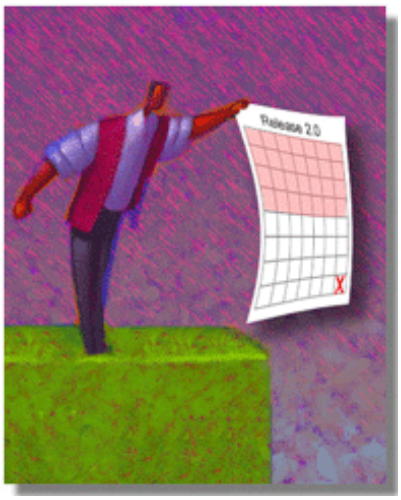
Saving for a Rainy Day

by **Karl Wiegers**
Principal Consultant
Process Impact



An article in a recent home remodeling magazine noted that "the prudent homeowner sets aside 20 percent of his budget for cost overruns and unexpected expenses." This sounds like good advice, doesn't it? Most remodeling projects include surprises, like the time the carpenter who built my deck spent an unplanned, exhausting half-day augering through a tangle of twisted tree roots to dig holes for the deck supports.

Most software project teams also experience cost overruns and unexpected expenses. They begin with uncertain estimates and encounter schedule setbacks. They experience requirements growth and technology shifts. They discover essential tasks that weren't planned. They are thwarted by slips in other projects that tie up people or resources they expected to be available on a specific date. Risks materialize into schedule-destroying problems. Life is just full of surprises.

But few software projects create a schedule or budget reserve to help them deal with such eventualities. To respond effectively to your project's changing realities, it's wise to save a little time and money for a rainy day.

This article makes the case for incorporating a contingency buffer (also known as management reserve or safety time) into your project plans to accommodate the unforeseen and the unknown. We'll discuss ways to determine the size of contingency buffers and how to include them in your negotiations with managers and customers. We'll also look at critical chain management analysis as a technique for deriving cumulative contingency buffers to account for estimation inaccuracies.

Selling the Skeptics

Before we dive into the mechanics, let's explore how to convince managers and customers that contingency buffers will increase our chances of success. Honest estimation and project planning often leads to schedules that senior

managers regard as unacceptable. When managers spot a task labeled "contingency" in the plan, they typically think of it as unnecessary padding that can be removed to shorten the schedule. They may not recognize that thoughtfully derived contingency buffers are a practical acknowledgment of reality, not a crutch for weak-kneed estimators who are afraid to make commitments.

If you're a project manager, discuss with your managers early on the value of including some safety time and money, instead of surprising them with a project plan that they will perceive as padded. Then, use these two strategies to produce convincing numbers:

- **Analyze Historical Records**. These can be your most effective weapons in battling unreasonable imposed schedules. Compare previous schedule and budget estimates with actual results, identify factors that contributed to differences, and use this information in your buffer calculations.
- **Use a Proven Method for Calculating Your Buffers**. You can make your case more effectively if you can describe how you came up with your contingency buffer durations. We'll examine a couple of options in the next section.

Once your plan is in place, resist the pressure to remove your buffers and make impossible promises, no matter what your managers, customers, or marketing want to hear. If your manager or customer claims that some other provider promises a shorter schedule without that pesky contingency buffer, ask how likely the other provider is to achieve that target. Does their company have better people and processes, or are they just more optimistic? Also ask about the other provider's track record of actual performance compared to estimates and promises (and be prepared to show your own).

In addition, you can point out that building in slack will increase the chances of delivering on schedule and avoiding litigation that might arise from an unsatisfied contract.[1] I recently consulted on a lawsuit involving a vendor that began missing deadlines partly because extra communication cycles were required to pin down the client's requirements. As the vendor's schedule included no contingency buffers, it was impossible for them to absorb any of these slips without directly affecting the delivery date.

Bring in the Reserves

Project managers typically use one of the following approaches to calculate contingency buffers:

> Option 1: Add a "safety increment" to every task.
>
> Option 2: Place a buffer at the end of a set of project activities (e.g., a major milestone, a development phase, or an interim release) or as a separate task at the very end of the project schedule.

I do *not* recommend Option 1 -- simply padding your estimate for each task -- because it increases the planned duration and cost of the entire project by the

expansion factor without differentiating among tasks. Protecting the tasks on the project's critical path (the longest sequential path of essential activities that runs from initiation to delivery) against overruns must take top priority. Tasks that do not lie on the critical path already have some slack time following them; therefore, they can tolerate some slip without affecting the project's schedule as a whole and shouldn't need additional safety margins.

Of course, if a task that is not originally on the critical path slips by too much, it might move onto the critical path. In addition, if your estimated duration for a specific task is far too low, then it might look as though that task is not on the critical path, even though it really is. Even when using contingency buffers, you need to generate the most accurate estimates you can for each task.

Expanding each task estimate also increases the risk of succumbing to Parkinson's Law: "Work expands to fill the time allocated to it."[2] That is, if you include a safety day in a task that you estimate will require four days of effort, then you're likely to spend all five days completing the task. It's an easy trap to fall into, although some studies have suggested that software people are not as prone to Parkinson's Law as you might expect.[3] The *fear* managers have that Parkinson's Law will prevail, however, leads them to cut estimates and compress schedules to keep the pressure on. This is usually counterproductive. As consultant Tim Lister pointed out, "People under time pressure don't *think* faster," and software development involves a lot of thinking.[4]

A far better approach is Option 2: to incorporate contingency buffers at the end of major development phases, at the end of the entire project, or both. In this scheme, you estimate how much additional money and time (as discussed below) you need to yield a high probability of completing on schedule and within budget. You assign these safety margins to discrete tasks placed at the end of your project schedule and/or after major milestones. Don't assign resources to these tasks, as they simply represent extra quantities of time or money that the resources already assigned to the work tasks might need to complete them. In the Rational Unified Process® you could include a contingency activity either at the end of the Inception, Elaboration, Construction, and Transition phases, respectively, or at the end of each planned iteration.

As your project progresses, some tasks will take longer than estimated, and some of those tasks will be on the critical path. As this happens, you must reduce your remaining contingency buffer time by one day for every day of overrun you experience on critical-path tasks. (Without a contingency buffer, if a task on the critical path slips, then the project as a whole will slip by the same amount.) Keep an eye on the balance remaining in your contingency buffer as part of project tracking. If you are depleting your buffer more rapidly than your team is completing the planned work, then you may need to renegotiate your plan with stakeholders.

Your team members should aim to meet the schedule targets they originally estimated for individual tasks *without* the contingency buffers -- which are intended as a safety margin for overly optimistic estimates. As a project manager, however, base your commitments to stakeholders on the estimated schedule for all project tasks *plus* the contingency buffers. That is, work internally to the nominal estimates, which will lead to your *planned* delivery

date. But include the buffers in your *committed* delivery date. And if you don't consume the entire buffer, you'll finish early!

Some contracts base a reward structure on the portion of the buffer that remains unused when the project is completed.[5] When the Rochester, New York, airport repaved its main runway, the contractors received a bonus for every day they came in ahead of the committed schedule. They finished early, they got their bonus, and those of us who lived in the flight path for the secondary runway were delighted. Of course, it's possible that a bonus incentive would lead a vendor to pad a bid with an excessive contingency buffer, which is why clients should prepare their own estimates as a reality check.

How Big Is Your Buffer?

As a general guideline, the total project schedule reserve should be 5 to 10 percent of the sum of the estimated duration of all project activities[6] (*not* 5 to 10 percent of the overall project schedule; that's not enough). The higher number is appropriate for especially complex projects or those that involve unusual risks, uncertain or churning requirements, or bleeding-edge technologies.

To estimate an appropriate contingency buffer for your project, first identify areas of uncertainty that could lead to poor estimates. These might be the most technically challenging or innovative parts of the project, tasks that use unfamiliar techniques or technologies, or poorly defined requirements. Once you've identified the major sources of uncertainty, set an appropriate safety margin (perhaps 25 percent) for the time you allocate to corresponding tasks.

Next, examine your past projects for recurring patterns that led to delays, such as chronic requirements growth, overlooked tasks, or consistently overly optimistic estimates. One company came to me for help because they were consistently overrunning their schedules by at least 25 percent. When we looked at the paper trail, we saw that their projects experienced an average scope creep of about 25 percent. It didn't take a rocket scientist to figure out that they should include a 25 percent contingency buffer in their project schedules to accommodate requirements growth.

Finally, use risk analysis to estimate the possible schedule impact if any of the serious risks (known factors that could pose a threat) materialize or if external dependencies are not satisfied.[7] Multiplying the estimated probability that the risk *could* become an actual problem by the potential schedule impact if it *does* become a problem yields a risk exposure in units of time. You should factor the risk exposures from your top ten or so risks into your contingency buffers.

For example, suppose you plan to subcontract a critical component of your next project to a specific vendor. The last four times you used this vendor, they delivered on schedule twice and were four weeks late twice. Therefore, your best estimate of the vendor's future performance (assuming that the vendor is doing similar work for each project) is that they have a 50 percent probability of being four weeks late, for a risk exposure of two weeks.

You should also reconsider any remaining contingency buffers at the end of

each phase or major milestone. Suppose you based your contingency in part on a possible late delivery for the component this vendor supplies. Once you receive the component, you can perhaps shrink the contingency buffer for a future phase, because that risk can no longer delay the project. Alternatively, you might want to increase a future buffer if you identify new risks, if the potential exposure from a specific risk increases, if a dependency fails, or if previous buffers turn out to be too small.

Your risk analysis will also help you sell managers and customers on the need for contingency buffers. I know of a telecommunications company that incorporates a risk analysis into its standard estimation process for that very reason.

Critical Chain Project Management

As we already noted, Option 1 above (add a "safety increment" to every task) is not a good approach because there's a risk that Parkinson's Law will prevail. In addition, you would have to increase the estimates by a surprisingly large amount to substantially increase the chance of meeting your schedule.

As Figure 1 illustrates, the probability distribution for an estimate is not symmetrical: A long tail extends to the right along the time axis.[8] That is because many factors and events could potentially delay the task completion, but the most damaging events have a low probability of occurrence. However, a minimum time is required to complete the task even under the best of circumstances, so the probability distribution is skewed to the right. The area under the curve from the left edge to a given time indicates the probability that the task will be completed by that time. Adding enough safety time to move from, say, the 50 percent probable estimate to the 90 percent confidence level can double the duration. Including a safety factor in each task therefore greatly extends the schedule, although it does increase your chances of success.

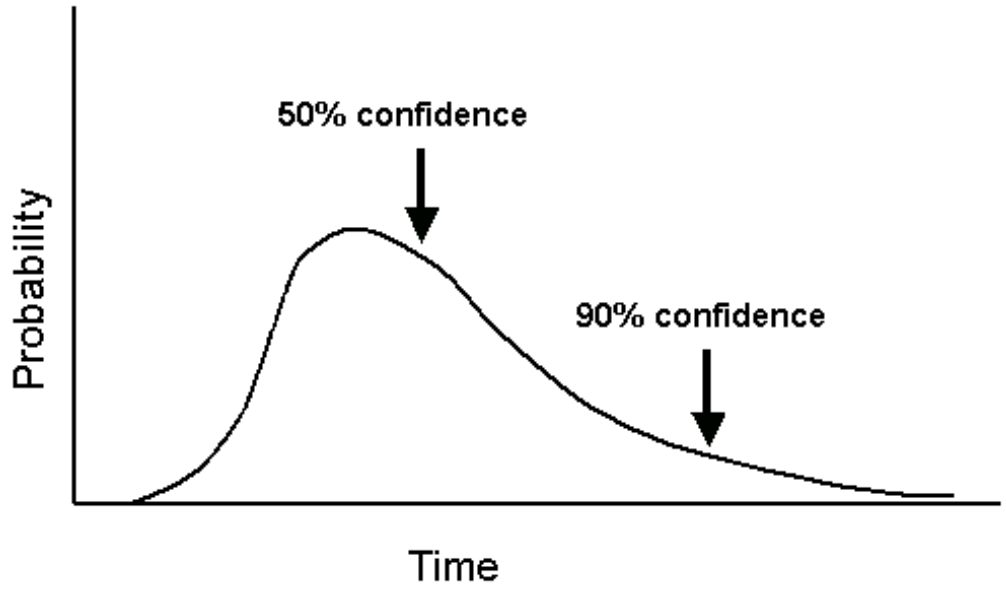


Figure 1: Probability Distribution for a Task's Estimated Duration. Note the long tail extending to the right. To move from a 50 percent probability of being done by a specific date to a 90 percent probability, you might have to double the estimated task duration.

Critical chain project management offers a more sophisticated approach to building contingency buffers.[9,10] Unlike the task-oriented critical path concept, the critical chain considers both task and resource dependencies. The critical chain of tasks is the longest sequence of tasks that leads to project completion, after all resource conflicts have been resolved. In critical chain planning, you estimate each task at the 50 percent confidence level, which means that about half of the tasks will be completed early and about half will be late. In addition, you remove the safety time from individual tasks, which encourages team members to strive to meet the stated estimates.

The schedule safety net is provided through *feeding buffers* included at the end of major activity sequences along the critical chain of project tasks and an additional *project buffer* placed at the end of the critical chain (Figure 2). To calculate the buffer sizes, first estimate the safety time that would increase the estimate probability from 50 percent to 90 percent for each task in the critical chain. Then, take the square root of the sum of the squares of these individual tas

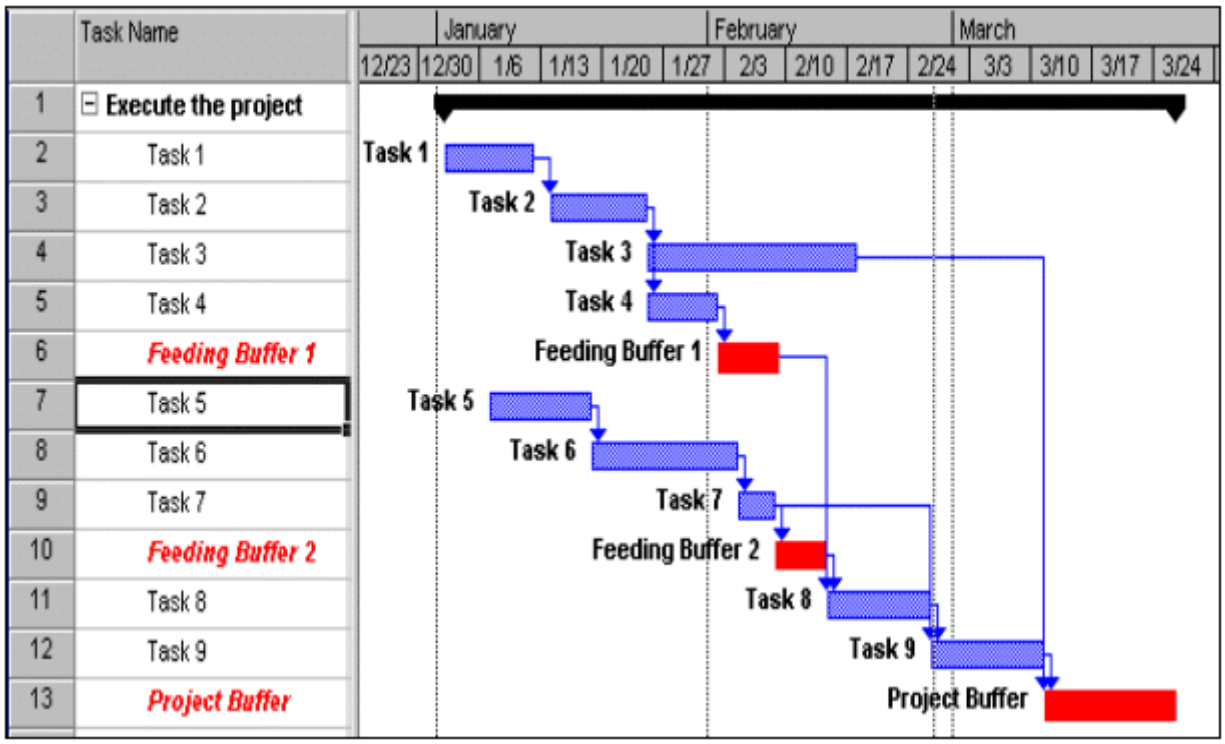


Figure 2: The Critical Chain Approach. *Feeding buffers* and a *project buffer* are treated as tasks in the schedule.

contingencies.[11] This results in a buffer considerably smaller than one you would get by simply totaling the individual safety times for the critical chain tasks. However, the project schedule buffer should be at least 25 percent of your critical chain duration.[12] The safety buffers in critical chain project management represent a pooling of risk -- the risk of not completing tasks on time -- as a technique for dealing with tasks that do slip. Studies suggest that using critical chain contingency buffers for planning can substantially improve a project's schedule performance.[13]

Your Turn

To start incorporating contingency buffers into your project plans, try the following steps:

1. Identify past projects and situations in which contingency buffers would have been helpful.
2. Select a method (e.g., Critical Chain Project Management) that will help you estimate appropriate contingency buffers on your projects.
3. Use historical records in combination with the calculations that result from applying the method you selected to persuade your managers and other stakeholders to accept your buffers.

As my colleague Michael Green says, "It's not my estimating process that's weak. It's my chronic inability to see the invisible, know the unknowable, and predict the unpredictable that keeps throwing me off." Given the many sources of uncertainty on software projects, it's wise to use contingency buffers to protect your team from undue stress and help them meet their commitments.

Acknowledgments

The author appreciates the contributions made to this article by Chris Fahlbusch, Ellen Gottesdiener, Michael Green, Karen King, Kathy Rhode, and Johanna Rothman.

Notes

[1] Tom DeMarco, *Slack: Getting Past Burnout, Busywork, and the Myth of Total Efficiency*. Broadway Books, 2001.

[2] C. Northcote Parkinson, *Parkinson's Law and Other Studies in Administration*. Ballantine Books, 1979.

[3] D.R. Jeffery and M.J. Lawrence, "Managing Programming Productivity," *Journal of Systems and Software*, vol. 5, no. 1 (January 1985), pp. 49-58.

[4] DeMarco, *Op. Cit*.

[5] Robert W. Wysocki, Robert Beck Jr., David B. Crane, *Effective Project Management,* 2nd Edition. John Wiley and Sons, 2000.

[6] Wysocki, *Op. Cit*..

[7] Karl Wiegers, "Know Your Enemy: Software Risk Management," *Software Development*, vol. 6, no. 10 (October, 1998), pp. 38-42. http://www.processimpact.com/articles/risk_mgmt.pdf

[8] Eliyahu M. Goldratt, *Critical Chain*. The North River Press, 1997.

[9] Lawrence P. Leach, *Critical Chain Project Management*. Artech House, 2000.

[10] Richard E. Zultner, "Project Estimation with Critical Chain: Third-Generation Risk Management," *Cutter IT Journal*, vol. 12, no. 7 (July 1999), pp. 4-12.

[11] Leach, *Op. Cit*.

[12] Leach, *Op. Cit*.

[13] Eli Schragenheim and H. William Dettmer, "Does Your Internal Management Meet Expectations?" *CrossTalk*, vol. 14, no. 4 (April 2001), pp. 19-25.

For more information on the products or services discussed in this article, please click here and follow the instructions provided. Thank you!