Estimating With Objects - Part I

Contents

This column starts a series on estimating. This is a big subject, however, so the series will take a number of months. Since there is a lot of interest in estimating, however, I have decided that this time is warranted. Objects can be very helpful in estimating. They can be helpful, that is, if engineers know how to use them. The proper use of objects for estimating, however, requires some work on your part. But then, few things in life are free.

My purpose with this series is to tell you how to make good estimates with objects. For those who are impatient to get to the punch line, the method I will describe is called PROBE. PROBE stands for PROxy Based Estimating. I developed PROBE as part of my research on the Personal Software Process (PSP). During this work, I developed 62 module-sized programs in Object Pascal and C++ and gathered a lot of data. Among other things, I found that it is possible to make rather good plans for even small programming jobs. I also found that the concepts of object oriented design can be a big help in planning. I have described PROBE and the PSP in some detail in my book, *A Discipline for Software Engineering*, Addison Wesley, 1995. Also, some of the material in this and subsequent columns is taken from my forthcoming book *Introduction to the Personal Software Process*. It will be published by Addison Wesley around the end of this year.

The PROBE method is taught as part of the PSP course, both by the SEI and a growing number of universities. Engineers who have taken the course report that they find the method very helpful. We don't yet have much data on industrial use of PROBE but early indications are that the course benefits carry over into industrial practice. I will show some data on both class and industrial estimating experience at the end of this series of columns.

In this first column, I talk about why you should make estimates and then I briefly discuss the elements of estimating. Finally, this column closes with a brief outline of the estimating problem and the role of objects in addressing this problem.

First, a war story

Some years ago while I worked at IBM, I was put in charge of all their commercial software development. This was several thousand programmers in 15 laboratories and six countries. Most of the projects were seriously late and senior management was very upset. My job was to

straighten out the mess. The first thing I did was to go to the major laboratories to review their projects. To my surprise, none of them had any documented plans. When I asked the managers, they all said the right way to develop software would be to start with plans, but they didn't have time. This was nonsense. When engineers don't have the time to do the job they way they know they should, they always get into trouble. I thus insisted that the managers produce plans for all their projects. Since they had never made plans before, it took them a couple of months to do it. We even had to set up a special class on project planning. After they had the plans, however, they could establish realistic schedules for their work.

This step of making plans had a dramatic effect. These development groups had never before delivered a product on time. Starting with these plans, however, they did not miss a single date for the next two and a half years. This had such a positive effect that these software groups have been planning their work ever since. Planning is a critical part of a software engineer's job and to be an effective engineer, you need to know how to make plans. To do that, you need to practice, and to get the most practice, you should start making plans now and continue to do so for all your future projects.

The need for estimates and plans

When software engineers work on development teams, they generally have many interdependencies. As they decide how to start, they also decide what each team member will do and what parts are needed first. As part of these discussions, the engineers need to commit dates to their teammates and, to do this responsibly, they should plan their work. This is the only sound basis for committing completion dates. When they all make plans and share these plans with their teammates, engineering teams can better coordinate their work and more efficiently support each other.

Product plans are used by businesses for many of the same reasons engineers should use them: to plan and manage the work. Plans also provide early estimates of project cost. This is essential when work is done under contract and the customer wants to know the price in advance. It is also necessary when products are being developed for a product line. Here, project cost is a key determinant of product price and price is generally an important element of product marketability.

Engineers also use product plans to help them understand the status of their projects. When they have reasonably accurate and detailed plans they can judge project status against the plan. This tells them if they are late and need to negotiate additional resources or change the schedule. It could also indicate that they are ahead of schedule and could help their team mates or deliver the product early. When engineers document their plans, they can also compare these plans with their actual results and learn to make better plans.

When engineers have plans, they do more orderly work, have fewer last minute crunches, and are less likely to make mistakes. Product planning thus helps to produce better products. Good planning has so many advantages that I have never understood why every university engineering program does not teach planning and why engineers don't all make plans for every job they do.

The arguments against making plans

The only arguments I have ever heard against planning are that it takes a lot of work, good people are more important, and planning reduces creativity. First, planning does take work. After you learn how to plan and have some historical data, however, you will find that it doesn't take very much time. For 2,000 to 4,000 line of code program components, my planning times were consistently only an hour or so. While requirements and initial design can take a lot of time, that work must be done anyway. When you have experience, the planning itself doesn't take very long.

Yes, good people are most important, but what does that have to do with planning? Once you have good people, plans would help them to do better work. At IBM, we only hired the top 10% from the top engineering schools. By definition, all we had were good people. Until they started to plan, the quality of their code was irrelevant. Nobody thought they did good work.

Creativity is the same story. What is creative about staying up all night to whack out the last few defects? If you consider cleaning up sloppy code a creative endeavor, I'm surprised you have the time to read this column. You can expect to be fully employed at least until your spouse leaves you, you have a health breakdown, or your company goes out of business. Crisis schedules and unplanned projects often fail, not because the work was technically unsound but because the management was so bad. Poor management inhibits creativity, it doesn't foster it. Without planning, you have poor management and with poor management you cannot be consistently creative.

Planning, in fact, actually helps creativity. When making a plan, you think in global terms about what you are going to do and you consider alternatives and their likely consequences. By thinking first, you are more likely to get creative ideas and better able to capitalize on them.

Making a plan

The first step in making a plan is to define the product you plan to produce. This is almost always difficult and often must be done in steps. Before you start to build even a preliminary version of a product, however, you must at least know the requirements for that version. While this seems obvious, it is surprising how often people dive into the mechanics of product development and forget to define what they are trying to do. Only after you know what you want to do should you start thinking about how to do it. That is when to start planning.

A properly produced product plan includes three things:

- the size and important features of the product to be produced
- an estimate of the time required to do the work
- the schedule

Here, the product is the intended project result. It could be a working program, a detailed requirements statement, or a test plan. A product plan thus must identify the product to be produced and contain data on such project characteristics as estimated product size, the expected hours to do the work, and a schedule. Depending on the type of product and the sophistication of the planning process, a product plan may also contain other kinds of information. Examples could be product or process specifications, dependencies on other groups, or special testing or quality provisions.

The elements of estimating

The estimating process is shown in simplified form in the following figure. I will walk through many of the steps of this process in some detail in subsequent columns. In summary, these steps are:

- Define the Requirements: While requirements are critical, I will not discuss them further in this series. This does not mean that requirements are less important, just that this series of articles is about estimating and planning and not requirements.

  Figure 1: The PROBE Estimating Framework
  Figures in this file are displayed in a separate browser window. This window will remain open to display figures in this file, although it might be hidden behind other browser windows.

- Produce the Conceptual Design: The requirements deal with what the user wants or needs. To make a good plan, you need an estimate for what you intend to build. The conceptual design is your first cut at the product's design.
- Estimate Object Size: The next step in the PROBE method is to determine what objects are called for by the conceptual design. Then, using historical data on object size, you can estimate total object size.
- Estimate Program Size: Again using data on prior projects, PROBE shows you how to estimate total program size.
- Estimate Resources: Finally, using historical data, you estimate the hours of work required for the job.
- Produce the Schedule: From the estimated hours, you next produce a schedule for the work.

In this series, I will walk you through all these steps and show how to gather and use historical data to make good plans. I will also show how to determine in advance the likely accuracy of your plan.

The size estimating problem

The principal problem with size estimating is that you generally need estimates and plans at the beginning of a project. This, unfortunately, is when you are least able to make a good estimate. The reason is that, to make a sound plan, you must start with an estimate of the size of the product you intend to build. Then, based on historical productivity data, you can calculate the time the development work will likely take. The problem is that at the beginning of the job, you know the least about the future product.

From my work with PROBE and the PSP, I have found that the principles of object oriented design can be a big help in determining a planned product's size. To find out how to do this, however, you will need to read the next several columns.

An invitation to readers

In these columns, I discuss software process issues and the impact of processes on engineers. I am, however, most interested in addressing the issues you feel are important. So please drop me a note with your comments and suggestions. Depending on the mail volume, I may not be able to answer you directly, but I will read your notes and consider them when I plan future columns.

Thanks for your attention and please stay tuned in.

Watts S. Humphrey
watts@sei.cmu.edu