4

Planning

Software development plans are often incomplete and inaccurate. During the 27 years when I worked at IBM, we once needed a critical new function for the OS/360 programming system. The engineering estimate was $175,000. Naively, that is all the funding I requested. Some months later, the developers found that the work would cost $525,000. They had omitted many necessary tasks from their original plan. They had forgotten documentation; testing; the integration, build, and release processes; and quality assurance. Sure enough, however, the coding and unit test costs were about $175,000. They had made a pretty good estimate, but their plan was painfully (for me) incomplete. I had to make up the difference out of department funds.

The problem is that few software organizations have a planning process that ensures that plans are complete, thoroughly reviewed, and properly approved. Even worse, few software developers have the knowledge and experience to make sound plans. When you start a project, management typically states very clearly when they want the job done, but they are not usually very clear about much else. It is not that they are bad, lazy, or incompetent; it is just that, except for the schedule, most requirements are complex and cannot be easily described. By emphasizing the schedule, management gives the impression that it is their highest-priority concern. But, although the schedule is important, you must also address all of management's stated and implied goals. In addition, while doing this, you must do your best to meet management's desired schedule.

PSP training will help you to build the needed planning skills. Planning is the first step in the PSP for three reasons. First, without good plans you cannot effectively manage even modestly sized software projects. Second, planning is a skill that you can learn and improve with practice. Third, good planning skills will help you to do better software work. This chapter introduces software planning and provides a general overview of the planning process. It discusses what a plan is and what it should contain. It reviews the reasons why planning is important and it describes the elements of an effective planning process. It then discusses plan quality and how your work with the PSP will help you to do better work and be a more effective team member.

4.1 The Planning Process

What management really wants is a completed project *now*, at no cost. Anything else is a compromise. However, because they know that development takes time, they will push for the most aggressive schedule that you and your team will accept as a goal. Not unreasonably, they believe that projects with the shortest schedules finish before ones with longer schedules. Therefore, they will keep pushing until they believe that the schedule is the shortest one you will agree to meet. As developers, however, we are responsible for doing the work. With an impossibly short schedule, it is difficult if not impossible to make a usable plan. Then, without a plan, we are generally in such a rush to code and test that we cut corners and don't do as good a job as we could. Such projects generally take much more time than they would with a realistic plan.

Typically, when management asks for an aggressive date, the developers tell them that this date doesn't allow enough time to do the work. Management then insists that the date is firm, and the team generally caves in and agrees to do its best. Such teams start out in trouble and almost always end up in trouble. As Greg's team found in Chapter 1, the best answer to this problem is to make a detailed plan and to review it with management. Because most managers want a schedule you *can* meet, they will negotiate a thoughtfully made plan with you. If you present a convincing case, they will then end up agreeing to your schedule. To have such a debate, however, you must know how to make a plan and how to defend it to management. PSP training provides these skills. Later, on a team, the TSP shows you how to prepare for and handle the plan negotiations with management.

4.2 Why Make Plans?

In software engineering, as in other fields, our role as developers is to devise economical and timely solutions to our employer's needs. To do this, we must consider costs and schedules. The connection between cost estimating, scheduling, and the planning process can best be illustrated by an example. Suppose you want to put an addition on your home. After deciding what you want and getting several bids, most of which are around $24,000, you pick a builder who offers to do the job in three months for $20,000. Although this is a lot of money, you need the extra space and can arrange for a home-equity loan. You then sign an agreement and the builder starts the work. After about a month into the job, the builder tells you that, because of unforeseen problems, the job will take an extra month and cost an additional $4,000.

This presents you with several problems. First, you badly need the space, and another month of delay is a great inconvenience. Second, you have already arranged for the loan and don't know where you can get the extra $4,000. Third, if you get a lawyer and decide to fight the builder in court, all the work will stop for many months while the case is settled. Fourth, it would take a great deal of time and probably cost even more to switch to a new builder in the middle of the job.

After considerable thought, you decide that the real problem is that the builder did a sloppy job of planning. Although you do not know precisely what went wrong, the builder probably got low bids on some of the major subcontracts, such as the plumbing, plastering, or painting. You can endlessly debate what the job should cost, but essentially the problem was caused by poor planning. If you had originally been given the $24,000 price, you could have decided then whether to proceed with that builder and how to finance the work. The odds are good that at this point you will try to negotiate a lower price but continue with the current builder. Because the other bids were close to $24,000, you know this is a pretty fair price. You would not use this builder again, however, and would probably not recommend him to anyone else.

The problem with incompetent planning is that everybody loses: customers receive late and more costly products, management must tie up more resources, and the developer gets a bad reputation. To be successful, businesses must meet their commitments. To do our part, we must produce plans that accurately represent what we will do.

Planning is serious business. It defines commitments and supports business decisions. Well-thought-out plans will help you to make commitments that you can meet, and enable you to accurately track and report your progress. Personal planning skill will be even more important when you work on a development team. The overall team plan is most likely to be realistic when it is built from competently made team-member plans. As you practice the methods described in this and the next four chapters, you will learn how to make competent plans.

4.3 What Is a Plan?

"The project plan defines the work and how it will be done. It provides a definition of each major task, an estimate of the time and resources required, and a framework for management review and control. The project plan is also a powerful learning vehicle. When properly documented, it is a benchmark to compare with actual performance. This comparison permits the planners to see their estimating errors and to improve their estimating accuracy" (Humphrey 1989). Plans typically are used as the following:

- A basis for agreeing on the cost and schedule for a job
- An organizing structure for doing the work
- A framework for obtaining the required resources
- The standard against which to measure job status
- A record of what was initially committed

The connection between plans and commitments is extremely important. Every project starts as a new endeavor. At the outset, the project must be created out of thin air. New projects typically start with no staff. A manager, user, or customer must commit funds, and some workers and suppliers must be convinced to participate in the work.

For substantial projects, management's first step is to assemble a planning and proposal team and produce an overall plan. Without a clear and convincing plan, they will not be able to get funding, hire the staff, and arrange for all the facilities, supplies, and other support needed to do the work. Nobody wants to pay for an undefined job, and few people will work on a project that has unclear objectives. Because an accurate plan is the essential first step in creating a successful project, planning is an important part of every project.

4.4 The Contents of a Software Plan

The focus of this and the next three chapters is on the personal planning process and the products it produces. Although personal planning, team planning, and project management are all related, the objective here is to practice personal planning for small software projects. Because detailed planning is the key to accurate planning, learning to make accurate plans for small projects is the essential first step in learning how to make accurate plans for large projects.

In deciding what a plan should contain, consider the needs of the people who will use the plan and what they will do with it. PSP plans have two users: you and your customers. You need four things from a plan:

1. Job sizing: How big is this job and how long do you expect it to take?
2. Job structure: How will you do the work? What will you do first, second, and so on?
3. Job status: How will you know where you are? Will you finish on time and are the costs under control?
4. Assessment: How good was the plan? Did you make any obvious errors? What mistakes can you avoid in the future and how can you do a better planning job?

For your personal plans, the possible users are the PSP instructor, your coworkers, your manager, or an end user. These people also want four things from your plan:

1. What is the commitment? What will you deliver, when, and at what cost?
2. Will the product be what they want? Will there be any interim checkpoints on product performance and quality?
3. Is there a way to monitor progress? What kind of warning can they expect for cost, schedule, or quality problems? Will scope changes be clear and identifiable?
4. Will they be able to evaluate the job? Can they separate planning problems from poor management? Is there a way to assess product quality during the job?

Although these issues will not be very complex for the small programs developed with the PSP, they will clarify several points:

- The plan must be based on doing a defined piece of work. Otherwise, there is no way to make an accurate plan, and you could even build the wrong product.
- The job steps should be clearly defined and measurable. This will provide a framework for the plan and a way to track progress.
- You should check your plan with the customer, manager, or instructor before starting the work. This is always a good idea and is essential for any but the smallest jobs.
- Periodically report progress to your customers and managers. For any but the shortest projects, this will alleviate concern and build trust.

When you plan personal work, your objective is to estimate the cost and schedule of the job you will actually do. Then, once you complete the job, if the estimate differs from the actual result, you have a planning problem. Although

there may have been development problems, the planning objective was to forecast what would actually happen. With the PSP, the saving grace is that the projects are simpler and more consistent than is typical for development projects. Although overall gross data on large projects are rarely helpful in planning the next job, the PSP data are. In addition, when estimating your own work, you decide how to do the job, you produce the plan, and you do the work. Then, when you are done, you can compare the plan with what you actually did.

4.5 Planning a Software Project

To make a personal plan, take the following steps:

1. Start with an explicit statement of the work to be done and check it carefully to ensure that you understand what is wanted. Even though the initial requirements will often change, start with the best requirements statement you can get.
2. Break any projects that take more than a few days into multiple smaller tasks and estimate each one separately. More detail means greater planning accuracy.
3. In making the estimates, compare this job with historical data on prior similar jobs.
4. Document the estimate and later compare it with the actual results.
5. When the requirements change, change the plan and get your customers and managers to agree with the new plan before you accept the change.

Before making an estimate, first ensure that the historical data you use are for similar work. On one project, for example, a software professional used historical data from a prior project to make the estimate. Partway through the job, he found that the work was taking three times longer than estimated. Although the number of requirements had increased a little, each one was taking much longer than expected. On examination, he found that the current requirements involved complex system functions, whereas the previous project concerned requirements for small, stand-alone applications. In this new work, he had to deal with many more people to resolve the system issues and to get the requirements approved. By tracking and analyzing his work, he could see the problem in time to alert management and get help.

The Planning Framework

The PSP planning framework is shown in Figure 4.1. The tasks you perform are shown in the rectangles; and the data, reports, and products are shown in the ovals. Starting with a customer need, the first step is to define the requirements. Next, to

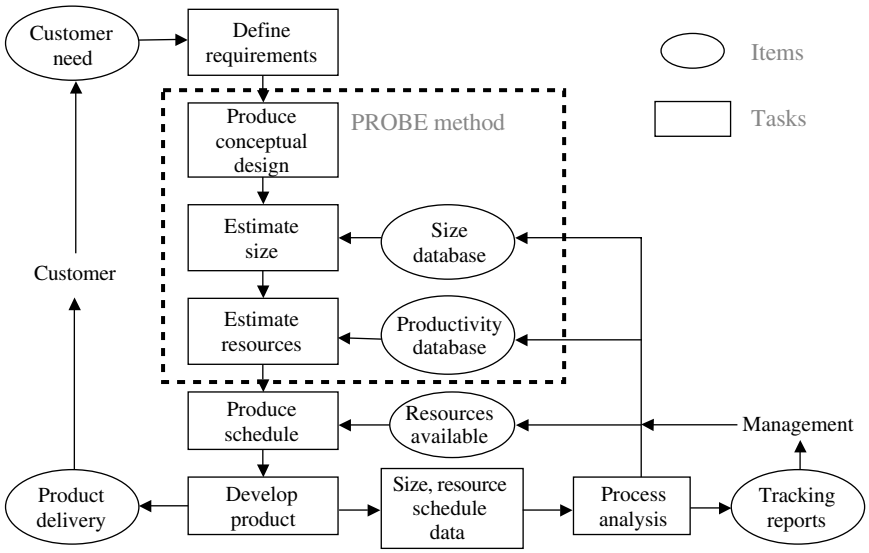

FIGURE 4.1 PROJECT PLANNING FRAMEWORK

relate your plan to the actual product you will build, produce a conceptual design. From the conceptual design, you can now estimate the size of the new product. Then, with a size estimate, use historical productivity data to estimate how long the work will take. Finally, starting with the planned project start date, spread your available working hours over the project calendar to produce the schedule. The final step is to calculate the date when you expect to finish the work.

With the plan in hand, and assuming that you have all of the needed information and facilities, you can start development. Then, during the job, you track the time you spend and the sizes of the products you produce. This provides the data you need to make future plans.

4.6 The Conceptual Design

To make an accurate plan, start with a conceptual design. It defines a preliminary design approach and names the product elements and their functions. Do not produce the complete design, but merely postulate the principal product parts and the functions they will perform. This is an abstraction process in which you say, "If I

had parts that performed functions A, B, and C, I would know how to build this product."

There are many ways to build a product to meet almost any requirement, and these different approaches will generally involve different amounts of work. For example, out of 810 times that the PSP exercises were developed from the identical requirements statements, the size distribution was as shown in Table 4.1. Although some of this enormous size range is due to skill and experience variations and some from use of embedded language functions, the principal difference is from different design strategies. By producing the conceptual design, you can base your estimate on the particular design strategy you plan to follow. This helps you to make a realistic estimate.

Although the conceptual design is important, it is not the actual design—and you should not feel obliged to implement it. You produced it solely to make the plan. When you actually do produce the design, examine several approaches, brainstorm design ideas with associates, or examine the designs of previously developed products. For larger projects, it is even a good idea to do this while producing the conceptual design. A few minutes spent in such exploration can often reveal approaches that could reduce the development work by large factors. Don't feel constrained to implement a design just because it is the first one you thought of. Although it is possible that your first idea will turn out to be the best, be suspicious of it until you have examined one or two alternatives and found that it is still best.

When estimating a large product, producing a conceptual design can be a significant task. Though you may question the need to go into a lot of detail just to make an estimate, the issue is accuracy. If you need an accurate estimate, you must refine the conceptual design into sufficient detail to define parts that you know how to build. Even for fairly large programs, however, this need not take more than a few hours.

In producing the conceptual design, remember that the objective is to make a plan, not to produce the design. You want to spend just enough time thinking about the product's design so that your estimate is more than a guess. The trick is

TABLE 4.1 PSP PROGRAM SIZE RANGES IN LINES OF CODE (LOC)

Program Number	1	2	3	4	5	6	7	8
Maximum	613	872	560	578	338	542	1043	913
Upper Quartile	113	97	104	117	113	197	199	279
Average	88.3	91.5	93.2	98.4	89.5	164.1	162.9	232.4
Median	76	62	71	84	81	153	138	208
Lower Quartile	54	43	50	63	54	114	100	156
Minimum	7	8	9	29	4	22	17	33

to produce a conceptual design that provides enough guidance for estimating but does not delve too deeply into the design work itself. With the small programs like those in the PSP course, the conceptual design work should not take more than a few minutes, 20 at most.

The challenge with the PSP is to limit the time you spend on the conceptual design. Although a more complete design will always produce a more accurate plan, the objective with the PSP is to quickly produce the minimum design needed to make a reasonably accurate plan. Because there is no magical approach that works best for everyone, experiment with various methods until you find one that works well for you.

4.7 Plan Quality

The objective of the planning process is to produce an accurate plan. The next three chapters address how to measure, predict, and improve planning accuracy. As you make plans for your PSP programs, don't worry about the errors in your individual plans, as long as they appear to be random. Although you should strive to reduce estimating error, some error is unavoidable. The key, however, is to make unbiased or balanced plans. When you can make balanced plans, five out of about every ten plans will be overestimates and five will be underestimates. This is important because when you work on a TSP project, you and your teammates will make individual plans to combine into the team plan. If your plans are all balanced, the errors in the individual plans will tend to cancel each other out, resulting in a much more accurate team plan. The following chapters describe size measures, software estimating, and the PROBE method for producing balanced plans.

4.8 Planning Issues

Although most planning issues are covered in the following chapters, three general issues must be covered first: requirements creep, planning tools, and the statement of work. Although these issues will probably not be significant in writing the PSP exercise programs, they will be important when you work on a project team.

Requirements Creep

The requirements for engineering work always grow. On my team at SEI, team workload increases by an average of 1% per week: Over a 20-week plan cycle, the

total work increases by about 20%. The first strategy for handling requirements creep is to plan for every proposed change and get all parties involved to agree to the cost and schedule consequences. Second, factor in a workload growth allowance during planning. I suggest you follow both strategies.

Estimating and Planning Tools

Many estimating and planning tools are available but they all require that you first make a development estimate. These tools can be helpful in checking estimates and in managing planning mechanics. Example cost estimating tools are COCOMO II, REVIC, and SEER, and example planning tools are Artemis Views, Microsoft Project, and Primavera.

The Statement of Work

Depending on the situation, a statement of work (SOW) can provide a useful record of your plan. It should contain a summary of customer and management agreements, key project assumptions, the names of the workers, and key dependencies. By attaching a copy of the Project Plan Summary form and sending the SOW to your management and the customer, you both document the plan and quickly identify any disagreements or misunderstandings. Though an SOW is usually helpful for a team, it is not generally required for individual work. When in doubt, however, produce an SOW and give it to your manager and the customer.

4.9 Summary

Planning is the first step in the PSP for three reasons. First, without good plans you cannot effectively manage even modestly sized software projects. Unplanned projects are often troubled from the start and are almost always in trouble at the end. Second, planning is a skill that you can learn and improve with practice. If you don't learn how to make good plans now, planning skills are almost impossible to pick up on the job. Third, good planning skills will help you to do better software work. Although the connection between planning and product quality is not obvious, unplanned projects are invariably under schedule pressure and the developers can rarely take the time to do an adequate job of requirements, design, or quality management.

In software engineering, as in other fields, the role of the developers is to devise economical and timely solutions to the employer's needs. This is the essential

issue of the planning process: making plans that accurately represent what you will do. This in turn will help you to better manage your personal work and to be a more effective team member. Although personal planning is an important part of project planning, it is only a part. Many more issues are involved in producing a complete plan for a large project. These larger plans, however, are most likely to be realistic when they are composed of the multiple personal plans of the individuals who will do the work. As the accuracy and completeness of these elemental plans improves, their composite will be of higher quality. Conversely, when individual plans are poorly developed, they provide a poor foundation for the overall plan. The PSP will help you build the skills needed to make balanced and accurate personal plans.

Reference

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