Project Cost Adjustments
This article describes how to make adjustments to a cost estimate for environmental factors, schedule strategies and software reuse.
Author: William Roetzheim
Co-Founder, Cost Xpert Group, Inc.

Project Cost Adjustments

Research by Capers Jones has indicated that formal software cost estimating techniques can roughly double the probability of your software project being completed successfully, but few companies and individuals really understand that software estimating can be a *science*, not just an *art*. It really is possible to accurately and consistently estimate development costs and schedules for a wide range of projects. This series of 4 articles provides you the tools you need to understand step-by-step approaches to estimating the cost and schedule for your projects. Although there is a wide range of software cost estimating tools on the market to help with this process, we focus in this series of articles on understanding the fundamental concepts. You will be able to implement the concepts in these articles using nothing more complicated than a spreadsheet.

In our first article, estimating software costs, we covered the various methods of estimating the size of a program (called *program volume*). We discussed the traditional measures of Lines of Code and Function Points, plus introduced other approaches. At the end of this article, we showed you how to prepare a preliminary, unadjusted estimate using this information.

In this article, *Project Cost Adjustments,* we cover the concept of project cost adjustments for variations in the project environment. At the end of this article, you will be able to create an accurate estimate of the time and cost required to develop a new application.

Next month's article , *Dealing with Reuse,* explains how to quantify the impact of software reuse and commercial components/libraries on your estimate.

Finally, Article 4, *Creating the Project Plan* describes how to use your insight into project cost and schedule to create a complete project plan.

Adjusting for the Environment

Obviously, the size of the development effort (the program volume) will have a significant impact on the project cost. In addition, your intuition probably tells you those additional factors, such as the experience and capabilities of the development team, also have a significant impact. In fact, the impact of factors such as these can often result in fivefold or tenfold variations in development

Cost Xpert Group, Inc.
www.costxpert.com
2990 Jamacha Rd, Suite 250
El Cajon, California 92019
Phone: 619.670.6168
Fax: 619.670.3230

efficiency, and hence cost. Luckily, there is a quantitative, methodical approach to adjusting for these project- specific environmental factors.

Value of Environmental Adjustments

Project environmental factors can have a major influence over the development efficiency of your team. Factors are rated using a scale of Very Low, Low, Normal, High, Very High or Extra High.

Recall from last month that a project's cost is adjusted up or down based on inefficiencies associated with increasing project size. This is handled using an exponent factor that acts on the total SLOC value. Nonlinear environmental factors modify this inefficiency adjustment factor up or down.

Factor	Very Low	Nominal	Extra High
Architecture/ Risk Resolution	.0423	.014	-.0284
Development Flexibility	.0223	.002	-.0284
Precedentedness	.0336	.0088	-.0284
Process Maturity	.0496	0.0184	-.0284
Team Cohesiveness	.0264	.0045	-.0284

The correct value for all factors is summed, then added to the unadjusted factor as presented last month.

Let's take a simple example. In our previous installment we worked out the effort to complete an e-commerce project consisting of 450 function points as follows:

Effort = Productivity * KSLOC $^{\text{Penalty}}$ = 3.08 * 20.7 $^{1.030}$ = 3.08 * 22.67 = 70 Person Month

The default project size inefficiency adjustment was 1.030 for e-commerce projects. Suppose that this project was nominal for architecture/risk resolution, development flexibility, and precedentedness but was very low for organizational process maturity and very low for team cohesiveness.

The new size inefficiency adjustment factor would now be:

$$1.030 + .014 + .002 + .0088 + .0496 + .0264 = 1.1308$$

Cost Xpert Group, Inc.
www.costxpert.com
2990 Jamacha Rd, Suite 250
El Cajon, California 92019
Phone: 619.670.6168
Fax: 619.670.3230

The new calculated effort would now be:

Effort = Productivity * KSLOC ^{Penalty} = 3.08 * 20.7 $^{1.1308}$ = 3.08 * 30.68 = 94.8 Person Month

There are also linear project environmental adjustment factors. Linear factors adjust a project's effort up or down in a linear fashion. For example:

- A value of 1.0 for an environmental factor means that the default equations should apply.
- A value of 1.5 means that the default calculated cost increases by 50%.
- A value of 0.5 means that the default calculated cost decreases by 50%.

When using more than one linear environmental variable, the total adjustment is determined by multiplying the adjustment contribution by each variable together.

The following table illustrates the magnitude of impact you may observe from some the various linear environmental factors.

Factor	Very Low	Nominal	Extra High
Analyst capability	1.42	1.00	0.71
Applications Experience	1.220	1.00	0.810
Language and Tool experience	1.20	1.00	0.840
Personnel Continuity	1.29	1.00	0.81
Management capability	1.18	1.00	0.87
Management experience	1.11	1.00	0.90
Platform Experience	1.19	1.00	0.85
Programmer capability	1.34	1.00	0.76
Execution time constraint	1.00	1.00	1.63
Main storage constraint	1.00	1.00	1.46
Platform Volatility	0.870	1.00	1.300
Effective management tools	1.22	1.00	0.84
Multisite Development	1.22	1.00	0.80
Office ergonomics	1.19	1.00	0.82

Cost Xpert Group, Inc.
www.costxpert.com
2990 Jamacha Rd, Suite 250
El Cajon, California 92019
Phone: 619.670.6168
Fax: 619.670.3230

Factor	Very Low	Nominal	Extra High
Use of S/W Tools	1.17	1.00	0.78
Database size	0.90	1.00	1.28
Documentation Match to Lifecycle	0.81	1.00	1.23
Internationalization	0.97	1.00	1.35
Product complexity	0.750	1.00	1.660
Required Reusability	0.95	1.00	1.24
Required Software Reliability	0.82	1.00	1.26
Graphics and multimedia	0.95	1.00	1.35
Legacy integration	1.00	1.00	1.18
Site security	0.92	1.00	1.40
Text content	0.94	1.00	1.16
Tool selection	0.95	1.00	1.14
Transaction loads	0.96	1.00	1.59
Web strategy	0.88	1.00	1.45

TABLE 1: IMPACT FROM ENVIRONMENTAL FACTORSRS

The numbers in the preceding table were empirically determined from extensive work by Barry Boehm, Capers Jones, and others. They have been validated against over 20,000 projects.

Let's take a simple example. Our current effort estimate is 94.8 person months of effort to deliver our 450-function point e-commerce system. Suppose that this project was nominal in every way, except that our Programmer capability and analyst capability were both extremely high. How much effort would be required to implement this project?

We start by noticing that analyst capability adjustment will be 0.71 and the programmer capability adjustment will be 0.76. The total effort can then be calculated as:

94.8 Person-Months * 0.71 * 0.76 = 51.2 Person-Months

Cost Xpert Group, Inc.
www.costxpert.com
2990 Jamacha Rd, Suite 250
El Cajon, California 92019
Phone: 619.670.6168
Fax: 619.670.3230

So far, we have described how to define the project volume and how to adjust for the project development environment. The result will be an estimated effort. Now, let's look at how we convert this effort into an optimal schedule.

The approach is actually very similar to the approach we followed to arrive at the estimated effort. For this calculation, we start with the estimated effort in terms of person months. We raise that to a power, then multiply the result times a linear factor. Although you can get relatively sophisticated when selecting or calculating the correct values for the linear and non-linear parameter, a reasonable approximately is to take the cube root of the effort (raise effort to 0.33), then to multiply the result times a value from the following table.

Project Type	Schedule Multiple
COCOMO II Default	3.67
Embedded development	4.00
E-Commerce development	3.20
Web development	3.10
Military development	3.80

Continuing with our e-commerce project example, the computed optimal schedule would be:

$3.20 * 51.2^{0.33} = 3.20 * 3.66 = 11.7$ Months

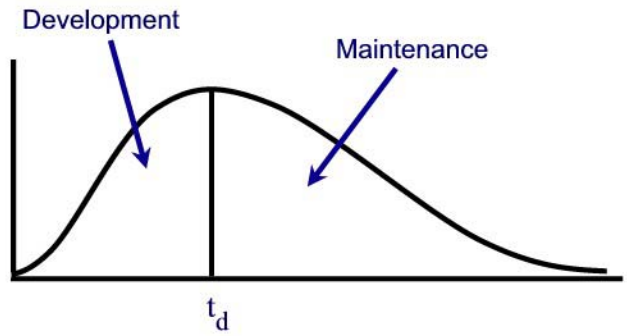
Norden (see reference below) discovered that optimal project staffing for typical projects that require communication and learning follows a Rayleigh distribution. This curve ramps up relatively steeply, then trails off gradually, resulting in a wave shape. Putnam (see reference below) confirmed that this result applied to software development projects. Figure 4 shows the optimal staffing curve for a hypothetical project. This characteristic curve is known as the Putnam-Norden-Rayleigh, or PNR staffing curve.

Note: You might want to take a look at the following outside references:

- *"Useful Tools for Project Management"* in *Management of Production*, Norden, P., , M.K. Starr, Ed., Baltimore, MD; Penguin Books, 1970.
- "A general empirical solution to the macro software sizing and estimation problem" , Putnam, L.; IEEE Transactions on Software Engineering, Vol SE-4, No. 4,(July, 1978), pp. 345-361.

Cost Xpert Group, Inc.
www.costxpert.com
2990 Jamacha Rd, Suite 250
El Cajon, California 92019
Phone: 619.670.6168
Fax: 619.670.3230

This computed optimal schedule includes all project phases through acceptance, shown as t(d) in the following figure. In the case of e-commerce projects this would normally include strategy, business process analysis, requirements, design, implementation, and test. Deployment and maintenance effort is part of the PNR staffing curves, but they fall after the computed delivery time.



$t_d = 2.5\ M^{0.33}$

M = staff months required during development only

FIGURE 1: PNR LIFECYCLE MODEL

Quite often you may be asked to adjust the schedule either to deliver the completed code at an accelerated pace, or to stretch the schedule out to improve efficiency. Any adjustment to the schedule will have a cost impact. Optimal development time, or $T_{d,}$ is the development time that is optimal in terms of a combination of development schedule and required resources. However, there are a variety of reasons why you might want to develop your project using a different schedule.

As shown in the figure, you can reduce costs by increasing development time up to T_o, defined as the optimal development time in terms of cost. T_o has been empirically determined to be twice T_d and to result in a cost reduction of 50%. As you increase the cost beyond T_o, the costs begin to climb again at roughly a linear rate.

Cost Xpert Group, Inc.
www.costxpert.com
2990 Jamacha Rd, Suite 250
El Cajon, California 92019
Phone: 619.670.6168
Fax: 619.670.3230

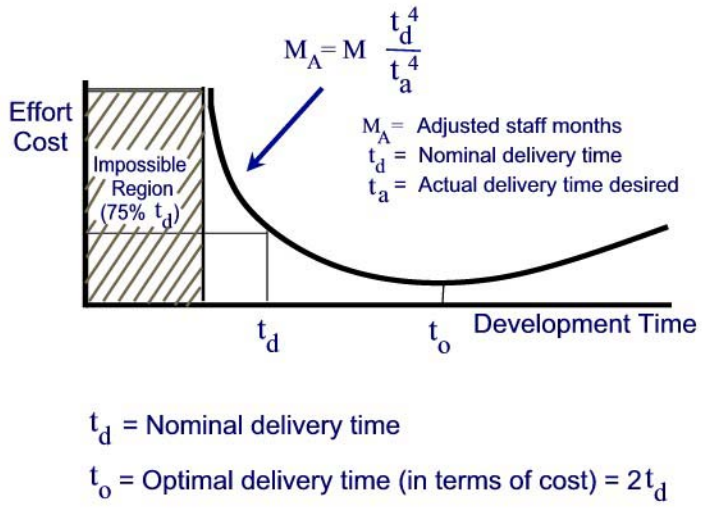


FIGURE 2: REDUCE COSTS, INCREASE DEVELOPMENT TIME

Although it is unusual for a project schedule to be pushed beyond T_d, it is common for software to be required prior to the time allowed by T_d. Accelerating the development schedule is possible, but the cost penalties are severe. As shown in Figure, the costs climb exponentially as the project schedule is accelerated.

Avoiding the Impossible Region

One interesting result of the study of over 750 projects that attempted to deliver the code in less than T_d, was the fact that none of the projects successfully achieved a crashed schedule beyond .75 T_d. Researchers noted this phenomena and this barrier has become knows as the *impossible region* because it is impossible to accelerate the schedule beyond this amount using traditional development approaches.

Alternate Strategies for Accelerating Delivery

There are times when you simply must accelerate the schedule beyond that allowed by the impossible region. When faced with this requirement, there are four basic approaches to follow:

- Reduce Functionality

Cost Xpert Group, Inc.
www.costxpert.com
2990 Jamacha Rd, Suite 250
El Cajon, California 92019
Phone: 619.670.6168
Fax: 619.670.3230

- Decouple Tasks
- Redundant Parallel Development
- Increase Reuse

Reduce Functionality

The best approach is to reduce functionality. In other words, scale back the volume to be created (functionality) to reduce both cost and required schedule.

Decouple Tasks

The cost schedule curves are based on a single project with complex interactions. If you decouple, you can increase parallelism and reduce both schedule and cost. For example, if you can split one project into two that have minimal and well-defined interactions, then both projects can be costed and scheduled independently.

Redundant Parallel Development

True redundant parallel development can significantly decrease your schedule if you have the resources (people and money) available. Basically, you assign multiple teams to write the same component. The first team to finish has their code used. The other code is either discarded or used as a back-up in case of inefficiencies or bugs with the winner's code. This can be repeated for each program module. This approach works because you are exploiting the statistical variances in development time among different independent teams. Unfortunately, it is rare to have adequate skilled resources available to make this approach effect on other than the most critical projects.

Increase Reuse

Increasing reuse can significantly lower your development schedule. This is the true power of component based development environments such as Visual Age Java, Delphi and Visual Basic. Factoring code reuse into your estimate of effort and schedule is the topic of the next article in our series.

Cost Xpert Group, Inc.
www.costxpert.com
2990 Jamacha Rd, Suite 250
El Cajon, California 92019
Phone: 619.670.6168
Fax: 619.670.3230