Estimating With Objects - Part VI

Contents

This column is the sixth in a series about estimating. The first was in the July 1996 issue. This column describes how to get a program size estimate from an estimate of a program's object LOC. The steps involve something called linear regression. This column describes what linear regression is, how it helps in making size estimates, and how to do the linear regression calculations.

The prior columns in this series gave an overview of estimating and defined some of the steps in making a size estimate. If you have not read these earlier columns, you should look at them first to understand the context for this discussion and to see how these various estimating topics relate. To repeat what I said in the previous columns, the estimating method described here is called PROBE. If you want to quickly learn more about PROBE, you should read my book A Discipline for Software Engineering, from Addison Wesley. This book introduces the Personal Software Process (PSP)^{SM}, which is an orderly and defined way for software engineers to do their work.

This column continues the discussion of how to make size estimates. To make a project plan, you need a resource estimate and, to estimate resources, you need to estimate the size of the product you plan to build. Also, to make a good size estimate, you need historical data on the sizes of the programs you have previously written. This and the previous columns describe how to gather these data and how to use them to make the size estimate.

What we have covered so far

To refresh your memory, the estimating steps we have covered so far are as follows.

In part I of this series in July, we discussed the general problem of estimating

In part II in August, we addressed size measures: what they are, the criteria for a good size measure, and how to define a size measure.

In part III in September, we introduced size estimating proxies and explained why objects make good estimating proxies.

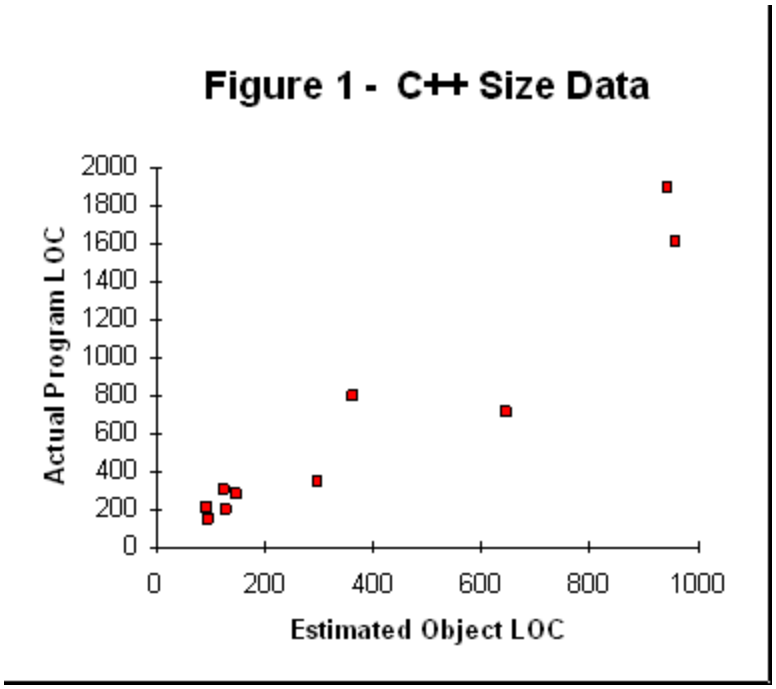
Part IV in October addressed the problem of size data. To make good estimates, you need to compare the new product with products you have previously developed. To do this, you need an orderly way to save and analyze size data on your prior programs.

Part V in November discussed the conceptual design and how to produce one. It is during the conceptual design that you make the object LOC estimate.

At this point, you know how to make a conceptual design for a new program and how to estimate the object LOC in that conceptual design. The next step is to convert this object LOC estimate into a program estimate.

The difference between object LOC and program LOC

Depending on your personal style, you could have most of the program function in the objects or you could use very little object code. When I started writing object-oriented programs, I essentially substituted objects for functions and procedures. This left the function and procedure declarations and the main program routine outside of the object LOC. With more experience, I started to define the main routines as objects, with only a few initializing LOC. My latter programs thus tended to be largely object LOC. Only the headers and other declaration statements were left out of the object LOC. Your personal style will have a significant impact on the relationship between object LOC and program LOC.



From an estimating point of view, it does not matter what your style is, as long as your estimates are based on your personal style. Thus, the historical size data you use should be consistent with the style you intend to use for the program you are estimating.

Some C++ data

Some data on my C++ programs are shown in Figure 1. Here, the estimated object LOC are shown on the x axis at the bottom, and total actual program LOC are shown on the y axis on the left. Note, however, that this is only new and changed LOC. When I reuse prior programs, I do not count them. The reason is that reuse takes considerably less effort than developing new code. This issue will be discussed in more detail next month.

From Figure 1 it is apparent that if you knew how many object LOC would be in the new program, you could make a pretty informed estimate of the program LOC. The reason is that there is a reasonably stable relationship between them. You should, however, check this relationship by computing the correlation between the program LOC and the object LOC. If the correlation is significant and if the $r^2 > 0.5$, then object LOC are a good predictor of program LOC. In the case of the data in Figure 1, the correlation gives an r^2 of 0.911, so the relationship is good enough for estimating purposes.

Linear regression

The way you estimate program LOC from object LOC is to determine the relationship between these two measures. The relationship suggested by Figure 1 is shown by the line in Figure 2. Here, we postulate a relationship between x and y that is defined by the equation

$$y = \beta_0 + \beta_1 x$$

This is called the regression equation. We call it linear regression because the equation is linear in the β terms. Thus, even $y = \beta_0 + \beta_1 x^2$ or $y = \beta_0 + \beta_1 e^x$ would still be linear regression. You should use whichever equation type gives the best fit to your data. For estimating purposes, however, we will stick with linear relationships in x and y as well as in the β's. Once you have the regression equation, you can estimate the object LOC and then calculate the program LOC.

The linear regression equation actually compensates for two relationships. We have already discussed one: the percent of the program LOC that is object LOC. The other relationship concerns the accuracy of your size estimates. If, for example, you consistently estimated low, the linear regression equation would adjust for this.

Calculating the regression line

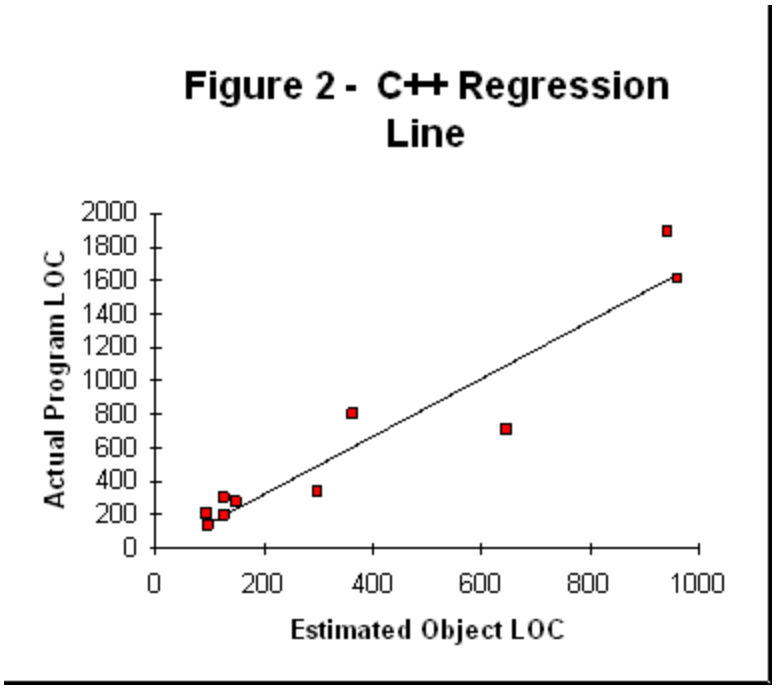
The regression line is called the least-squares fit to the data. That is, the data have minimum variance from the regression line. If you summed the squares of the distances of all the points from the regression line, you would find that you have a minimum value. That is, there is no other straight line that will give a lower variance of the data from the line.

The values of β_0 and β_1 can be calculated from the data with the following formulae:

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n} x_i y_i - n x_{avg} y_{avg}}{\sum_{i=1}^{n} x_i^2 - n\left(x_{avg}\right)^2}$$

$$\hat{\beta}_0 = y_{avg} - \hat{\beta}_1 x_{avg}$$

$$\beta_0 = y_{avg} - \beta_1 x_{avg}$$



An example regression calculation

With the 10 data values given in the table, the regression calculations are done as follows:

$$\beta_1 = \frac{4,303,108 - 10 * 382.8 * 638.9}{2,540,284 - 10 * 382.8 * 382.8} = 1.7279$$

$$\beta_0 = 638.9 - 1.7279 * 382.8 = -22.55$$

What this means is that, for these data,

Program LOC = -22.5 + 1.7279*(Object LOC)

Program Number i	Estimated Object LOC x_i	Total LOC y_i	$x_i y_i$	x_i^2
1	130	186	24,180	16,900
2	650	699	454,350	422,500
3	99	132	13,068	9,801
4	150	272	40,800	22,500
5	128	291	37,248	16,384
6	302	331	99,962	91,204
7	95	199	18,905	9,025
8	945	1890	1,786,050	893,025
9	368	788	289,984	135,424
10	961	1601	1,538,561	923,521
Sum	3828	6389	4,303,108	2,540,284
Average	382.8	638.9		

Now you can use the regression line and your prior program data to estimate the size of a new program. This assumes that you have followed the guidelines described in the August column, that you have properly saved your prior project data, and that you have estimated the Object LOC as described in the November issue.

Estimating development time

The regression equation can also be used in exactly the same way to estimate development time. The only difference now is that you use the following formula:

Development time = $\beta_0 + \beta_1$(Object LOC)

Of course these βs are different from the ones calculated above. To determine them, you need another column of data for the total development time in the above table. Then you can use the previous formulas to calculate the new βs and plug in your object LOC estimate.

Next month's topics

You now know how to make size and resource estimates for a new program. The next month's columns address the various categories of program size, how to determine estimate accuracy, and how to make schedules.

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 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