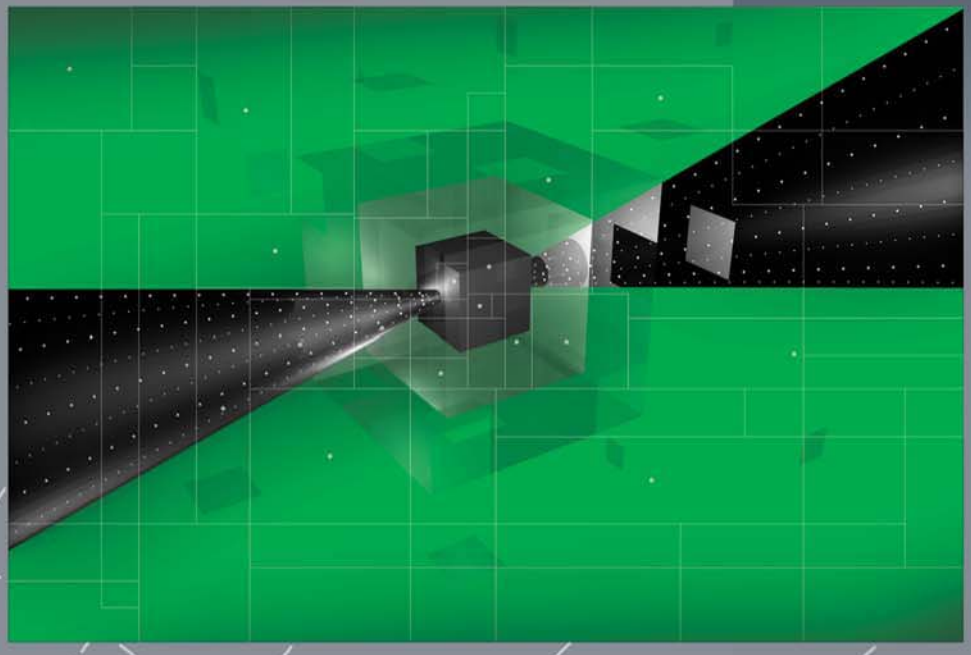
Microsoft

BEST PRACTICES

SOFTWARE ESTIMATION



Demystifying the Black Ar t

Steve McConnell

Two-time winner of *Software Development* magazine's Jolt Award

PUBLISHED BY
Microsoft Press
A Division of Microsoft Corporation
One Microsoft Way
Redmond, Washington 98052-6399

Library of Congress Control Number 2005936847

ISBN: 978-0-7356-0535-0

Printed and bound in the United States of America.

7 8 9 10 11 12 13 14 15 QGT 7 6 5 4 3 2

Distributed in Canada by H.B. Fenn and Company Ltd..

A CIP catalogue record for this book is available from the British Library

Microsoft Press books are available through booksellers and distributors worldwide. For further information about international editions, contact your local Microsoft Corporation office or contact Microsoft Press International directly at fax (425) 936-7329. Visit our Web site at www.microsoft.com/mspress. Send comments to mspinput@microsoft.com.

Acquisitions Editor: Ben Ryan
Project Editor: Devon Musgrave
Copy Editor: Becka McKay
Indexer: Seth Maislin

Body Part No. X11-82276

[2012-03-30]

Chapter 6

Introduction to Estimation Techniques

Chapters 1 through 5 described the critical concepts that underlie all software estimates. This book now turns to a discussion of detailed estimation techniques that can be applied to specific estimation problems.

One important consideration in the use of these techniques is that different techniques will apply in different circumstances. This chapter introduces the major considerations in choosing which techniques to choose.

6.1 Considerations in Choosing Estimation Techniques

The most useful techniques in any given situation are determined by both a desire to account for the estimation influences described in Chapter 5, "Estimate Influences," and a desire to avoid the sources of estimation error described in Chapter 4, "Where Does Estimation Error Come From?" The following sections describe the major issues you should consider.

What's Being Estimated

Some projects determine their features and then focus on estimating the schedule and effort needed to deliver those features. Other projects determine their budgets and development time frames and then focus on estimating how many features they can deliver.

Many estimation techniques are applicable regardless of what is being estimated; a few techniques are better suited to estimating how much effort a project will require, how long a project will take, or how many features can be delivered.

In this book, estimating *size* refers to estimating the scope of technical work of a given feature set—in units such as lines of code, function points, stories, or some other measure. Estimating *features* refers to estimating how many features can be delivered within schedule and budget constraints. These terms are not industry standards; I'm defining them here for the sake of clarity.

Project Size

Project size is another factor to consider in choosing the best estimation technique.

Small I characterize a small project as a project with five or fewer total technical staff, but that is a loose characterization. Small projects typically can't use the statistically oriented techniques that larger projects can use because variations in individual productivity drown out other factors.

Small projects are more likely to use a flat staffing model (using the same number of people on the team for the entire project), which invalidates some of the more algorithmically oriented large-project estimation approaches.

The best estimation techniques for small projects tend to be "bottom-up" techniques based on estimates created by the individuals who will actually do the work.

Large A large project is a project with a team of approximately 25 people or more that lasts 6 to 12 months or more.

The best techniques for large projects change significantly from the beginning of the project to the end. In the early stages, the best estimation approaches tend to be "top-down" techniques based on algorithms and statistics. These are valid at the point in the project when specific team members are not yet known—when plans are based on a team that consists of, for example, "11 senior engineers, 25 staff developers, and 8 testers," rather than specific individuals.

In the middle stages, a combination of top-down and bottom-up techniques based on the project's own historical data will produce the most accurate estimates. In the later stages of large projects, bottom-up techniques will provide the most accurate estimates.

Medium Medium-size projects consist of approximately 5 to 25 people and last 3 to 12 months. They have the advantage of being able to use virtually all the estimation techniques that large projects can use and several of the small-project techniques, too.

Software Development Style

For purposes of estimation, the two major development styles are *sequential* and *iterative*. Industry terminology surrounding iterative, Agile, and sequential projects can be confusing. For this book's purposes, the primary difference between these kinds of projects is the percentage of requirements they define early in the project compared to the percentage they define after construction is underway.

Here is how several common development approaches stack up according to these criteria.

Evolutionary prototyping Evolutionary prototyping is used when requirements are unknown, and one of the primary reasons to use evolutionary prototyping is to help define requirements (McConnell 1996). For estimation purposes, this is an iterative development style.

Extreme Programming Extreme Programming deliberately defines only the requirements that will be developed in the next iteration, which typically lasts less than a month (Beck 2004). For estimation purposes, Extreme Programming is a highly iterative approach.

Evolutionary delivery An evolutionary delivery project can define anywhere from "hardly any" to "most" of its requirements up front (Gilb 1988, McConnell 1996). Depending on which end of the scale the project falls on, an evolutionary delivery project can be either sequential or iterative. Most evolutionary delivery projects leave enough requirements undefined at the beginning of construction that the approach as normally practiced is iterative.

Staged delivery Staged delivery attempts to define the majority of its requirements prior to beginning the majority of construction (McConnell 1996). It uses iterations within design, construction, and test, so in some sense it is iterative. For estimation purposes, however, it is a sequential development style.

Rational Unified Process The Rational Unified Process (RUP) describes its stages as "iterations." However, a nominal RUP project seeks to define about 80% of its requirements before construction begins (Jacobson, Booch, and Rumbaugh 1999). For estimation purposes, RUP is a sequential development style.

Scrum Scrum is a development style in which a project team takes on a set of features that it can implement within a 30-day "sprint" (Schwaber and Beedle 2002). Once a sprint begins, the customer is not allowed to change requirements. From the point of view of an individual sprint, for estimation purposes, Scrum is sequential. Because features are not allocated for more than one sprint at a time, from a multiple-sprint (multiple iteration) point of view, Scrum is iterative.

Effect of Development Style on Choice of Estimation Techniques

Both iterative and sequential projects tend to start with top-down or statistically based estimation techniques and both eventually migrate toward bottom-up techniques. Iterative projects transition to refining their estimates more quickly using project-specific data.

Development Stage

As a team works its way through a project, it develops information that supports more accurate estimates. Requirements become better understood, designs become more detailed, plans become firmer, and the project itself generates productivity data that can be used to estimate the remainder of the project.

This book defines development stages as follows:

Early On sequential projects, the early stage will be the period from the beginning of the project concept until requirements have been mostly defined. On iterative projects, *early* refers to the initial planning period.

Middle The middle stage is the time between initial planning and early construction. On a sequential project, this time will extend from requirements and architecture time until enough construction has been completed to generate project productivity data that can be used for estimation. On iterative projects, *middle* refers to the first two to four iterations—the iterations that occur before the project can confidently base its estimates on its own productivity data.

Late *Late* refers to the time from mid-construction through release.

Some techniques work best in the wide part of the Cone of Uncertainty. Others work better after the project has begun to generate data that can be used to estimate the remainder of the project.

Accuracy Possible

The accuracy of a technique is a function partly of the technique, partly of whether the technique is being applied to a suitable estimation problem, and partly of when in the project the technique is applied.

Some estimation techniques produce high accuracy but at high cost. Others produce lower accuracy, but at lower cost. Normally you'll want to use the most accurate techniques available, but depending on the stage of the project and how much accuracy is possible at that point in the Cone of Uncertainty, a low-cost, low-accuracy approach can be appropriate.

6.2 Technique Applicability Tables

Most of the remaining chapters in this book begin with tables that describe the applicability of techniques in the chapter. Here's an example:

Applicability of Techniques in this Chapter—SAMPLE

	Group Reviews	Calibration with Project-Specific Data
What's Estimated	Size, Effort, Schedule, Features	Size, Effort, Schedule, Features
Size of project	- M L	S M L
Development Stage	Early–Middle	Middle—Late
Iterative or Sequential	Both	Both
Accuracy Possible	Medium–High	High

The entries in the table are based on the considerations described in the previous section. Table 6-1 describes how to interpret the entries in these tables.

Table 6-1 Possible Entries in the "Applicability of Techniques in This Chapter" Tables

Table Entry	Possible Entries
What's estimated	Size, Effort, Schedule, Features
Size of project	S M L (Small, Medium, Large)
Development stage	Early, Middle, Late
Iterative or sequential	Iterative, Sequential, or Both
Accuracy possible	Low, Medium, High

Tip #29

When choosing estimation techniques, consider what you want to estimate, the size of the project, the development stage, the project's development style, and what accuracy you need.