			0	23.27		
8/16	24	20	450			4.09	27.36		
8/23	25	20	470			0	27.36		
8/30	26	20	490			1.99	29.35		
9/6	27	20	510			0	29.35		
9/13	28	20	530			2.75	32.10		
9/20	29	20	550			0	32.10		
9/27	30	20	570			0	32.10		
10/4	31	20	590			4.09	36.19		
10/11	32	0	590			0	36.19		
10/18	33	0	590			0	36.19		
10/25	34	10	600			0	36.19		
11/1	35	20	620			0	36.19		
11/8	36	20	640			0	36.19		
11/15	37	20	660			4.09	40.28		
11/22	38	20	680			0	40.28		
11/29	39	20	700			0	40.28		
12/6	40	20	720			3.42	43.70		
12/13	41	20	740			0	43.70		
12/20	42	20	760			2.75	46.45		
12/27	43	0	760			0	46.45		
1/3	44	10	770			0	46.45		
1/10	45	10	780			0	46.45		
1/17	46	10	790			2.75	49.20		
1/24	47	10	800			0	49.20		
1/31	48	10	810			3.42	52.62		

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 123

TABLE 7.4 EXAMPLE TASK PLANNING TEMPLATE

Phase	Task	Plan Hours	Cum. Plan Hours	PV	Cum. PV	Plan Week	Actual Hours	Actual Week	EV	Cum. EV
Plan	Book Plan	15.10	15.10	0.93	0.93	1	13.8	2	0.93	0.93
Draft	Preface	8.14	23.24	0.50	1.43	2	35.3	3	0.50	1.43
Draft	Chapter 1 Draft	21.70	44.94	1.32	2.75	3	28.6	10	1.32	10.93
Draft	Chapter 2 Draft	44.75	89.69	2.75	5.50	6	62.5	10	2.75	13.68
Draft	Chapter 3 Draft	55.60	145.29	3.42	8.92	9	41.2	13	3.42	20.52
Draft	Chapter 4 Draft	44.75	190.04	2.75	11.67	12	27.3	14	2.75	23.27
Draft	Chapter 5 Draft	66.45	256.49	4.09	15.76	15				
Draft	Chapter 6 Draft	66.45	322.94	4.09	19.85	18	67.6	7	4.09	9.61
Draft	**Planning Drafts**	**0**	**322.94**	**0**	**21.23**	**18**				
Draft	Chapter 7 Draft	55.60	378.54	3.42	23.27	21				
Draft	Chapter 8 Draft	66.45	444.99	4.09	27.36	24				
Draft	Chapter 9 Draft	32.55	477.54	1.99	29.35	26				
Draft	**Quality Drafts**	**0**	**477.54**	**0**	**31.40**	**26**				
Draft	Chapter 10 Draft	44.75	522.29	2.75	32.10	28				
Draft	Chapter 11 Draft	66.45	588.74	4.09	36.19	31	57.5	5	4.09	5.52
Draft	Chapter 12 Draft	66.45	655.19	4.09	40.28	37				
Draft	**Design Drafts**	**0**	**655.19**	**0**	**43.08**	**37**				
Draft	Chapter 13 Draft	55.60	710.79	3.42	43.70	40				
Draft	Chapter 14 Draft	44.75	755.54	2.75	43.45	42				
Draft	Chapter 15 Draft	44.75	800.29	2.75	49.20	46				
Draft	Appendix A Draft	55.60	855.89	3.42	52.62	48	68.8	13	3.42	17.10
Draft	**All Drafts**	**0**	**855.89**	**0**	**52.62**	**48**				

7.7 Comments on the EV Example

This example illustrates the importance of task size. The tasks shown in Tables 7.2 and 7.4 averaged 44.5 hours each with a maximum of 66.45 hours and a minimum of 8.14 hours. Furthermore, out of the 48-week schedule, this plan earned EV in only 18 of the 48 plan weeks. This does not provide enough feedback to manage even a one-person job like this. This is a simplified example, however, because my

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 124

actual book-writing process had 21 tasks for each chapter, with 10 of them used to produce the chapter drafts. By getting feedback on my progress every week, I had a highly motivating sense of steady progress. This helped me to maintain my productivity.

In addition, as shown in Tables 7.3 and 7.4 and Figure 7.3, my plan for writing the first PSP book was seriously off. At 14 weeks I had reached 23.3 EV, which is the level planned for week 23. This meant that I was nine weeks ahead of schedule and would finish the manuscript four months ahead of the plan. To make these calculations, I used the same method that TSP teams use to track their work. The kinds of data TSP teams use are shown in Figure 7.3. In 14 weeks, I had earned 23.3 points of EV: a rate of 1.66 EV points a week. To finish the manuscript, I had to earn 52.62 – 23.3 = 29.32 more EV. At my historical rate, this would take 17.66 more weeks, meaning that I would finish 16 weeks ahead of schedule, or in week 32 instead of the planned week 48. This assumed that I would continue working at the same rate and that nothing would change the size of the tasks.

Another important conclusion can be drawn from Figure 7.3. The row for *To-date hours for tasks completed* compares the planned time for the completed tasks with the actual times. The work done to date was planned to take 378.5 hours, but it actually took me 402.6 hours, an underestimate of 6.37%. However, I averaged 63% more task hours than planned, which more than compensated for my 6% project underestimate. I had based the task-hour estimate on my previous book, for which I had averaged 26 hours per week. Because I did not believe I could maintain that same rate on the new book, I used a more conservative 20 hours per week. This was not a planning system problem, but rather an error in planning judgment. Because I was able to work at home and didn't have any children around to distract me, I was able to achieve an unusually high task-hour rate. I suggest that you not use these data as an example, at least until you have data on your own personal task-hour experience.

TSP Week Summary - Form WEEK

Name	Watts Humphrey	Date	7-Jun
Team	PSP		
Status for Week	14	Cycle	
Week Date	6/7		

Weekly Data	Plan	Actual
Project hours for this week	20.0	31.9
Project hours this cycle to date	250.0	407.6
Earned value for this week	0.0	2.8
Earned value this cycle to date	11.7	23.3
To-date hours for tasks completed	378.5	402.6
To-date average hours per week	17.9	29.1

FIGURE 7.3 THE TSP WEEKLY SUMMARY REPORT FORM

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 125

A planning system based on historical data will accurately project future performance if you continue working as before. I had not expected to spend so many hours per week, but I was wrong. By using the earned value system, I recognized this planning error after only a few weeks. After the first month of a two-year project, I could see that I would finish the manuscript draft much earlier than planned. After the first ten weeks, I could confidently commit to finishing the manuscript draft four months early. That is when I actually did complete it.

I had the reverse problem in writing the final manuscript. Because I obtained a great deal of feedback from manuscript reviews, the rewriting process took much longer than planned. However, by using earned value tracking, I could quickly recognize and adjust for this problem. With all of these changes, I sent the final manuscript to the publisher in June, exactly as planned a year and a half earlier.

7.8 Estimating Accuracy

It is helpful to think about estimating with an analogy to throwing darts at a target to get a number. Assume that your historical data have values for ten prior throws that clustered around 100, with a couple at 80 and one out at 60. Statistically, your new estimate is like another dart throw. The presumption is that its likely value is best represented by the variation of the previous throws. If, however, you use a bow and arrow, or switch to a different-sized dart, or stand twice as far away, your previous data will not be a good predictor of the error in the new throw.

With PROBE, the quality of an estimate is a direct function of the quality of the historical data. As shown in the dart-throwing example, however, it also depends on the degree to which these data correspond to what you plan to do. For example, if you changed your design method, built a much larger program, or developed an unfamiliar type of application, your historical data might not produce a very accurate plan. You could still use the PROBE method, but you should recognize the likely higher estimating error.

Process stability is also important. With the PSP exercises, you will use several different processes and learn a number of new methods. Because these changes will probably cause fluctuations in your performance, the quality of your estimates will likely have similar fluctuations. Although this instability is a necessary part of learning, a more stable process will probably improve your estimating accuracy.

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 126

7.9 The Prediction Interval

When using the regression method to make estimates, you can calculate the likely error in each estimate. The prediction interval gives the range within which the actual program size or development time will likely fall. Figure 7.4 shows actual data for nine programs, as well as the 70% prediction interval for the ninth estimate. This prediction interval shows the range within which the actual values have a 70% chance of falling. As the figure shows, seven of the nine points are between the upper and lower prediction ranges, and two are very near the lower prediction interval limit.

As you can also see from Figure 7.4, when the data points cluster closely around the regression line, the prediction interval range is narrower. This means that when you have historically made accurate estimates, the prediction interval for the next estimate will be narrow. Similarly, when your estimating errors have been large, the prediction interval will be large.

The prediction interval is very sensitive to outlier points. As discussed in Chapter 6 (see p. 106), outlier points are points that are unusual in one or more respects. Do not exclude points simply because they were bad estimates. Only do so if they do not represent what you plan to do. In this example, the developer had actually written ten programs, but he excluded one because he had misunderstood the requirements and started over partway through the job. The effect of including an outlier point can be seen in Figure 7.5. Including the tenth point raises the regression line above six of the ten data points, thus introducing an estimating bias. In addition, with this point, the 70% prediction interval includes 90% of the data points. An excessively large prediction interval indicates data problems. In this case, the prediction interval includes many more of the points than it should. A wide 70% prediction interval that included only about 70% of the points, however, would merely indicate a lot of variation in the data.

The prediction interval calculations are given in Box 7.1 and are described in the assignment kit for one of the PSP exercise programs. Many TSP tools automatically calculate the prediction interval, but to make these calculations yourself, write the appropriate program or produce a spreadsheet to do the calculations.

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 127

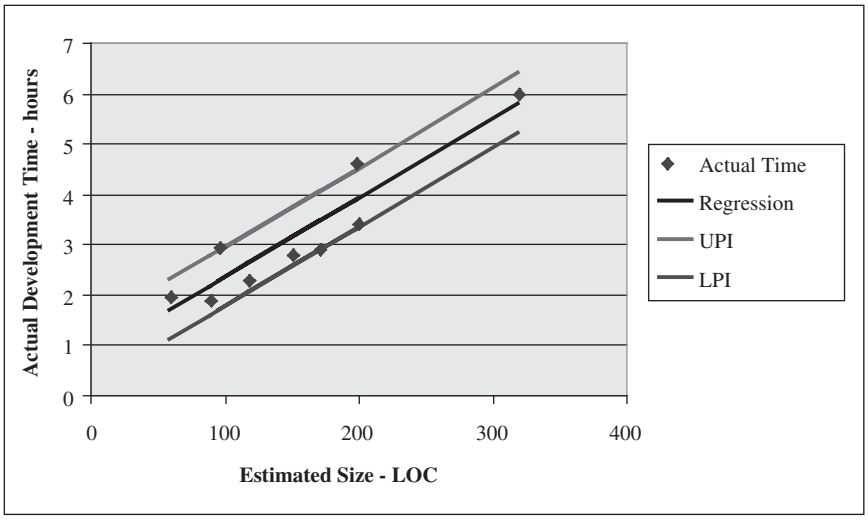

FIGURE 7.4 THE PREDICTION INTERVAL

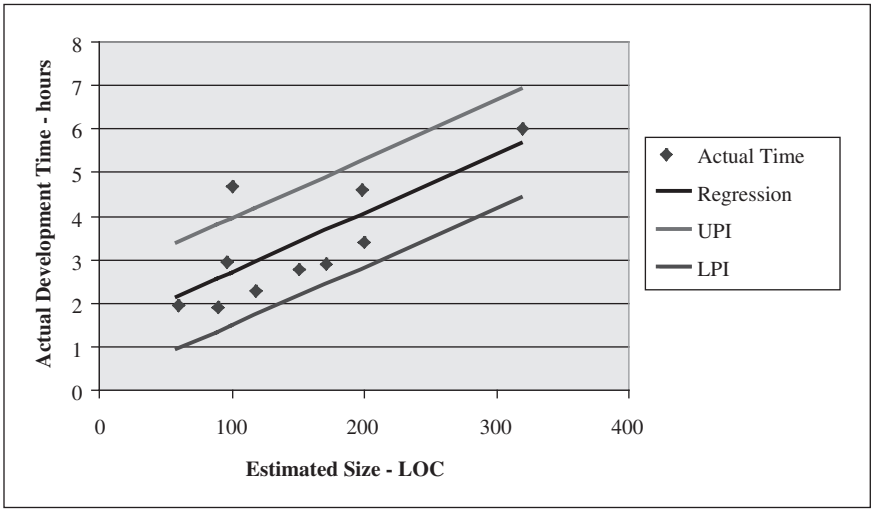

FIGURE 7.5 PREDICTION INTERVAL WITH OUTLIER POINT INCLUDED

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 128

BOX 7.1 CALCULATING THE PREDICTION INTERVAL

1. To determine the likely error of a regression projection for the term x_k from a set of n data values x_i and y_i, first calculate the variance of the y_i terms from the regression line:

$$\text{Variance} = \sigma^2 = \frac{1}{n-2}\sum_{i=1}^{n}\left(y_i - \beta_0 - \beta_1 x_i\right)^2$$

$$\text{Standard Deviation} = \sigma = \sqrt{Variance}$$

2. Find the value $t(p, n)$ of the t - distribution for p = 70% or 90% and $d = n - 2$ degrees of freedom (values for the t - distribution are shown in Table 3.1 on p. 38).
3. Calculate the value of the prediction interval range:

$$\text{Range} = t(p,n)\sigma\sqrt{1 + \frac{1}{n} + \frac{\left(x_k - x_{avg}\right)^2}{\sum_{i=1}^{n}\left(x_i - x_{avg}\right)^2}}$$

   Here, x_{avg} is the average of the x terms, and x_k is the term for which the estimate is made and the prediction interval is to be calculated.
4. Calculate the upper and lower prediction intervals, UPI and LPI, where y_k is the estimate ($y_k = \beta_0 + \beta_1 x_k$).

   $\text{UPI} = y_k + \text{Range}$

   $\text{LPI} = y_k - \text{Range}$

7.10 Alerting Management to Changes

With sound plans and the EV tracking system, you can see precisely where you stand on a job. If you are ahead of schedule and believe that this rate of progress will continue, you could tell management about the improvement. Similarly, if you are falling behind, you could warn management and possibly even get their help. When you have schedule problems, don't just hide your head; talk to your manager and see what he or she can do to help. Although Brooks' law says that adding staff late in a project will delay it, with a plan and a defined process, adding trained people will usually accelerate the work (Brooks 1995). When you have a detailed plan and precise status measures, you can explain the issues to management and

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 129

help them resolve any problems that block your progress. When you have problems, use your managers, but give them the data they need to help you.

An example shows how management can often help in ways you might not have considered. Some years ago, before everybody had workstations, I was walking through the machine room of IBM's Poughkeepsie programming development laboratory late one night and heard loud cursing. This poor programmer was late finishing testing and was frustrated because the file hardware kept failing. He told me that this machine failed so often that he could not get his program tested. He was developing the I/O software for a new file system and he was under intense pressure to finish it.

The next morning, I called the vice president in charge of the file hardware and discovered that they had a warehouse full of these machines. They couldn't ship them until the program was ready. The VP arranged to have four machines delivered to the machine room that morning, along with a maintenance engineer to keep them running. This accelerated the testing and the program was soon completed. If you have a problem that you can't handle, go to your managers for help. With good schedule planning and tracking information, you can explain the problems clearly enough so that they can help you solve them.

7.11 Planning Considerations

Planning is a learning process: You learn from each estimate and plan, from evaluating your data, and from reviewing your process. The best time for this learning is during the project postmortem. Based on experience with PSP estimating and planning, I have developed some planning guidelines. The following sections describe three aspects of planning: motivation, productivity, and part-time assignments.

Motivation

Earned value tracking can cause you to delay starting tasks that do not earn EV credit. With the first version of this book, for example, I was reluctant to work on the unplanned Appendix A. When I finally adjusted my plan to assign EV credit to it, however, I had no further problems. Promptly assign EV credit to essential new tasks. Conversely, if some partially completed task is no longer necessary, drop the remaining work on that task from the plan. Remove the earned value for the deleted work and increase the value of everything else you have done.

Even when you are working productively, if you cannot sense steady progress, it is easy to get discouraged. This is one reason why software developers spend so little time validating requirements and documenting designs: such work rarely feels

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 130

very productive. When you are in a hurry and do not seem to be making progress, you are likely to switch to something that feels more productive. Because most developers view coding and testing as productive, that is where they prefer to spend their time. By showing progress during *all* phases of a programming job, however, the EV method can actually make the early requirements and design work feel productive, thereby improving the quality and productivity of your work.

The key is to adjust your plan with new task additions and deletions whenever necessary. I do so about every week or two. For small task changes, make minor plan adjustments, but for major task additions or deletions, redo the plan. Because every project is unique, however, the only general guideline is to regularly check whether the plan represents the way you are actually working. If it does not, either change the plan or, if the plan is still the right way to do the work, follow the plan.

Productivity

When you use the regression method and your productivity is improving, your historical data will tend to produce overestimates. This is a problem when you use data from an evolving process because the regression method uses average productivities and not productivity trends. If your productivity has been changing, adjust your planning process to use only your most recent data.

Part-Time Assignments

One of the more common problems developers face is multiple project assignments. When management is shorthanded, they often resort to assigning developers to several projects. Although it is normal to have some cleanup work to do on a project you are just completing or to be called back to help on a previous job, it is almost impossible to be productive on more than one major project at a time. Occasional interruptions to give temporary help are rarely a problem, but it is unreasonable to expect developers to give priority to more than one project at a time. Part-time assignments are troublesome for three reasons: task time, task switching, and team coordination.

1. **Task Time:** When simultaneously assigned to several projects, you must divide your weekly task hours among all the projects. If you were averaging 15 task hours per week and you were assigned to 3 projects, you could expect to average only 5 hours per week on each. It is hard enough to get much done in 15 hours a week, let alone 5. Even with 2 projects, an average of 7.5 task hours a week would still not provide enough time to accomplish very much.

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 131

2. **Task Switching:** We all have task-switching problems when we are interrupted in the middle of a challenging task. It is hard enough to produce a complex design, for example, without having to do it in short, disconnected bursts of time. Every time you return to the task, you must reconstruct the mental context for the work before you can make further progress. Even then, if you did not stop with a fully documented record of the work done to date, there is a good chance you will overlook something critical.
3. **Team Coordination:** In addition to the task-time and task-switching problems, multiple assignments make team coordination very difficult. When some team members are also assigned to different projects, it is hard for the others to arrange team meetings, inspection reviews, or design sessions. The full-time team members will not know where the part-time members are or be able to rely on them for help. This is inefficient and it destroys team spirit. Without a cohesive and dedicated group, teams rarely have the energy and enthusiasm that produces truly great work.

7.12 Summary

The PROBE method shows you how to use historical data to produce accurate estimates. Then, once you have estimated the size and development time for a project, you can produce the project schedule. With an estimate of the hours you will spend on project tasks each week, you can spread the estimated development time over the project schedule. Weekly task hours are typically about 50% or less of the normal working time. With an estimate of weekly task hours and a detailed task plan, you can calculate when each task will start and end. You can also set dates for the key project milestones. The milestone dates then provide intermediate project goals and a basis for reporting job status.

Earned value provides a way to measure and track job progress. It establishes a planned value for every task based on that task's percentage of total project development time. When you complete a task, you earn the planned value for that task and it is added to your project's cumulative earned value. Earned value provides a common measure for the value of every task, regardless of the type of work involved. A task gets earned value only when it is completed. Partially completed tasks get no credit.

When using the PROBE method, you use your data to make size and development time estimates and to calculate the likely error of these estimates. The prediction interval gives the 70% or 90% range around an estimate where the actual development time or product size will fall 70% or 90% of the time. Although the prediction interval is not a forecast, it does provide a useful guide to the expected error of the estimate.

31422 07 pp. 109-132 r2ah.ps 2/9/05 5:21 PM Page 132

7.13 Exercises

The standard assignment for this chapter includes exercises R3 and R4 and uses PSP1.1 to write one or more programs. For the R3, R4, and program assignment kits, see your instructor or get them at **www.sei.cmu.edu/tsp/psp**. These kits contain the assignment specifications and the PSP1.1 process scripts. In completing this assignment, faithfully follow the PSP1.1 process, record all required data, and produce the R3, R4, and program reports according to the specifications given in the assignment kits.

References

Boehm, B. W. *Software Engineering Economics.* Englewood Cliffs, NJ: Prentice-Hall, 1981.

Brooks, F. P. *The Mythical Man-Month.* Reading, MA: Addison-Wesley, 1995.

Humphrey, W. S. *Managing the Software Process.* Reading, MA: Addison-Wesley, 1989.

———. *A Discipline for Software Engineering.* Reading, MA: Addison-Wesley, 1995.

Zells, L. *Managing Software Projects.* Wellesley, MA: QED Information Sciences, Inc., 1990.