feature

project estimation

Schedule Estimation and Uncertainty Surrounding the Cone of Uncertainty

Todd Little, *Landmark Graphics*

Metrics collected from Landmark's projects show that the estimation accuracy of project duration followed a lognormal distribution, and the uncertainty range was nearly identical throughout the project, in conflict with popular interpretation of the "cone of uncertainty."

Software development project schedule estimation has long been a difficult problem. The Standish *Chaos Report* indicates that only 20 percent of projects finish on time relative to their original plan.[1] Conventional wisdom proposes that estimation gets better as a project progresses. This concept is sometimes called the *cone of uncertainty*, a term popularized by Steve McConnell.[2] The idea that uncertainty decreases significantly as one obtains new knowledge seems intuitive.

I wanted to quantify some of the conventional wisdom about uncertainty. My company had collected metrics for three years on more than 100 commercial software projects, and I saw the opportunity to mine that data to expose trends that I could compare to other industry data. My findings led me to question aspects of that conventional wisdom.

Background of the project data

Landmark Graphics is a leading vendor of commercial software for the oil and gas exploration and production market. Landmark has grown largely via acquisition, and our current software portfolio includes more than 60 products consisting of more than 50 million lines of source code. The study reported here looked at three years of data from 1999 to 2002, during which Landmark collected data weekly about all 106 of its software development projects.

For each project, the project manager recorded several aspects weekly, including status, nominal phase, and current estimated delivery date. Estimates were updated whenever the core team reached consensus—approximately eight times during an average project. The average project lasted 329 days.

Most teams and project managers were quite seasoned. The average project manager had approximately 20 years of industry work experience. The project manager collected estimates from the entire team using the common industry practice of the "expert method" (that is, expert judgment using historical data and analogous projects).[3] The teams generally didn't use a formal Delphi estimation technique, but everyone could give input before the group reached an estimation consensus. The corporate culture was such that the team viewed estimates as targets—similar to "what's the earliest date by which you can't prove you won't be finished?"[4]

 0740-7459/06/$20.00

Landmark didn't follow any formal software development methodology, although several projects used iterative milestone development, and many project teams followed Microsoft Solution Framework guidelines.[5] For recording purposes, we used the four MSF phases: Envisioning, Planning, Developing, and Stabilizing.

Actual versus estimate

Figure 1 shows Landmark project data along with data from Tom DeMarco for reference.[6] For the Landmark data, the *x*-axis shows the initial estimate of project duration, and the *y*-axis shows the actual duration that the projects required. For DeMarco's data, I've cross-plotted estimated effort versus actual effort. The solid line shows the ideal case where the actual equals the estimate. The data shows significant scatter, but by and large, the actual duration or effort was longer than the initial estimate—in some cases significantly longer.

Figure 1 clearly shows a quite similar distribution scatter of the Landmark data and the DeMarco data. I recognize that comparing effort data with duration data might be like comparing apples and oranges. In our environment, and in what I've witnessed in other software product companies, staffing profiles are relatively constant throughout a project. So, for our projects, effort was roughly proportional to duration. Thus, I'll use *data* to mean both duration and effort.

In figure 2, I plotted the two data sets from figure 1 to show a cumulative probability distribution as a function of the ratio of actual to estimated data using a log scale for the *x*-axis. The solid curves represent a *lognormal* distribution curve fit through the respective data points. A lognormal distribution is similar to a normal distribution in that the frequency distribution of the log of the data is normal. It's common to report values at various probability confidences, particularly using p10, p50, and p90 to represent 10 percent, 50 percent, and 90 percent confidence respectively.

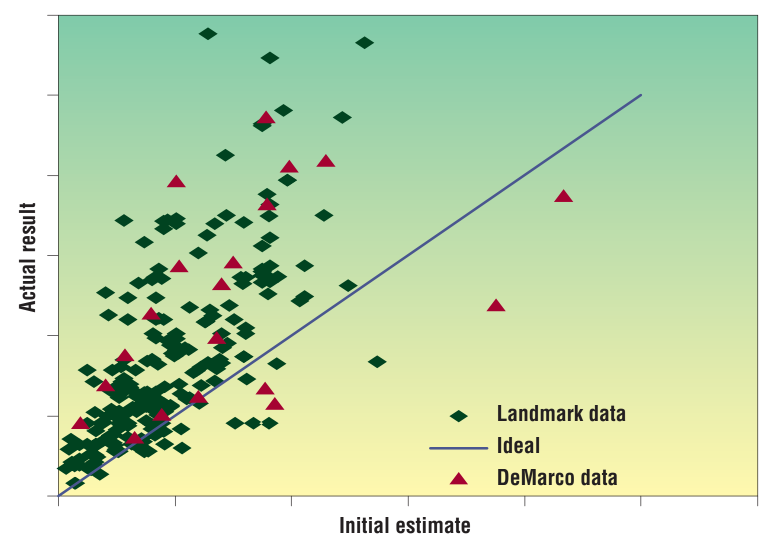


Figure 1. Initial estimated project duration (or effort) versus actual duration (or effort).

Cumulative distribution

A cumulative distribution plot shows on the *y*-axis the percentage of data samples that have a value lower than the value on the *x*-axis. For example, 20 percent of the projects from DeMarco's data were lower than the ini-

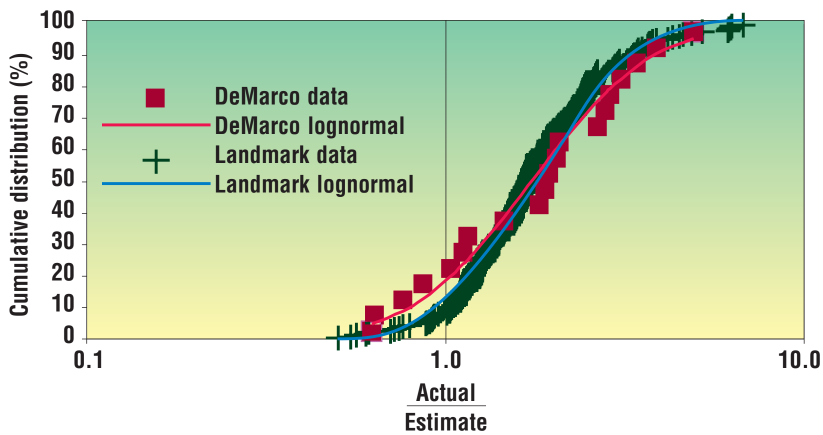


Figure 2. The cumulative distribution function of the ratio of actual to estimated data, using a log scale for the *x*-axis, for the Landmark and DeMarco data.

Just Double the Estimate

I've seen data from another company's projects that showed behavior similar to but somewhat different from the Landmark data this article discusses. Figure A depicts the cumulative distribution of this company's projects, plotting the ratio of actual over initial estimates on a log scale similar to figure 2. While the tail end greater than p50 resembles figure 2, the early portion of the curve less than p50 looks to be truncated as a straight line. The curve fit through the data is a lognormal distribution truncated at 1.0. In this example, the p50 is approximately 1.1, the p90 is approximately 1.7, and the ratio p90/p50 is approximately 1.6. This ratio is just slightly less than the 1.8 that we found from the Landmark data.

I believe that this company used a more conservative estimation policy, coming closer to a p50 estimate rather than a p10 target. This eliminated the skew but not the overall distribution. A side effect of this conservatism was that because projects had little incentive to finish early, few did—roughly half of the projects finished within a very small tolerance of their initial estimate. The projects that exceeded the estimate followed a distribution similar to what we observed with both the Landmark data and Tom DeMarco's data (*Controlling Software Projects*, Prentice Hall, 1982).

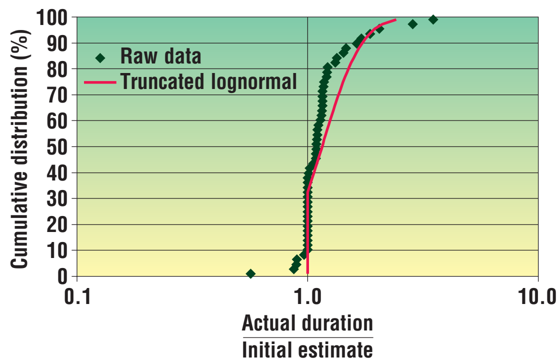


Figure A. Data from company X.

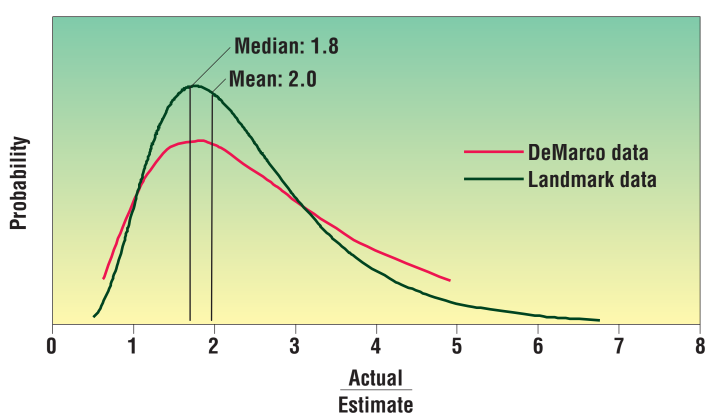


Figure 3. The probability distribution curve of actual versus estimated data.

tial estimate (ratio = 1.0). A normal distribution has a frequency or probability distribution in the shape of the classic bell curve. When plotted as a cumulative distribution function, it takes on the integral of the bell curve and shows up as an S-curve.

Figure 3 shows the curve fits displayed as probability distribution curves. Because this is plotted on a Cartesian axis, you can see that the lognormal curve shape is skewed to the right. Because of this skew, the mean (or expected value) is greater than the median (p50). In Landmark's case, the mean is approximately 2.0 and the median is approximately 1.8. This seems to validate the old adage "take the initial estimate and double it" (see the sidebar).

Landmark's data distribution is nearly identical to that of DeMarco's data, and both demonstrate that the data follows a lognormal distribution. The data distribution seems to validate DeMarco's observed definition: "An estimate is the most optimistic prediction that has a non-zero probability of coming true."[6] A variation on this definition is to say that a development target is an estimate with about a 10 percent chance of being met. Also of interest is the range between p10 and p90. For our data, the ratio of p90/p10 is 3.25, while for DeMarco's data the ratio is 5.2. Barry Boehm reported that at the Project Concept phase, the bounding uncertainty was a ratio of 4.[7] Lawrence Putnam, Douglas Putnam, and Donald Beckert published data indicating a range of plus or minus one standard deviation (p86/p14) of roughly 3.8.[8] It's not just good enough to double the initial estimate—some teams have found it appropriate to multiply by 3, 4, or even π.

The cone of uncertainty

As I mentioned earlier, the cone of uncertainty seems to imply that estimation uncertainty decreases as a project progresses. Figure 4 shows the cone of uncertainty as Boehm first reported it,[7] with relative cost ranges (total-project-cost multipliers) plotted at multiple project phases. At the Feasibility phase, the uncertainty band is a factor of 16 (from 0.25 to 4.0), while at the Concept phase, it has narrowed to a factor of 4 (comparable to our finding of a factor of 3.25 for the Landmark data). By the end of the Requirements Specification phase, it has decreased to a factor of 2.25. NASA published similar results in 1990.[9]

Using Landmark's weekly data, I cross-plotted the ratio of actual total duration over the estimated total duration as a function of relative time (the ratio of elapsed time over total actual time). I chose relative time as a means of normalizing the data so that we could easily compare it. The result in figure 5 is quite similar to the cone of uncertainty. It's not directly comparable to figure 4 because figure 5's x-axis represents relative time while figure 4's is project phase. The skew to the top half of the cone is as expected, given that the initial Landmark estimates are roughly p10.

However, what aspect of uncertainty has really been reduced? By definition, figure 5 will converge to 1.0 at project completion. What does it tell us about remaining uncertainty? An alternative view in figure 6 shows the Landmark data with the ratio of remaining actual duration over remaining estimated duration, plotted as cumulative distributions at each phase. It's surprising that these cumulative-distribution-function curves are nearly identical for each phase, showing no decrease in the relative-uncertainty bands. In contrast to what most people seem to think about uncertainty decreasing, our data shows that the remaining uncertainty bands are roughly constant and that the range of uncertainty at all project stages is approximately a factor of 3 to 4 between p10 and p90.

The pipe of uncertainty

Figure 7 shows a scatter plot of the Landmark data along with published data from Microsoft WinWord 1.0[10] and Microsoft PowerPoint 3.0.[3] The y-axis is the ratio of the actual remaining duration over the current estimated remaining duration and is plotted as a function of relative time for each project at each week. To the extent that there's a "cone," the data shows an inverted cone behavior. This is almost certainly a result of the project manager

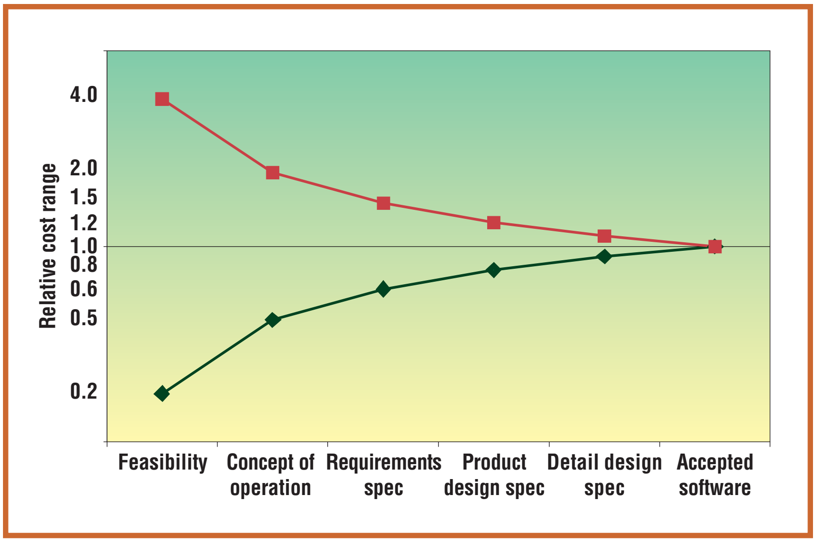


Figure 4. Barry Boehm's cone of uncertainty.[7]

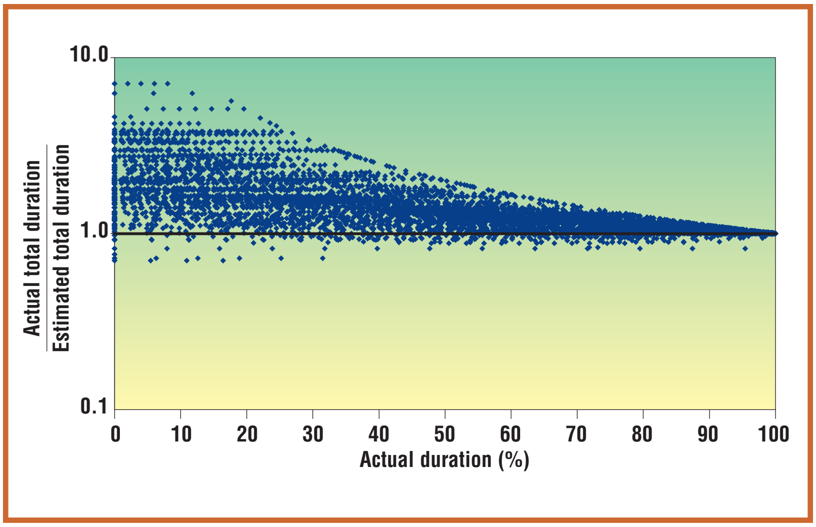


Figure 5. Estimation error over time using Landmark's weekly data.

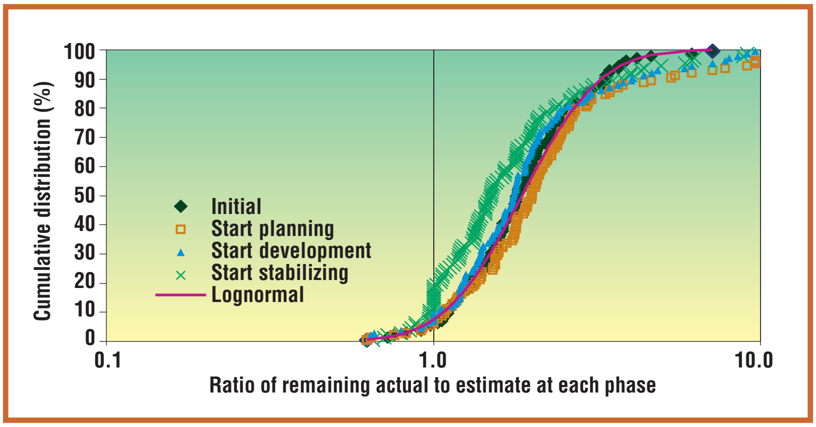


Figure 6. Cumulative distribution by project phase using the same Landmark data.

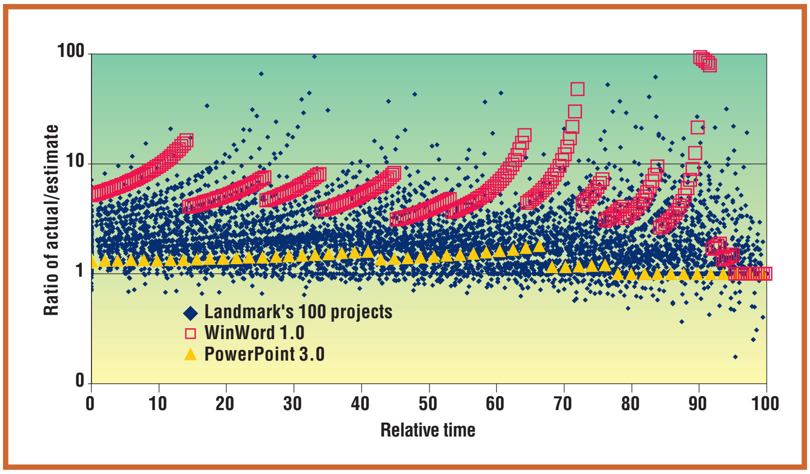


Figure 7. Estimation ratio versus time using Landmark and Microsoft data.

holding on to a deadline in hopes that a miracle will occur and the software will release. Finally the day of reckoning occurs, with no miracle in sight. In the extreme case, the ratio is a divide-by-zero as the deadline arrives, but significant work remains; at this point, the project estimate is usually reset. In many cases, this cycle repeats until the software releases.

Estimation quality factor

I also evaluated the Estimation Quality Factor, a metric proposed by DeMarco.[6] EQF measures project estimation accuracy, essentially the estimation error's reciprocal. Higher EQFs indicate a higher degree of accuracy in the overall estimation. An EQF of 5.0 represents an aggregate estimation error of 20 percent. Landmark's median EQF was 4.8 compared to the industry median EQF of 3.8 reported by DeMarco.[6] This is additional confirmation that the Landmark data represents an estimation accuracy similar to or slightly better than the industry average. See the "Estimation Quality Factor" sidebar for more on EQF and the finding that EQF also follows a lognormal distribution.

The Estimation Quality Factor

In *Controlling Software Projects* (Prentice Hall, 1982), Tom DeMarco proposed the Estimation Quality Factor as a tool for measuring a team's ability to adjust its project estimates over the project history. Figure B shows a graphical explanation of EQF. At the beginning of a project, the team makes an initial estimate; over time, they might revise the estimate up or down (the dashed green line). At project completion, the actual value is known. The estimate's variation from the actual, integrated over time, is the green area. The slashed area is a rectangle below the actual value and represents the actual value multiplied by the final duration. The EQF is the ratio of the slashed area over the green area. The reciprocal of the green over slashed areas is the estimate's relative error.

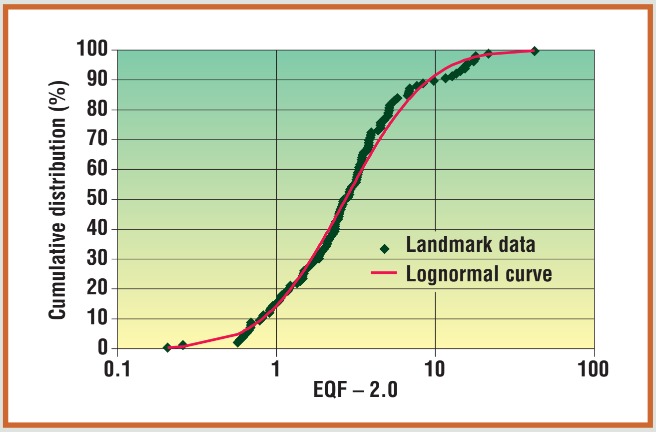


Figure C. EQF cumulative distribution for Landmark data.

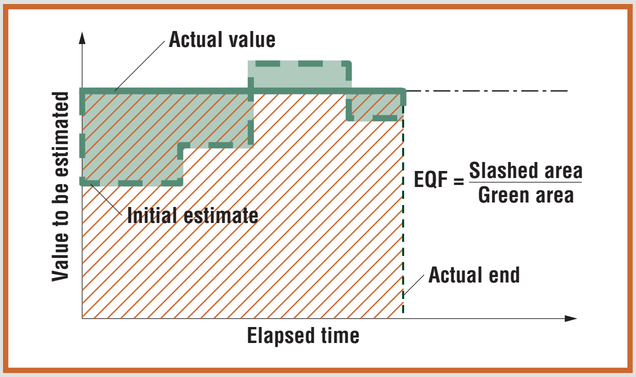


Figure B. A graphical explanation of the Estimation Quality Factor.

When the estimated quantity is schedule duration, then the worst case would be to guess completion at the end of the day and to make that same assumption each day until the product ships. The green area would then be a triangle, the slashed area would be a square, and the EQF would be 2.0. Figure C shows a cumulative distribution function for all our projects of EQF less this bottom limit (that is, EQF minus 2.0). The data fits well with a lognormal distribution curve. Approximately 10 percent of Landmark's projects had EQFs lower than 2.8, half the projects had EQFs less than the median of 4.8, and 90 percent of the projects had EQFs lower than 11.7. These results compare to DeMarco's reported median value of 3.8.

Analysis of the findings

Although I don't have definitive evidence to explain the variation in estimation accuracy I observed, I've identified what I believe are the primary causes:

- optimistic assumptions about resource availability,
- unanticipated requirements changes brought on by new market information,
- underestimation of cross-product integration and dependency delays,
- a corporate culture using targets as estimates, and
- customer satisfaction prioritized over arbitrarily meeting a deadline.

Each of these contributed somewhat to both the distribution's variation and the skew toward p10. With these characteristics in mind, some general interpretations of the data and the distributions are possible.

The data clearly indicates that Landmark's software project duration estimation accuracy followed a lognormal distribution with an uncertainty range between p10 and p90 of roughly a factor of 3 to 4. This distribution pattern is nearly identical to that found in DeMarco's effort estimation data and matches other authors' findings. It's not a new finding that software estimation follows a lognormal distribution. In fact, the log axis of the cone of uncertainty implies this type of distribution. The PERT (Program Evaluation and Review Technique) model uses a beta distribution, which has the flexibility to resemble a lognormal distribution. The use of the beta distribution appears to have more to do with its flexible characteristics rather than observed activity.[11] Nonetheless, estimates of espoused high confidence of +/– 2 months, or +/– 20 percent, are still common. Ranges that are given as +/– a constant time or constant percent are missing the problem's exponential nature.

The data confirmed that the corporate culture favored using targets as estimates. There's really nothing wrong with estimating targets. That's probably a good starting point for a p10 estimate. DeMarco and Tim Lister recently coined this the *nanopercent* estimate and recommend it as a basis for determining a more realistic estimate.[12] Planning a business around a p10 estimate would be foolish, but if the pattern I observed is typical of most software development organizations, then the full range of uncertainty could be defined by a p50 estimate of roughly twice p10, and a p90 estimate of roughly three or four times p10.

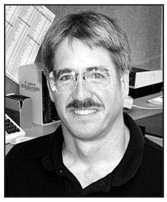
While the data supports some aspects of the cone of uncertainty, it doesn't support the most common conclusion that uncertainty significantly decreases as the project progresses. Instead, I found that relative remaining uncertainty was essentially constant over the project's life. Although one company's data might not be enough to extrapolate to other environments, I believe that the Landmark data casts doubt on the reduction of relative uncertainty being a naturally occurring phenomenon.

Perhaps Landmark has poor estimators or uses poor estimation techniques? The data doesn't demonstrate this. We find something similar to a p90/p10 ratio of 4 in multiple places in the literature. Furthermore, the EQF distribution shows Landmark to be above average with regard to overall estimation accuracy.

Reducing the range of uncertainty must be possible, right? Traditional project management approaches, several of which are based on a strong belief in the cone of uncertainty, advocate stronger project control and greater planning. While controls and planning are useful, an overly strict focus can result in attempts to solve the wrong goal. Shipping on time, to specifications, and within budget might be meaningless if a competitor is shipping software that has a greater value to the market. In that case, the competitor will win nearly every time, and the prize for "good" project management might be losing market share.

Landmark's measure of success over these three years had much more to do with customer satisfaction and market share than with meeting knowingly aggressive targets. During

About the Author



Todd Little is a senior development manager for Landmark Graphics. His interests include the generation of business value through agile software development. He's a member of the board of directors of the Agile Alliance and the Agile Project Leadership Network and program director for the Agile 2006 Conference. He's also a member of the IEEE Computer Society and the Society of Petroleum Engineers and a registered Professional Engineer in Texas. He received his MS in petroleum engineering from the University of Houston. Contact him at Landmark Graphics, PO Box 42806, Houston, TX 77242; tlittle@lgc.com.

these three years, customer satisfaction was consistently rated very high and steadily increased each year. Market share, revenue, and profitability grew as well.

Agile and iterative development approaches acknowledge the presence of uncertainty and adapt to situations rather than trying to control situations to meet the plan. Continuous feedback, particularly from customers or customer representatives, provides real control and steering. The uncertainty isn't eliminated; rather, it's managed to increase the probability of meeting market need.

Acknowledgments

I thank Karl Zachry for many fruitful discussions and for having the determination to implement the Landmark Portfolio Database that enabled the data capture. I also thank Stan Rifkin, Hakan Erdogmus, and Craig Larman for helping me clarify this article's message.

References

1. *Chaos Report*, The Standish Group, 1998.
2. S. McConnell, *Rapid Development: Taming Wild Software Schedules*, Microsoft Press, 1996.
3. M. Jørgensen, "A Review of Studies on Expert Estimation of Software Development Effort," *J. Systems and Software*, vol. 70, nos. 1–2, 2004, pp. 37–60.
4. T. Kidder, *The Soul of a New Machine*, Little, Brown, and Co., 1981.
5. M.A. Cusumano and R.W. Selby, *Microsoft Secrets*, The Free Press, 1995.
6. T. DeMarco, *Controlling Software Projects*, Prentice Hall, 1982.
7. B.W. Boehm, *Software Engineering Economics*, Prentice Hall, 1981.
8. L. Putnam, D. Putnam, and D. Beckert, "A Method for Improving Developers' Software Size Estimates," *Crosstalk*, 2005, vol. 18, no. 4, pp. 16–19.
9. L. Landis et al., *Manager's Handbook for Software Development (Revision 1)*, tech. report SEL-84-101, NASA Software Eng. Laboratory, 1990.
10. S. McConnell, *Software Project Survival Guide*, Microsoft Press, 1998.
11. K.R. MacCrimmon and C.A. Ryavec, "An Analytic Study of the PERT Assumptions," *J. Operations Research Soc. of America*, vol. 12, no. 1, 1964, pp. 16–37.
12. T. DeMarco and T. Lister, *Waltzing with Bears*, Dorsett House, 2004.