



## Why can't we estimate better?

[www.davidconsultinggroup.com](http://www.davidconsultinggroup.com)

by  
**David Herron, DCG Founder**  
**Sheila Dennis, VP Sizing and Software Measurement**

### ARTICLE

Running a profitable business is, at heart, an apparently simple process of buying goods and services at a known cost then selling them at a higher price. Of course, the difference between success and failure is dependent on how well you manage the difference between cost and price. Historically, predictable costs have been a problem where the goods or services include software.

Software estimating has been an ongoing problem for programmers, project managers, and senior level IT managers. Like most issues in IT, there are a number of perspectives on the topic of estimating. Most organizations consider their estimating practices to be ineffective and they have no real sense of how to make it better. However, if an IT organization is serious about improving their estimating practices and they want to estimate more effectively, there are solutions available.

### Different Perspectives

When it comes to IT organizations and their estimating practices there are different views on how they assess the effectiveness and importance of estimating. Generally speaking there are three perspectives that IT organizations have with regard to estimating.

#### Estimating isn't a problem

*First we have the organization that doesn't view estimating as a problem or has deemed it an insoluble problem. For example - During the development lifecycle milestones are monitored for schedule and budget compliance. Often times when there are slippages in the schedule or cost overruns the immediate issues are addressed and new milestone schedules are created. Upon delivery of the software, if it is significantly late or over budget then a post-implementation review is conducted. Experience shows that at this point in the lifecycle contributing factors can be numerous: an unrealistic schedule, ambiguous user requirements, the availability of appropriate resources, etc. However, in this scenario, we seldom hear anyone identifying the organization's inability to estimate properly as being one of the core problems to missed schedules and cost overruns. Estimating simply isn't considered among the various problems attributed to poor delivery of software. Since the first step towards improvement is to recognize there is a problem, there is a level of awareness that much be reached before this type of organizational environment can progress.*

#### I want it delivered now

*This dynamic shows itself, not so subtly, when management doesn't really want an estimate at all; they want the software delivered when they want it delivered.* How many times have we seen a situation where the sales/marketing group, the business users or even our own senior management has requested a software solution

Copyright © 2007 David Consulting Group  
1770 E. Lancaster Ave, Suite 15  
Paoli, PA 19301  
+1.610.644.2856 (F) +1.866.293.0120



that has a fixed delivery date already attached to it? And even though the user or senior manager may ask for an estimate they really aren't interested in the response unless they are told what they want to hear. In this type of management environment, the IT organization doesn't invest much time in their estimating practice because they don't realize the power of good estimation as a vehicle to properly manage the project and/or their customer's expectations.

### **We don't have time to get the estimate right**

*This third perspective involves an organization that wants to improve its estimating capability but is unwilling to make the resource investment to make necessary improvements.* The organization may understand the value of properly estimating the project deliverables, and they may even understand at some level what it would take to do it right, but they simply don't want to make the investment necessary to achieve a higher level of estimating accuracy. Their perception is that they don't have the time or resources to get the estimate process right, and yet they end up taking the time and resources to correct the problems resulting (in part) from a lack of properly estimating. Performing a cost analysis may well prove to themselves that the investment to improve their estimating model is well worth the effort.

### **Positioning the Process of Estimating**

All of the above perspectives represent different realities and, at some level, are understandable. The need to get the software out the door and into the hands of the customer is a very real demand of the business. Not wanting to invest the time to get the estimate right is a bit shortsighted; but the organization is most likely not aware of the cost of poor estimating. Or, the IT organization is aware that their inability to properly estimate is a problem, but the issue may be one of many problems they face on a daily basis; and they may not be aware or have confidence that a solution is possible.

The process of estimating should be viewed as a means to managing customer and management's expectation not a black box magic process from which the perfect (or absolute) answer appears. For example, an airline estimates time of arrival for flights, or a building contractor provides an estimate for building repairs. Seldom is it expected that these estimates represent the exact final outcome. So too with software, an estimate is just that – an estimate.

Estimating is a disciplined process that requires quantitative data and qualitative knowledge with regard to the expected outcomes. We need to reframe our thinking about estimating and view it as a vehicle to manage expectations based on best available information at that point in time. If a project ends up being late because the user changed the scope of work, or the project manager is called off to work on another project, there is no way to produce an initial estimate that would have considered those unforeseen delays. However, once a change has been introduced it is perfectly reasonable, indeed essential, to re-estimate and to set expectations anew.

### The Estimating Model

The basic components of an estimating model are well defined and easy to understand. A software project estimating model is comprised of three components. In order to achieve a reasonable estimate, the model needs to solve for (1) the size of the problem, (2) the complexity of the problem and (3) the capacity (ability) of the software development team to design, develop and deploy a satisfactory solution.



Within each of the three components there are a number of variables that are analyzed in order to create a reasonable value for that component. Additionally, the interrelationship among the components needs to be considered. The resulting estimating model can therefore become highly complex.

Each of these three components is required for any project that needs an estimate. Without a clear understanding of each of these pieces and how they fit together, any attempt at creating a reasonable and responsible estimate will fall short.

Solving for this level of modeling complexity can be a barrier for some IT organizations. It requires either an investment in a commercially available software estimating tool or an investment in developing the internal experience base necessary to compute a reasonable result.

### Sizing the Problem

The most effective sizing technique used today for software is Function Point Analysis. Function Point Analysis (FPA) is an industry accepted sizing technique and has been adopted worldwide. The methodology is supported by a user group, The International Function Point Users Group (IFPUG), which maintains the defined FPA methodology, supports the current counting practices and certifies professional counters. The key advantages of FPA are: statistically demonstrable repeatability, availability of expertise, speed of implementation, and portability across multiple platforms and languages.

The Function Point method is dependent upon the identification of five elements; inputs, outputs, inquiries, internal stores of data and external references to data. The definition of these elements is logical and therefore aligned to the users requirements. The next step in the methodology requires a detailed examination each of the individual elements to determine a 'true value' of its size. It is this step in the process that can become time consuming and depending upon where you are in the development lifecycle it can be difficult to obtain all the necessary information required to properly size each individual element.

Other valid sizing techniques have been developed for particular circumstances. With good, disciplined design and implementation these can work effectively within the limited domain for which they were designed. However, this path takes longer to design, implement and normalize than the "ready cooked" FPA.

## **Complexity and Capability**

The complexity of a software problem is varied. The various estimating techniques and tools on the market today have a wide variety of complexity definitions. These may require analysis of such variables as logical and mathematical algorithms, data relationships, reusability, memory and performance requirements, code structures, etc. These variables and many other risk factors are certainly important and will affect the outcomes of your software solution. The difficulty is in determining what elements to evaluate and having a proper method for evaluating the selected elements.

IT organizations that are effectively estimating their software projects are using either a commercial software estimating tool or they have historical data that they use to develop an accurate set of algorithms to compute a complexity value. Alternatively, you can develop a simplistic complexity evaluation method whereby you evaluate an appropriate list of complexity factors and assign an overall complexity measure of low, medium or high. Associated with each of the three designations would be a variable that you would apply to your estimating model. By doing this you have accomplished two things – you have raised the level of sensitivity regarding the complexity level in the software problem domain, and secondly, you are using a consistent method to create an estimate based upon the experience gained from actual outcomes. In addition, you can adjust the complexity factor over time as project characteristics or development environments change.

Solving for capability also requires the use of an automated tool or access to an internal or external data base of information regarding performance levels based on a variety of performance factors. These factors include data relating to the processes being used, the skill levels of the resources, tools and techniques available to the developers, etc. Such data bases are available commercially and are usually aligned by industry types. The most effective approach is to develop your own historical baseline of performance using available tools and techniques.

Either alternative for calculating capacity (tool vs. baseline) has its pros and cons. A commercial estimating tool can work very effectively but will need to be calibrated to fit the behaviors of the organization. These adjustments take time. Similarly, developing your baseline of performance will prove to be more accurate in the long run, but you do need time to gather and analyze the data.

## **Exploring the Options**

If an investment in time and resources is one of the barriers for the IT organization to move forward with the development of a more effective estimating practice, then there are several alternatives that they should consider.

The sizing of a project or application can be difficult and time consuming particularly in the early stages of development. A variance on the detailed method of function point sizing noted above is the FP Lite method. The FP Lite methodology utilizes the same definitions for identifying the five elements; however, it does not require a detailed examination of each element and thereby significantly reduces the time and increases the ability to obtain a relative size value. Studies

show that there is some trade-off in accuracy; however, it is not considered to be significant in light of the value gained by quickly calculating a size value\*.

As noted earlier, complexity and capability variables can be obtained by creating a baseline of performance where by quantitative and qualitative data is collected, analyzed and built into the estimating model. By examining recently completed projects over a period of 6 to 12 months, the proper data points of complexity and development team performance can be collected and analyzed. The knowledge gained from this experience will allow for the creation of relatively simple algorithms that make use of past performance levels and simply apply a rate of delivery depending upon the size of the new project and the matching profile of performance. Creating an internal baseline of performance is often best performed by a consultant who has experience with selecting the proper variables and developing the delivery rates appropriate for your industry and technical environment.

A secondary benefit of baselining qualitative data is the discovery of interdependent process weaknesses that influence the estimation modeling (e.g. requirements management). This discovery enables the organization to actively address the weaknesses in order to strengthen the foundation of the estimation process framework.

### Estimating as a Best Practice

To further support the benefit or the 'correctness' of the options expressed above we need only look to the Software Engineering Institute (SEI) requirements for good estimating to underscore the value of these options. The SEI lists the requirements for good estimating to involve the following:

- An historical database
- Structured processes for estimating product size and reuse
- Mechanisms for extrapolating benchmark characteristics of past projects
- Audit trails
- Integrity in dealing with dictated costs and schedules
- Data collection and feedback processes to foster correct data interpretation

Prior to establishing good estimating practices an IT department may have gone about their business of estimating by trusting the project manager's 'gut feel' for how long the process would take, or the project team would burn extra unaccounted hours trying to fit the actual work load to the originally estimated work load. Once best practices for estimating are established, project managers reference historical data points for similar project types and then calculate estimates based on known parameters and statistical calculations of risk. Actuals are recorded and stored for use in future estimates.

### Conclusion

In summary, being able to estimate more efficiently and effectively is both possible and practical. The following key points will guide you to a successful outcome.

- We need to recognize estimation as a problem and a potentially costly problem at that. Not until we fully understand that improper estimating is a potential barrier to efficiency will we be able to consistently and successfully deliver software.



david consulting group

## Why can't we estimate better?

[www.davidconsultinggroup.com](http://www.davidconsultinggroup.com)

- The way we think about estimating should be reframed in the context of managing expectations based upon the best information available at the time. Estimating is not a crystal ball used to predict the future.
- We can no longer afford to compromise on what we need regarding the input components that make up a successful estimating model. It will require some investment of time and resources but the payback will be well worth the investment.
- And above all, don't overly complicate your estimating model. Adopt practices such as FP Lite to generate size information that is statistically accurate enough for the job of early estimating. Collect the baseline data you need and compute your own internal delivery rates of performance.

\*"FP Lite™ – An Alternative Approach to Sizing"; David Herron and Sheila P. Dennis, 2006.

For further information on this topic or to talk to a DCG expert, contact us at 610.644.2856 or send inquiry e-mail to [info@davidconsultinggroup.com](mailto:info@davidconsultinggroup.com)

Copyright © 2007 David Consulting Group  
1770 E. Lancaster Ave, Suite 15  
Paoli, PA 19301  
+1.610.644.2856 (F) +1.866.293.0120