# Reducing Bias in Software Project Estimates

*"It's tough to make predictions, especially about the future."* – **Yogi Berra**

Nearly every software development estimate has been, or will be, biased. Biases in the estimating process contribute to poor estimates, which can affect the success or failure of a project. To understand the psychological impact of bias in developing software project task level effort estimates is essential for information technology Project Managers and their teams. The key questions become:

1. How do biases affect bottom-up task level effort estimates for software development?
2. What bias-reduction strategies can you employ to improve the quality of your estimates?

In spite of impressive advances in processes and tools, software project estimating remains more of an art than a science. Software projects continue to finish behind schedule and over budget, if they finish at all. According to a recent study, only 37 percent of software projects are completed on time and only 42 percent are completed within budget [1]. This is due in part to the difficulties in acquiring accurate estimates of software development effort.

"The subject of software estimating is definitely a black art," says Lew Ireland, former president of the Project Management Institute. Furthermore, understanding the role of judgment and bias in software estimating is even more elusive. An extensive literature search yielded virtually no research and few articles dealing with the topic. Therefore, we applied Amos Tversky and Daniel Kannamen's seminal research done in the areas of judgment and bias to the topic of software project estimating [2].

## The Judgement-Bias Curve

One of the most widely used methods of estimating software development tasks from a bottom-up or task-level approach is expert judgment, sometimes known as a "best guess" [3]. Expert judgment, although a very valuable method, is subject to human biases. Biases are more pronounced in the development of bottom-up estimates because expert judgment, by its very nature, is a very subjective estimating method. The estimates are most often developed through the use of best guesses due to the relative immaturity of software development as an engineering discipline. In many cases, the expert judgment estimate is produced by team members experienced in the work at hand, but not necessarily experienced in estimating techniques.

When estimating, our judgment has a fairly large degree of uncertainty associated with it. By incorporating the bias-reduction techniques outlined in this paper you can increase the level of certainty inherent within your estimates, move down the judgment-bias curve, and ultimately improve the quality of your estimates (See Figure 1.). Software estimating is more of an art than a science, and inherently more prone to the negative aspects of human biases.

## Description of Bias, Example, and Bias-Reduction Strategies

The mental shortcuts, or heuristics, we use to solve complicated, uncertain problems, like estimating software development work effort, are subject to biases. A bias is a partiality, or prejudice, underlying the decision-making process (the bias emanates from the heuristic) [4]. Some biases that have an impact on the development of task-level software project estimates, particularly when derived from expert judgment, are:

Figure 1. *The Judgement-Bias Curve*

Judgement
Bias
Certainty

- Availability bias.
- Representative bias.
- Overconfidence bias.
- Confirmation bias.
- Insufficient anchor-adjustment bias.
- Prudence bias.

### Availability Bias

This reflects our unconscious attempts to predict the future based on memories of the past. The fact that our memory is marked by more vivid or recent experiences allows the availability bias to skew our judgment. A common example is the tendency for individuals to overestimate the occurrence of a more memorable or graphic cause of death, such as a plane crash, rather than a less memorable event, such as a car crash.

Estimates are also subject to our own bounded rationality [5]. The first reasonable number that seems to make sense is often used as the starting point for an initial estimate, which often acts as an anchor (See insufficient anchor adjustment bias). The search for information on which to base this estimate is less than rigorous and often subject to mental shortcuts.

Software estimates based on expert judgment are often derived from estimating by analogy, either formally through the use of historical data or informally from past experience. Many experienced software engineers use completed projects, particularly projects they have worked on in the past, as a heuristic for current software development estimates. The availability bias predicts that information recalled from memory that is used to develop task-level estimates is most likely the very best or the very worst memories of completed tasks or projects, since these experiences would be most readily remembered. Vivid, compelling, or otherwise interesting instances from past projects can bias the estimate for the project or task.

#### *Availability Example*

Consider the case of Alex. He works on your project team, but spends most of his time talking about the Kennedy assassination and the Challenger disaster. He also refers to the last project he worked on as the "best darn project ever done on time and under budget." Alex uses his experience on his last project as an analogy to come up with his task-level estimates for your project, which is good news. The bad news is that the estimate may be biased due to the fact that a cross-section of projects was not used.

### *Bias-Reduction Strategies*

There are tactics you can use to stress-test Alex's estimates:

1. Ask him what assumptions were used, and whether they make sense. Basing the estimate on a similar task completed for a past project where everything (or nothing) went well is a recipe for disaster. Not only should the task be similar, but the project should be, too.
2. Encourage Alex to adjust his initial estimate so it is based on historical data or metrics such as productivity rates and size measures, if available. The objective is to use more than one project as a reference.
3. Discuss Alex's estimate as a team. Groups have been shown to exhibit less of an availability bias than individuals [6].

## Representative Bias

The representative bias is most often expressed in software estimating as insensitivity to base-rate data. We can think of base-rate data as existing metric data from past projects. Individuals may tend to ignore metric data in assessing their estimates when other anecdotal information is available, even if the anecdotal information is irrelevant [7]. The representative bias predicts that even if we have extensive metric data, our teammates may not be inclined to consider it when coming up with their estimates. Under this bias, the estimate is probably constructed using information with a higher degree of uncertainty than would have been the case had we used the existing metric data.

### *Representative Example*

Ralph has just joined your project team from another organization that did not have historical data or metrics. He avoids metrics like the plague and points out that "past experience is a poor indicator of future results." To no one's surprise, Ralph does not use historical project data or other metrics to derive his task-level estimates for your project.

### *Bias-Reduction Strategies*

As the project manager, you are ultimately responsible for the accuracy of Ralph's estimates. To ensure you will not be in a soup line any time soon:

1. Encourage Ralph to use any and all historical data and metric information available. While intuition is good, so are data.
2. At the very least adjust Ralph's estimate in the direction of the historical average. We are more likely to perform closer to the average on subsequent trials. Statisticians call this "regression to the mean."
3. Make sure Ralph does his homework before presenting this estimate to the team. Groups generally show a higher rate of representative bias than individuals. In other words, groups are less likely to use available data and metrics [6].

## Overconfidence Bias

This bias (sometimes referred to as the optimistic bias) demonstrates that we tend to overestimate our abilities and underestimate the effort involved in completing a difficult task, particularly when we are completing the task ourselves. Studies have shown that the more difficult and uncertain a task, the more prevalent the overconfidence bias [7]. In other words, individuals tend to drastically underestimate large, complex tasks when using expert judgment as an estimating method.

### *Overconfidence Example*

Olive is assigned to the coding changes for your high-profile project. When asked about the amount of work involved in completing her tasks, she often prefaces her response with phrases like, "This is a piece of cake," and "No problem." Even you, the project manager, are taken aback by the aggressiveness of Olive's schedule. Ironically, the more difficult the task the quicker she plans to get it done. "No problem" might end up being a big problem.

### *Bias-Reduction Strategies*

There are ways to decrease the risk of having an estimate that reflects reality in only the most fortunate of situations. May luck be your constant companion, but just in case, you can:

1. Encourage Olive to develop a range instead of a point estimate. Make sure she considers the worst case, Murphy's Law-type of scenario for the high-end of the range (a pessimistic estimate).
2. Ask Olive on what assumptions this estimate range is based. If the answer is "We will have two fully dedicated clairvoyants developing the requirements," adjust the estimate upward.
3. Gather information from a variety of sources to get a broader picture of what needs to be done.
4. Tell the team that you are not interested in best-case estimates, but realistic estimates. Some studies have shown this will help; some have shown it will not matter, but it cannot hurt.

**Remember**—studies show our estimates are even more optimistic the more complex and difficult the task [7].

## Confirmation Bias

This is in effect when people search for information that supports their idea and ignore information that does not support their idea. An analyst who develops a task-level estimate may consider information supporting the estimate, and ignore information that the task at hand may be significantly larger or smaller than that initial estimate indicates.

### *Confirmation Example*

Cathy is a perfect example. She tends to stick to her guns, and can be narrow-minded. She tends to start estimating her work effort with an estimate in hand and, after limited research, usually concludes she was right in the first place.

### *Bias-Reduction Strategies*

Cathy can be helped. Here is how:

1. Encourage her to research historical data and metrics, and ask other experienced team members for help. It is important to get her to look at

a variety of information, not just data to support her initial estimate.
2. Play devil's advocate. Question her sources and assumptions.
3. Most importantly, ask if she adjusted her initial estimate based on information she found after her research.

## Insufficient Anchor Adjustment Bias

This occurs when an initial estimate is made (the anchor), and upon re-estimating the effort, insufficient adjustments are made from that anchor. It does not matter if the initial estimate is derived from historical data, parametric modeling tools, or a random number generator.

### *Insufficient Anchor Adjustment Example*

The task of creating a test plan falls to Alice, who likes to get other team members' opinions on how much effort they think the task entails. She hates a blank sheet of paper. Someone estimates the task at 25 effort hours. After assessesing the available information, she determines that 25 hours seems low, and decides to double the estimate to 50 hours based on limited research. Chances are this adjustment is insufficient, simply because it is based on the initial anchor of 25 hours. The task turns out to require 250 effort hours. This is the danger of the anchoring bias.

### *Bias-Reduction Strategies*

There are methods you can employ as a project manager, or conscientious team member, to try and avoid the negative aspects of the anchor bias:
1. Encourage Alice to research the problem and really dig into it.
2. Ask her to specify a range rather than a point when researching the effort required. This will denote the uncertainty involved and reduce the tendency to insufficiently adjust subsequent estimates. Stating the estimate as about 40 to 80 effort hours is less specific and probably easier to adjust, than an early estimate of 52 hours. It pays to be approximately accurate, rather than precisely inaccurate.
3. Do not ask leading questions when inquiring about an estimate. Avoid saying, "Alice, what do you think? Is this about 50 hours?" Or, "Can we get this done by my birthday next week?" Let Alice do her homework and then negotiate.

## Prudence Bias

When faced with high-profile tasks, or the first few times *accountability* is used in the same sentence as *task-level estimates*, team members may respond by coming up with over-cautious estimates [8]. Padding task estimates can be just as dangerous as wildly optimistic low-ball estimates.

### *Prudence Example*

Paul follows all the rules, but you do not want to get behind him on the freeway if you are in a hurry. He takes it pretty slow to be safe, and also pads his task-level estimates to be safe. If several team members follow Paul's lead, the result can be a wildly over-cautious project estimate. Have you heard of of Parkinson's Law? Work expands to fill the amount of time available for completion

### *Bias-Reduction Strategies*

Paul is an asset; he realizes the need to take a closer look at the estimating process. In order to get a more accurate estimate, however, try these techniques:
1. Ask him if he added a cushion or padded his estimate. Pad the project estimate, not each task estimate.
2. Emphasize the need for accurate effort estimates at the task level and show how padding each task will inadvertently lengthen the critical path.
3. Olive the optimist and Paul probably do not have a lot to talk about, but it is a good idea to have them review each other's estimates.

See Table 1 for a summary of the biases that impact software development task-level estimates.

Table 1. *Summary of the Biases and Bias Reduction Strategies*

| Bias | Description | Bias Reduction Strategies |
|---|---|---|
| Availability | Vivid or graphic events overshadow objectivity | • Challenge the assumptions of the estimate.<br>• Use more than one project/task as a reference.<br>• Discuss the estimate as a team. |
| Representative | Not using base rate data or metrics | • Use data as well as intuition.<br>• Adjust estimate toward the mean.<br>• Formulate estimate before discussing as a team. |
| Overconfidence | Too optimistic | • Use an estimate range vs. a point estimate.<br>• Challenge the assumptions.<br>• Use more than one source of information.<br>• Set expectations for realistic estimates. |
| Confirmation | See what you want to see | • Use historical data and metrics.<br>• Play devil's advocate.<br>• Stress the importance of adjusting the estimate. |
| Anchor Adjustment | Insufficient adjustment of subsequent estimates | • Foster a research-based estimate.<br>• Use an estimate range vs. a point estimate.<br>• Do not ask leading questions or throw out a guess. |
| Prudence | Too pessimistic | • Pad the project estimate, not the task estimates.<br>• Discuss the estimate as a team. |

## General Bias-Reduction Strategies

It is virtually impossible to eliminate the impact of human biases on software project estimating. Biases are by definition subconscious. The same psychological mechanism that creates the bias works to conceal it [4].

The first and most important step is awareness that human biases impact decision-making, particularly decisions with uncertainty—like task-level estimating in software development. If the project team can anticipate, identify, and minimize the negative impact of biases in the software estimating process there will be greater certainty in the validity of the project estimate. General strategies will help reduce the human biases in software project estimates. These strategies are:

- Provide feedback on the accuracy of the estimates.
- Collect data to provide rules of thumb.
- Challenge team members to defend and justify their estimates.
- Emphasize the importance of estimating.
- Use more than one estimating method.

### *Provide Feedback on the Accuracy of the Estimates*

Compare the actual effort hours logged on each task to the current estimate. That way the team member knows where he or she is vs. where he or she should be, and can make adjustments accordingly. Do not discount the team member's perception of what work remains or how far along he or she is, but do not rely on that information alone. It is also a good idea to collect data across several completed projects to use as starting points for future estimates. In addition, be sure to collect the original estimate as well as the actual effort required to complete the task [9].

### *Collect Data to Provide Rules of Thumb*

In addition to the original estimate and actual hours logged, other data are useful to provide a history of a project task. At a minimum collect data related to:

- The source of the estimate (best guess, past projects, etc.).
- The activity driver (number of installs, number of requirements, lines of code).
- The assumptions used (especially skill level, resource dedication, requirements volatility).

Let us take Alice's test plan as an example. She should document where she collected the information for the task-level estimate, the activity driver for the task (perhaps the number of test cases), and her assumptions. This data will be very useful the next time she or anyone else develops a task-level estimate for a test plan. Over the course of many projects, it may be found that given this type of project and testing environment, it takes approximately four hours per test case to complete the test plan. If she estimates she will have about 50 test cases, her initial effort estimate might be around 200 hours. Of course the estimate should be adjusted (and perhaps not insufficiently), but the data collected provides an excellent place to start, and a handy rule of thumb. It also provides a repeatable process, which can be improved upon.

### *Challenge Team Members to Defend, Justify their Estimates*

Estimates are based largely on uncertainty. The more information you can find related to the task at hand, the less uncertainty is involved. Question and challenge the estimates, the source of the data, and the assumptions. The adage of garbage in, garbage out applies here.

### *Emphasize the Importance of Estimating*

To paraphrase President Eisenhower, estimates are nothing, estimating is everything. Discourage the path of least resistance or the permanent sacrifice of accuracy for a temporary reduction in effort. Do not just settle for an estimate, but encourage estimating. The real *expert* in the expert judgment approach is homework, not just experience, and the team needs time to do it right.

### *Use More Than One Estimating Method*

Use a variety of estimating methods and sources of information. Use historical data (if you do not have any, start collecting it), industry statistics, estimating tools, organizational metrics data, experienced team members, best guesses, and even intuition. Comparing multiple estimates lets you know if your team is really getting a handle on the project. For example, it is always a good idea to compare the phase-level estimates from a top-down approach, using a parametric modeling tool, to the aggregated task-level estimates from a bottom-up approach. If they are close, you know you are talking apples and apples.

Imagine being dropped off in a remote location. Being lost is a lot like coming up with an estimate. You are not sure where you are, but you have to know before you can figure out where to go. Imagine you reach in your pocket and find a hand-held global positioning system. Things are looking up. You hit a button and find out where you are. However, that estimate of your location is not based on one satellite (a single source of data); it is based on two or three, as your team's estimates should be. Estimating is like putting together the pieces of a puzzle. There is no *answer*, just indicators that need to be analyzed and managed. See Table 2 for a summary of the general bias-reduction strategies and examples.

Table 2. *Summary of General Bias Reduction Strategies*

| General Bias Reduction Strategies | Example |
|---|---|
| Promote awareness | Talk about the impact of biases on estimates |
| Provide feedback on the accuracy of estimates | Track and report estimates to actuals for tasks |
| Collect data to provide "rules of thumb" | Record estimates, actuals, assumptions, and size measures for future reference |
| Challenge team members to justify their estimates | Document and question assumptions and sources |
| Use more than one estimating method | Combine task level estimates and compare to phase estimates |
| Emphasize the importance of estimating | Give team members the opportunity to research their estimates – encourage estimating |

## Conclusion

Human biases influence and generally have a negative impact on the development of task-level estimates. Although it is impossible to obviate these biases, awareness, understanding, and the incorporation of bias-reduction strategies can help mitigate their negative impact.

We have taken a step back to discuss what we feel to be the root cause of poor task-level estimates using the expert judgment approach during bottom-up estimating. The expert judgment method is viable, and likely to remain one of the most popular methods of developing software project estimates for some time. The next step will be determining to what extent these biases impact software project estimates, and where information technology project managers should focus their efforts to reduce the negative consequences of bias in the software estimating process. Our hope is that the suggestions we have provided here can help you and your team develop better task-level software project estimates.

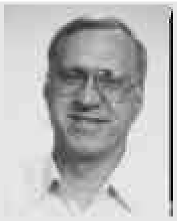
## References

1. Godson, Philip. To Err is Human, to Estimate Divine. *InformationWeek,* January 18 ,1999, pp. 65-72.
2. Kahneman, Daniel, and Tversky, Amos The Framing of Decisions and the Psychology of Choice. *Science,* Vol. 211 No. 30, January 1981, pp. 453-458.
3. Hughes, Robert T. Expert Judgement as an Estimating Method. *Information and Software Technology,* Vol. 38, 1996 pp. 67-75.
4. Jones, Morgan D. *The Thinker's Toolkit,* 1998, Random House, Inc.
5. Simon, Hervert A. *Psychology Review,* Vol. 63, p. 129, 1956.
6 Lim, Lai-Huat., and Benbasat, Izak. A Framework for Addressing Group Judgment Biases with Group Technology. *Journal of Management Information Systems,* Vol. 13, No. 3, Winter 96-97, pp. 7-24.
7. Bazerman, Max. H. *Managerial Decision Making,* 2nd Ed., 1990. John Wiley & Sons, Inc.
8. Hammond, John. S. Keeney, Ralph. L., Raiffa, Howard. The Hidden Traps in Decision Making. *Harvard Business Review,* Sept.-Oct. 1998, pp. 47-58.
9. Demarco, Tom *Controlling Software Projects,* Yourdon Press Computing Series, 1982.

## About the Authors

**David Peeters** is a project manager in the Information Services Division at American Family Insurance in Madison, Wis. He has more than 10 years experience in application development, project planning, and project management. Peeters has a bachelor's degree in computer information systems from Southwest State University and a master's of business administration from Illinois State University.

David Peeters
American Family Insurance
6000 American Parkway
Madison, Wis. 53783
Voice: 608-242-4100 ext. 31348
Fax: 608-243-6558
E-mail: DPeeters@AmFam.com



**George Dewey** is President of Pathfinder Global Group Inc. He is a certified Project Management Professional, with more than 20 years of experience in corporate planning, project scheduling, cost control, estimating, and management with numerous consulting firms and client organizations. Dewey holds a bachelor's degree in industrial engineering from North Dakota State University and a master's of business administration from Virginia Polytechnic Institute.

George Dewey
Pathfinder Global Group Inc.
5358B N. Lovers Lane #113
Milwaukee, Wis. 5225
Voice: 713-827-4481
Fax: 414-464-3005
E-mail: GDewey_PGGI@MSN.com

## About the Authors of Does Calibration Improve Predictive Accuracy?, continued from page 17

**Daniel V. Ferens** is a corporate affordability officer at the Air Force Research Laboratory at Wright-Patterson AFB in Dayton, Ohio. He is also an adjunct associate professor of software systems management at the Air Force Institute of Technology (AFIT), Graduate School of Logistics and Acquisition Management, at Wright-Patterson AFB, where he teaches courses in software estimation and software management in general. He was the advisor for the 10 theses described in this paper, is an active member of the Society of Cost Estimating and Analysis, and a lifetime member of the International Society of Parametric Analysts. Ferens has a master's degree in electrical engineering from Rensselaer Polytechnic Institute, and a master's degree in business from the University of Northern Colorado.

AFRL/IFSD, Bldg 620
2241 Avionics Circle, Room S1Z19
Wright-Patterson AFB, Ohio 45433-7334
Voice: 937-255-4429, ext. 3558
FAX 937-255-4511
E-mail: daniel.ferens@sn.wpafb.af.mil

**David S. Christensen** is an associate professor of accounting at Southern Utah University. He was the reader for the 10 theses described in this paper. He received his doctorate degree in accounting from the University of Nebraska-Lincoln in 1987, and has published more than 50 articles in the area of defense cost management. He taught cost management topics at the Air Force Institute of Technology from 1987-97. He is a member of the American Accounting Association, the Institute of Management Accountants, and the Society of Cost Estimating and Analysis.

Southern Utah University
College of Business
351 West Center St.
Cedar City, Utah 84720
Voice: 435-865-8058
Fax: 435-586-5493
E-mail: ChristensenD@suu.edu