

ON TIME AND UNDER BUDGET

Probability Management for Projects

How to Manage the Odds of Project Success

© Marc Thibault

marc@smpro.ca

November 2011

Version 4

Contents

1	Introduction.....	3
2	The Flaw of Averages in Project Estimates	4
3	Probability Management Tools and Techniques	9
4	Modeling with Probability Management.....	24
5	Calibrating Estimators	31
6	Manage the Odds of Success.....	33
7	Resources	34
8	The Author	35

For over two decades, I've been helping my clients refine their objectives and requirements, and then creating plans to realize them. Over most of this time, I've been honing 'The Art of the Plan' and perfecting the tools and techniques I use to create realizable plans.

One thing always eluded me: No matter how well the plan was executed, projects tended to finish late and over budget. It was clear that the numbers were wrong, but there was nothing in 'best practices' to fix them.

The fallout from this wasn't just the operational cost of overruns. Repeated failures and impossible deadlines take their toll on morale. Crunch time heroics are exciting the first few times, but it quickly wears thin, with predictable effects on productivity and staff retention.

Then I ran into Sam Savage and his book, *The Flaw of Averages*, and discovered that PERT and CPM calculations give us wrong results. Not just that, but the errors are persistently one-sided; they give us optimistic estimates and projects with poor odds of making their targets.

Probability Management has the potion that will end this curse – with realistic estimates and attainable targets.

Probability Management won't make projects finish faster or cost less, but it can help reduce the number and severity of project failures.

There's an Excel workbook to go with this document at <http://smpro.ca/ProbMan>.

The plan said we'd be done in six months. In the end it took ten months. What happened?



Every other week, yet another study comes out with a list of the things that cause projects to miss their targets and fail. And it's always the same list.

When something gets to be predictable, we have a saying: We say "Plan on it."

So why didn't we plan on it? It should be clear from the rate of project failures, especially in infrastructure and software projects, that our plans are unrealistic. The plan should have allocated more time and it didn't.

The plan was wrong.

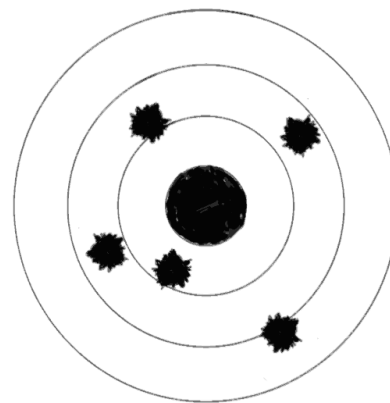
I'm going to focus on one aspect of that wrongness.

The Math is Wrong

With competent project management, capable staff, and skillful execution, projects still come in late and over budget. The way we estimate cost and duration is fatally flawed.

Calculating with averages and expected values runs afoul of the Flaw of Averages. It produces hopelessly optimistic estimates and unattainable targets. Speaking of targets...

The average placement of the five shots is in the center of the bullseye. If you were going to bet on where the next shot will go, would you bet that it hits dead center, in the black?



Probably not.

But that's exactly what you do when you choose an average as a project target.

If you're going to bet on the next shot, you want to see where the other shots have gone. It's an easy bet that the next shot will hit within the outer ring, but what are the odds for each of the other rings?

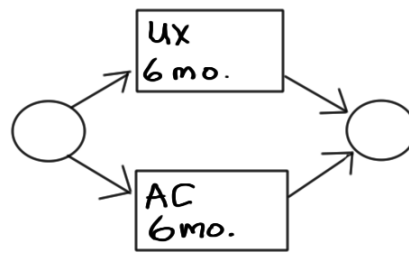
An average throws away information that you need. To make informed management decisions, you need something that shows you the uncertainty in your data; you need to see the range of possibilities.

The Average is Wrong

Let's look at a small part of a project plan. In our project plan we have two tasks, UX and AC that we plan to do concurrently. The

estimated average duration turns out to be the same for both tasks: six months.

If we leave it at that, we'll be planning on finishing them both in the same six-month period – and we'll be wrong.



The six month duration is an average, so looking at each task individually, the probability

that it finishes early or on time is pretty much a coin toss; heads it's early, tails it's late. That's not necessarily exact, but changing the odds, or using a different 'expected' value instead of an average, doesn't eliminate the problem – it just makes it harder to describe.

The way the plan is structured, we can't pass the next milestone until both UX and AC are done; the milestone waits for whichever one finishes last.

Heads or Tails

Let's simulate this bit of the project with two coin flips – one for each task. If we get two heads, both tasks are early. But there's only one chance in four of that happening. For the other three possibilities, one or both tasks are late, and the odds are 3:1 that this piece of the project is late.

The big question is "How late?"

If we're going to insist on calculating with averages, we need to know the average time to have both tasks done and reach the milestone.

Rather than guessing, maybe we can fix the problem by making an assumption about how the average was derived. We could put numbers to heads and tails so that they average six months, and simulate the possible results. Let's assign 5 months to heads, and 7 months to tails. Usually we like to have a lot more numbers to calculate an average, but in this experiment we only have heads and tails, so we're stuck with only two.

	HH	HT	TH	TT
UX	5	5	7	7
AC	5	7	5	7
BOTH	5	7	7	7
AVERAGE = $\frac{26}{4} = 6.5$				

The simulation gives us an average completion time of 6½ months – problem solved.

Or is it? Suppose we use 4 and 8 months instead of 5 and 7:

	HH	HT	TH	TT
UX	4	4	8	8
AC	4	8	4	8
BOTH	4	8	8	8
AVERAGE = $\frac{28}{4} = 7$				

Now the average outcome is 7 months.

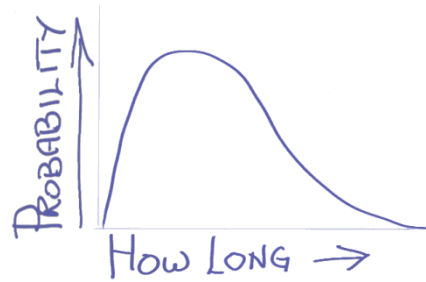
The combined duration depends on how the individual averages are derived. If we want to estimate the combined task duration, we need to know where the six-month estimate came from, and *that information was discarded when we reduced it to a single number.*

We really need a lot more numbers – not to average, but to make something useful for estimating.

The one thing we know is that the six-month estimate is optimistic.

And it doesn't stop there; what we're passing on to the plan is yet another average, making sure that the error cascades through the rest of the plan, getting increasingly optimistic as it goes.

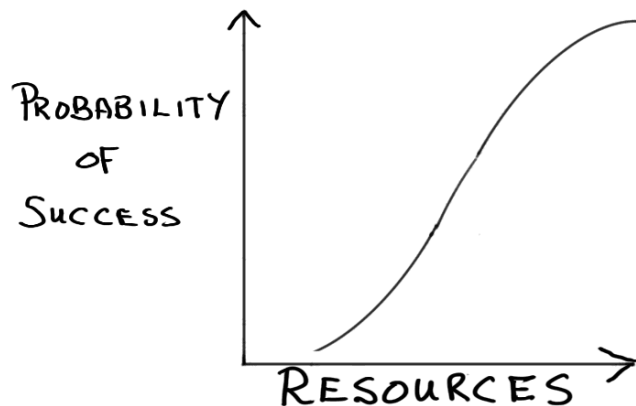
Whether it's about a whole project or a single task, "How long will it take?" doesn't have just one answer. It has a whole bunch of answers, each with its own probability of being right.



The six-month estimate can't be characterized with just heads and tails. The plausible durations go from a little less than six months to a lot more than six months, with many possible combinations – each with its own probability.

This uncertainty – many possible values, each with its own probability – applies to every value in the project model. And each one reflects all of its predecessors' uncertainty.

When you choose a target, or make a resourcing decision, you also choose the probability of success (or the chance of failure if the glass is half-empty). Whether you're conscious of the choice or not, you're making a risk management decision. Making the choice conscious is where Probability Management comes in.



3 PROBABILITY MANAGEMENT TOOLS AND TECHNIQUES

Probability Management is a set of tools and techniques for working with sample distributions instead of single numbers.

–It’s about factoring uncertainty into your estimates.

–It’s about banishing the Flaw of Averages from your calculations.

–It’s about creating plans with realistic targets.

Probability Management includes five basic tools for modeling and estimating projects: Sample Distributions, Monte Carlo Simulation, Distribution Strings, Visualization, and Interaction.

3.1 PM TOOL #1: SAMPLE DISTRIBUTIONS

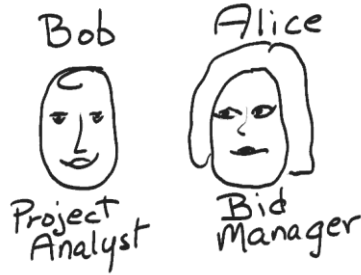
Estimating is about dealing with uncertainty – with the many answers to how much or how long. How long will it take to build this software module or this bridge? How much will it cost to build this house or this tunnel? These are *uncertain variables*.

We draw a circle around the uncertainty – quantify it, constrain it – with historical data, expert opinion, or a combination of the two, expressed in a *sample distribution*.¹

Let’s look at a simple use of a sample distribution:

¹ In *The Flaw of Averages*, Sam calls this a SIP: a Stochastic Information Packet

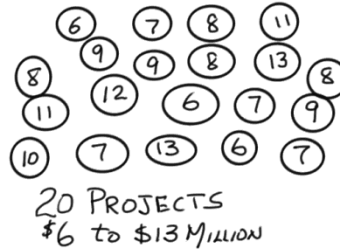
Bob is briefing Alice on the scoping effort for a project the firm is thinking of bidding on. Bob tells her that his preliminary analysis indicates they can do it for \$8.75 million.



Alice wants to know what the probability is that they can do it for \$8.75 million or less, and make a profit on the project.

Bob explains that he's collected a reference sample set of 20 projects similar to this one. Their final costs ranged from \$6 to 13 million, and the average was \$8.75 million.

"Since it's the average," Bob says, "the probability is around 50%."



"So," Alice notes, "it's a coin toss; heads we're over budget and take a bath, tails we make some money."

Alice isn't happy with odds that thin. "What would it take to raise the probability of at least breaking even to 80%?"

Alice understands the first lesson of the Flaw of Averages:

On average, the average is wrong.

The outcome will probably be something else, higher or lower, and half the time we won't like it.

So, what does Bob do?

The first rule of Probability Management is:

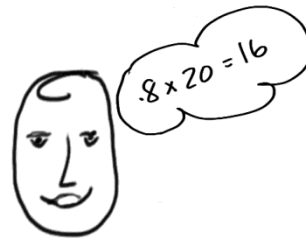
Let the data speak for itself.

Bob goes back to his sample set. Each sample is a project taken from real world history. The sample set is carefully selected and structured so that there's no reason to believe that any one is any more likely than any of the others—no reason to believe that any one sample project is a better or worse predictor of this project's outcome.

Bob does a quick mental calculation and finds that 80% of 20 is 16. Then he lists the costs for the 20 projects in ascending order, and he counts to 16:

6, 6, 6, 7, 7, 7, 7, 8, 8, 8, 8, 9, 9,
9, 10, 11, 11, 12, 13, 13.

Sixteen of the projects – 80% – had final costs of \$11 million or less. This is the answer to Alice's question.



All Bob needed was simple arithmetic – no regression analysis, no calculus. This is a thread that runs through Probability Management – we don't do Greek.

$$s^2 = \frac{1}{n-p} e'e = \frac{1}{n-p} y'My = \frac{1}{n-p} S(\hat{\beta})$$

Bob used two of Probability Management's more powerful tools:

- he let the data speak for itself by casting it as a *sample distribution*, and
- he used a percentile to make the link between value and probability.

With Bob's little list, Alice can explore other possibilities, decide how much risk is appropriate and whether it's likely that they can come up with a competitive bid.

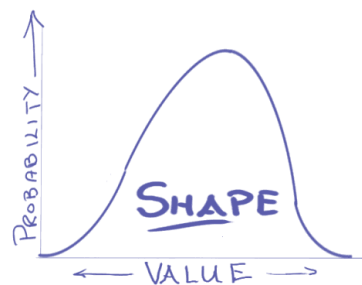
Sample Distributions Quantify Uncertainty

Let's say a biologist wants to estimate how many squirrels there are in a large conservation area. She decides to use the number of squirrels per square kilometer as her uncertain variable. The standard approach is to select a small but representative area of the park, count the squirrels in it, and divide by the area of the sample area, to get the squirrels per square kilometer. If the squirrels move around a lot, she'll use another method, called 'mark and recapture' or some other variation on the theme. They also produce a single number. Then she uses some assumptions and statistical gymnastics to estimate the sampling error and other statistical properties of the sample, including the minimum and maximum. This is one sample, with one number, carrying a heavy load.

The Probability Management way would be to repeat the measurement twenty, fifty, or a hundred times in different areas. The list of results would be a sample distribution with many sample values. No assumptions are needed. The statistical properties are all right there in the sample distribution. The data can speak for itself.

A sample distribution is always a list of numbers. In software terms it's a one-dimensional array or vector. Each element of the vector holds a value drawn from the plausible values of the uncertain variable, like the dollar values in Bob's reference projects or squirrels per square kilometer. It could be a list of all the possible values or, if that's unwieldy, it could be a representative subset.

Like all probability distributions, a sample distribution has a characteristic *shape*. A distribution's shape refers to the relationship between values and their probabilities. Every value has exactly one probability,



whereas a probability can have zero or more associated values.

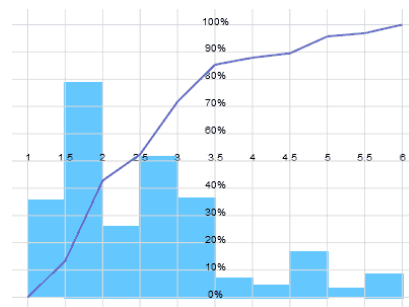
A sample distribution is a *non-parametric probability distribution*. It makes the connections between values and their probabilities without making any assumptions about the shape of the distribution.

Parametric probability distributions, such as the Normal (Bell-Shaped) Distribution, have well-defined shapes and assume a variable's uncertainty can be captured with a formula and some parameters. One of the most interesting challenges for parametrics is figuring out just which type of distribution is the right one for a particular variable. With sample distributions, we don't care; only the data matters.

A significant problem with parametric distributions is that, like averages, they throw information away. For every parametric distribution, there's at least one heroic calculation that will tell you how badly the distribution matches the data.

A sample distribution can describe any real-world uncertainty; it not only matches the data, it *is* the data.

If you keep track of how long it takes to get from home to the office every workday without fail for a year or two, the list of travel times you compile is a sample distribution.



Each element in the list is a sample. Any element in the list is as good a predictor of tomorrow's travel time as any other. Whenever we build a sample distribution, we always make sure that each sample in the distribution has about the same probability of being right as every other. How we do that depends on the data source and how we collect our samples. In this case, it just means including every day's time – including the days when something interesting and unusual happened.

Although the samples are equally likely, the sample times are not; there will be more samples with times around the typical travel time

and fewer samples with times like on the day two feet of snow fell. The way sample distributions link values to their probabilities is that there will be more samples with higher probability times, and fewer samples with lower probability times.

More formally: In a sample distribution, the probability that the actual value lies within a particular range is approximately proportional to the number of samples whose values fall within that range. (If you aced Statistics 101, you'll have noticed that sample density substitutes for probability density.)

It's that simple, but satisfying that specification may sometimes be a challenge. Read Douglas Hubbard's *How to Measure Anything* for inspiration. If we can measure anything, we can come up with a distribution for anything. Hubbard's pragmatic approach to calibrating estimators is also useful.

3.2 PM TOOL #2: SIMPLE MONTE CARLO SIMULATION

Conventional project estimates are the result of one set of calculations – a single run through the project model with idealized values for each of the input assumptions. The calculations might also include best and worst cases but that just means you have three flawed numbers instead of just one.

Because the calculations run headlong into the Flaw of Averages, the estimates will be hopelessly optimistic and will result in setting unattainable targets.

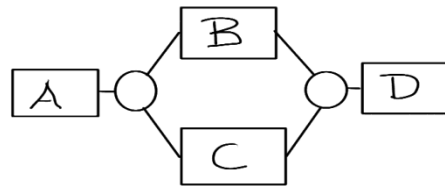
The Probability Management approach is to simulate the project hundreds or thousands of times, each time with a different combination of input values and each time producing a different but equally likely outcome. The results give probabilities for the full range of plausible cost and duration for the project.

Each model input value is taken from a sample distribution and each output value is a sample distribution. This approach is a variation on *Monte Carlo Simulation*. (If you need a refresher, Wikipedia has an excellent article at http://en.wikipedia.org/wiki/Monte_Carlo_simulation.)

Using sample distributions simplifies and streamlines Monte Carlo simulation. We aren't burdened with generating random numbers on the fly, calculating probability density functions or curve-fitting results. The math is only as interesting as the model itself demands. For projects, that's pretty much limited to addition, multiplication and comparison. We don't do complex math for simple functions.

Perhaps we should call it 'Las Vegas Simulation' – it's not as sophisticated as Monte Carlo and it's more pragmatic.

Let's say we have a project whose model is really simple. The model inputs are the durations for four tasks, two of them concurrent, producing a total elapsed time calculation like



$$R = A + \max(B, C) + D$$

Each of the duration inputs is an uncertain variable with a sample distribution. We're going to simulate the project with 1,000 trials. A thousand trials may seem excessive, but it only takes 3 inputs with ten possible values each to generate a thousand possible outcomes. In practice, we'll have many more inputs, each with many more than ten possible values, so our result distribution will be a subset of all the possibilities. In this particular case we have four inputs with 1,000 values each.

'Las Vegas' Simulation

The Probability Management approach to simulation is breadth-first. For each mathematical operation in the model, we feed the whole thousand values from each input into a single array operation, producing a thousand-value result.

This result gets fed into the next operation. In this case the operations would be an 'array maximum' with B and C, and two 'array add' to sum everything up.

The input sample distributions are randomly ordered to begin with, so we don't need to generate random variates for simulation. (More about this below.)

The final result has the sample distribution for the model output. The 'Las Vegas' approach is conceptually and computationally simpler than conventional Monte Carlo, uses fewer compute cycles and works with sample distributions throughout, banishing the Flaw of Averages.

The Road to Vegas

Let's say we're costing some road work and we know we're going to need some fill but we're not sure how much. We're also not sure of how much we'll have to pay for it when the time comes. So, we get some history and make sample distributions for the number of yards and the cost per yard.

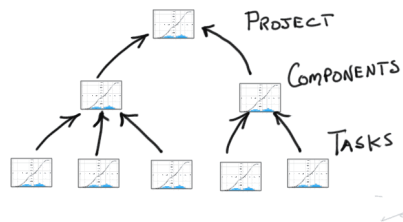
```

$/yard = [ 8, 12, 14, 10, ... a(i) ]
Yards = [ 500, 700, 300, 400, ... b(i) ]
$ Cost = [4000, 8400, 4200, 4000, ... a(i)x b(i)]
    
```

Then we multiply the two vectors. The result is a sample distribution for the cost of fill that we can roll into our project estimate.

This is basic array math that's a breeze to do with a spreadsheet or any programming language (e.g. Excel and VBA).

Using sample distributions this way, we can start with sample distributions at the task level and roll them up to the top of the tree as distributions all the way.



This banishes the Flaw of Averages from our estimate.

There's No One Right Number

All of these rolled-up calculations won't give us the 'one right number'. The result is a distribution, because the 'one right number' doesn't exist. The uncertainty in the project outcome is inescapable.

If a commitment is needed, we can make a commitment. Thanks to sample distributions, we can choose cost and duration numbers with whose odds we're comfortable (like Alice's 80%). But rather than a commitment, we should think of it as a resource allocation. We want to allocate the resources the project needs to have reasonable odds of succeeding. We can manipulate those odds by adjusting the allocations.

In some contexts we may need to talk about the risk of failure rather than the probability of success, but the principle is the same. Human nature and the perception of probabilities come into play; an 80% probability of succeeding sounds generous until it's restated as a 20% chance of failure. Probability Management can give us the numbers, but how much risk to take on is a purely human decision.

The Importance of Order

Sample distributions have two independent properties. The first is the *shape* – the sample properties we've been focused on so far. The other is the *order* – the sequencing of the values in the sample distribution's vector.

Order is normally represented by a distribution's *rank order*, which is easier to show than to explain:

Sample Vector	40	30	80	70	60	20
Rank Order	3	2	6	5	4	1

The sample vector and its rank order are sorted the same way. The smallest sample: 20, has the smallest rank: 1. The next is 30 and 2, and so on up to the largest: 80 and 6.

All the common statistical properties are functions of the shape. The order comes into play when we're operating on multiple distributions; correlation, or lack of it, is a function of order. The first thing correlation analysis does is throw away the shape.

In general, two distributions are considered positively correlated if smaller values of the one align with smaller values of the other. They're negatively correlated if smaller values of the one align with larger values of the other. The small-to-large property of a distribution is in its rank order. The absolute values and shapes of the distributions don't matter. We could be relating housing starts to bushels of oranges shipped each week and, as long as the size relationship is there, they're deemed to be correlated.

We manage order to preserve correlation on the one hand, and to avoid manufacturing false correlation on the other.

If we're modeling a function of two or more distributions (like the road-fill example) that are independent – that is, not correlated – their orders should be different. We want to mix up the orders so that the simulation trials involve many different combinations. If the orders are similar, we'll be combining small values of one with small values of the other, large values with large values, medium values ... etc., when what we want to do is mix it up.

Think of two distributions as two decks of cards and shuffle both decks. Then, repeatedly taking one card off the top of each deck and matching them, we may get some pairs, but we'll also get all sorts of other combinations. That variety is what we want for simulation so that we get a picture of the full range of possibilities.

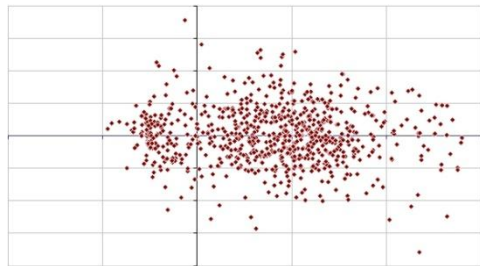
The standard tactic is to shuffle the order of a sample distribution when we create it. That is, we permute the sample vector with a unique *random permutation index*. Since the number of different permutations is the factorial of the number of samples, there isn't much likelihood of an accidental collision – if you're careful with

your random number generation. You need to make sure that the random sequences generated don't repeat.

Sample set sorted	20	30	40	60	70	80
Random Permutation Index	3	2	6	5	4	1
Sample Distribution	40	30	80	70	60	20

In the other case, where the distributions aren't independent, we want to preserve the *coherence* in their rank orders. We still want to randomize them to avoid false correlation with other distributions in the model. The tactic here is to randomize them the same way; we permute them with the same random permutation index.²

The best way to check for correlation is to plot one of the distributions against the other in a scatterplot. If the result is a formless blob, correlation is slight or non-existent. If a pattern emerges, the distributions are probably correlated and you'll need to maintain coherence.



Imagine, in the road-fill example, that the price of fill is somewhat volume sensitive; the price is negotiable and volume can affect the final price. This is not good enough to replace the price with a function of the volume, but for reliable simulation results, we want to keep the connection. In this case, we'd create the two sample distributions together, keeping the historical real-world price/volume pairs aligned. Then we permute them both with the same random index.

² Sam calls a collection of coherent distributions a SLURP: Scenario Library Units with Relationships Preserved.

With very large distributed models, guaranteeing uniqueness in the permutation indexes may become interesting. Careful management and long-period random number generators are needed. This is one of the very few places in Probability Management that the math gets interesting and expert help may be needed. The best known insanely long period generator is the *Mersenne Twister* (See Wikipedia for details). It won't repeat itself for 2^{19937} iterations. It has been implemented on a variety of platforms, isn't too much trouble to manage, and will deliver unique permutations for more than the life of the universe. For really large projects or project portfolio management, it may be worth the effort.

3.3 PM TOOL #3: THE DIST™ STANDARD

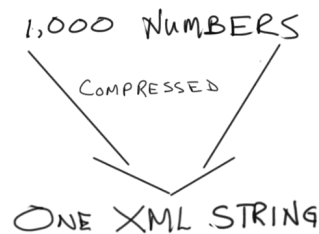
One problem with sample distributions is carrying around all those numbers.

It was to solve this problem that Sam Savage invented DISTs, *Distribution STRings*. A DIST packs a whole sample distribution – hundreds or thousands of numbers – into a single XML text string.

The string for a small DIST looks like this:

```
<dist name="User Interface, weeks"
avg="3.3751" min="2.03" max="7.75"
count="100" type="Double"
origin="DistShaper3 at smpro.ca" ver="1.1"
>G00Z9SIDCIEmC0nYFtMi6R0XKZ+KvSzBI85ui5tMZgo
DlbGtdF1d/CqEMwUlmCfVMMg6oUByUXQyIATsaSw1Qhg
rhOwaaAI9D6oks9M+IDk0XQyIDlI2mhJZBkQXRnm7IR4
5ST3D///IDlgrHDI38VraK2kLownZf41jWw1tROxTsS/
jGRAUJCbwHfwougAAEXRr3A83FQnpnhXukBxM+kswByk
eb0gOQ5RByk83PxtV7mCrH1QQjy6LPGstpgFYRrYKvqZ
9Ez8AAAAA</dist>
```

It's essentially some metadata and a Base-64 encoding of the sample values. The DIST Version 1.1 Standard encoding uses a lossy compression



algorithm that limits precision to between four and five digits.

A variety of software packages, and the Excel workbook that accompanies this document, have code to convert DISTs to arrays and back again. Go to ProbabilityManagement.org for more about the DIST standard and applications that work with DISTs.

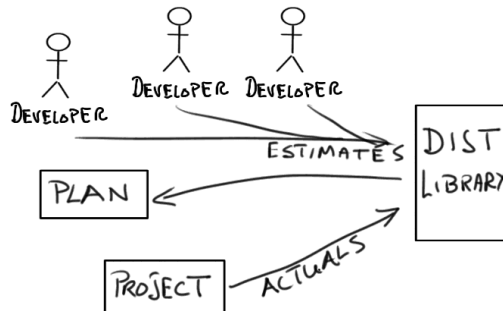
Modeling with DISTs

This is not just a convenient way to store and communicate sample distributions; because it's a string, we can put a whole sample distribution into a single spreadsheet cell, instead of just one number.

With DISTs and a few formulas, we can have a project model with the same structure as a conventional PERT or CPM model. But, when something changes, the spreadsheet calculates hundreds or thousands of possible outcomes – and their probabilities – instead of just one outcome, whose probability is unknown.

Because they're strings, DISTs can be conveniently emailed, embedded in web pages, stored in a library for re-use, or passed around on a thumb drive.

Full-depth synchronous modeling is a thing of the past; you can break up your model and farm out the parts, then collect the results as DISTs whenever and wherever you need them.



Repeatable, Auditable, Anywhere, Forever

You can also use DISTs as a permanent record for re-use in other projects or to meet audit requirements. A DIST's sample values and sequence are permanently preserved in the DIST XML string. This makes the model input and output reusable, repeatable and

auditable, anywhere, forever. You can take your input DISTs and model to a completely different platform and be confident that you'll get exactly the same results.

(*DIST* is a trademark of ProbabilityManagement.org.)

Manage DIST Correlation

The previous section talked about sample distribution order and correlation, and the need to maintain coherence to not lose correlation between distributions. There is, as yet, no DIST standard for grouping related DISTs (the DIST version of a SLURP). The only proposal I know of is to call it a CCD, a *Coherent Collection of DISTs*. That name came up in a teleconference including Sam and a few others from Vector Economics. I like the name; it's suitably descriptive. The format of a file or string containing a CCD awaits a standard, so we're on our own in that respect.

3.4 PM TOOL #4: VISUALIZING UNCERTAINTY

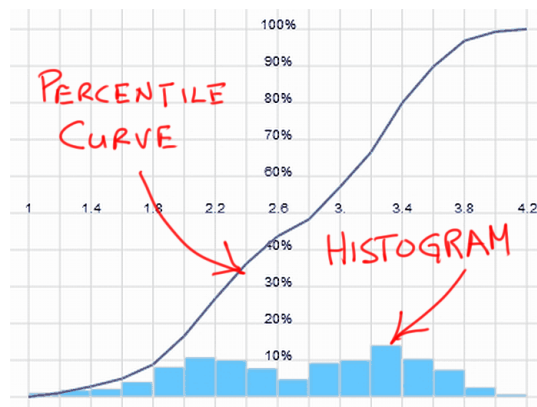
With Bob's 20 samples it was reasonable to simply list the sample values.

However, when we get to large numbers of samples, and interesting models, that's not practical, so we use a convenient visualization.

The graphics can be quite creative – and they should be – but the workhorse is a *probability chart*, something like this:

It has two parts. The *histogram* represents the distribution's *shape*. It plots the probability that the actual is within a particular range.

The *percentile curve* is much more interesting and useful; it plots the



probability that the actual is equal to or less than a particular value. Since both are probability measures, they can be plotted together, using the same axis values.

The percentile curve is the 'meet or beat' line. You can see in the upper right quadrant that there's an 80% probability that the actual will be 3.4 or less. If this is cost in \$millions, and we set the project target at \$3.4 million, there's a 20% chance of an overrun.

Let Management Manage Risk

A percentile curve is the useful answer to "How much?" or "How Long?" It lets management choose the odds that are right for the circumstances. No computer program can make that judgment.

3.5 PM TOOL #5: INTERACTIVE SIMULATION

One of the important features of a useful model is the ability to ask "What if?" – to change assumptions and see the impacts quickly and easily.

Probability Management's fifth tool combines the others with user controls to do interactive project simulation. The next section describes an example, using a small construction project.

4 MODELING WITH PROBABILITY MANAGEMENT

4.1 THE SCENE

A family and their aging grandmother live in separate houses. They plan to build a bigger house, move everybody into it, then renovate the two original houses and rent them out.



They've asked some construction firms for a bid with a fixed price and guaranteed time. They've specified a \$1,000/day penalty if the completion is late.

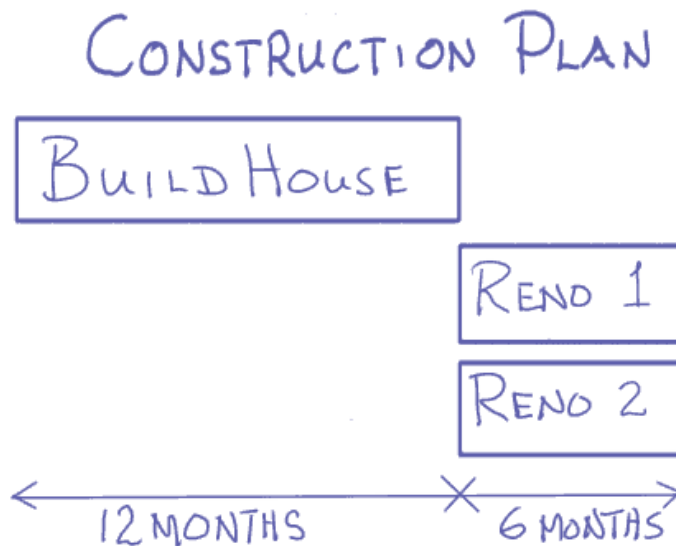
4.2 THE PLAN

We have a lot of history about previous projects and standard costs that we normally use to estimate a job. Our data tells us that, on average, a house with the requested design should cost \$400,000 and take 12 months to build. Also on average, the renovation of each of the existing houses should cost \$200,000 and take about 6 months.

Build House	\$400,000
Renovate 1	\$200,000
Renovate 2	\$200,000
Total Cost	\$800,000
Profit	\$80,000
Bid	\$880,000

We can bid \$880,000, taking home an \$80,000 profit. It looks good.

That's the cost – let's look at the time.



Doing the renovations together, the project should take twelve months for the new house and six months for the two renovations, an elapsed time of 18 months.

4.3 THE FLAWS

You know there's a "gotcha!" in there.

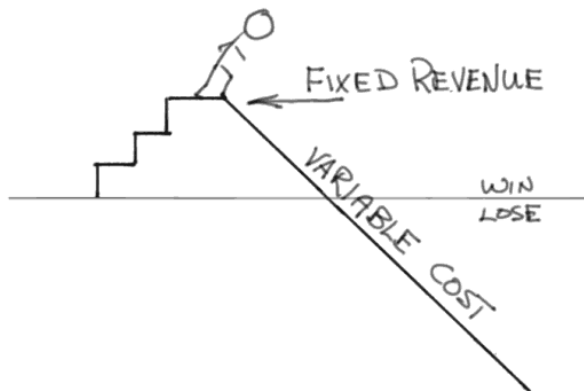
The first flaw in these averages comes with doing the renovations concurrently.

The probability of each renovation finishing in six months or less is around 50%. It's a coin toss; heads it's early, tails it's late.

The probability of both finishing in six months or less is like flipping two heads in a row: only 25%.

The odds are three to one that one or both renovations will be late, the job will be late, and we're paying penalties.

Then, there are two sources of financial imbalance. Both of these bias the odds against making a profit.



The revenue for the job is fixed at the bid price, but the cost of doing it has no intrinsic limit short of bankruptcy. The downside can be a lot bigger than the upside.

If we're late we pay a penalty. If we're early there's no bonus. Over all of the possibilities, on average, we pay a penalty.



4.4 INTERACTIVE MODELING AND DECISION SUPPORT

But we know this isn't the way to work up a bid.

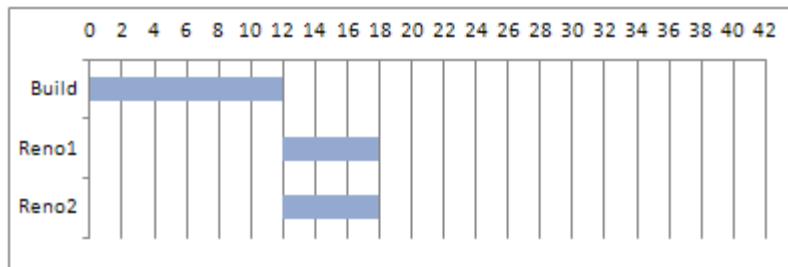
The right way is to build a model and use our sample distributions as input to:

- simulate many variations of cost and time and track the outcomes,
- eliminate the Flaw of Averages in our calculations, and
- make informed decisions about what to bid to make a profit.

If you haven't already, now's the time to open up the Excel workbook (pm4pm.xlsm) on the 'Simulation' sheet.

Let's start with the timeline. We'll simulate doing the project 1,000 times. Each of the thousand trials uses a different combination of completion times for the three tasks, taken from real-world sample distributions. This ensures that each trial outcome is as good a predictor as every other.

Project Timeline Simulation:



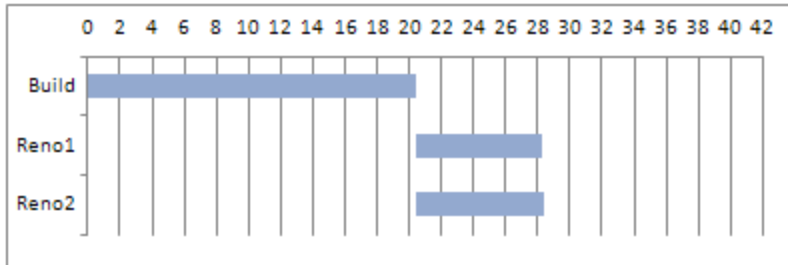
	Build	Reno 1	Reno 2	Total
Months	12	6	6	20
k\$	400	200	200	800

Trial #
0
Average

In this screen shot, the initial Gantt chart shows the averages over all the trials and, as expected, the average time is more than 18 months; the concurrency tax is about two months.

If you've opened the Excel file, the spinner near the bottom right, lets you walk through the trials, seeing the completion time in each case.

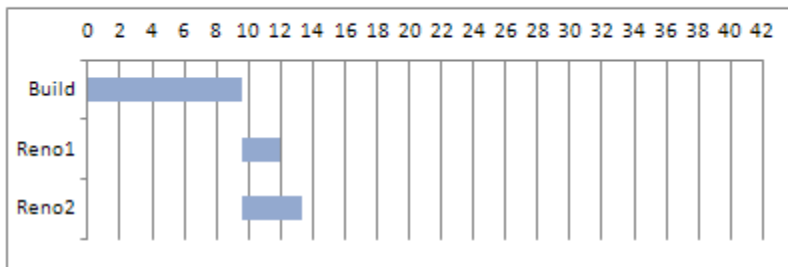
Project Timeline Simulation:



	Build	Reno 1	Reno 2	Total	Trial #
Months	20	8	8	28	9
k\$	449	202	160	811	

Clicking through the trials, we can see the different possibilities. In trial #9, for example, everything is running slow and over budget.

Project Timeline Simulation:



	Build	Reno 1	Reno 2	Total	Trial #
Months	10	2	4	13	10
k\$	257	132	104	493	

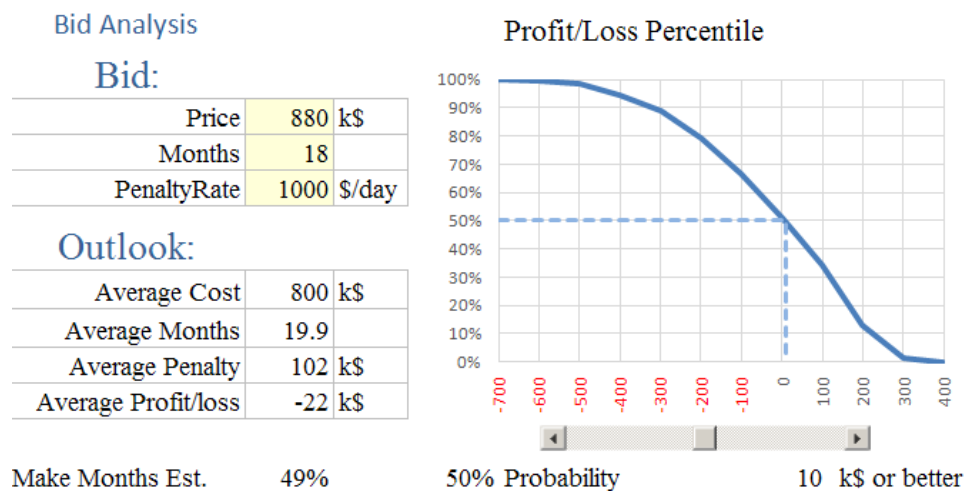
By way of contrast, in trial #10, everything is going especially well.

This display is fine for looking at the simulations, but to make decisions, we need some analysis tools.

4.5 INTERACTIVE WHAT-IF

The decision support tool is a Bid Analysis dashboard that we put together to help us decide what to bid, taking into account both the competitive environment and our risk tolerance. It shows the results of running both the time and cost models with 1,000 variations driven by the real-world data.

This display is driven by just the three tasks in a trivial project plan, but if there were hundreds of tasks, the display would look the same. It presents the essentials without the distraction of plan details.



The inputs to the model (in yellow) are the bid price, promised delivery time in months and the penalty rate – assuming it can be negotiated. Each time we change an input, the simulation gets run again. Thanks to modern technology, that happens in milliseconds.

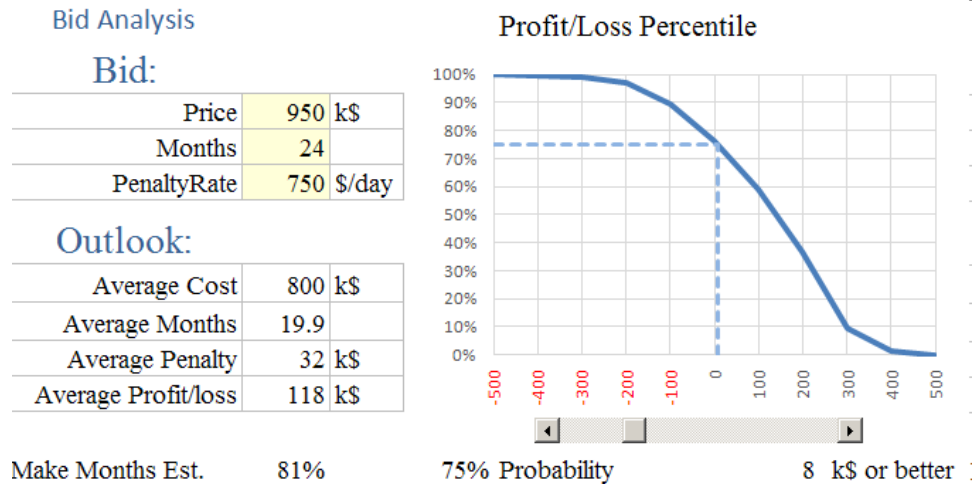
The outputs include the project outlook in averages: time, costs and profit, and the probability of making the time estimate.

As you can see, the outlook is not outlooking good. The late renovation stage is likely to kill us with penalties. The average over all the trials is a two-month overrun and a \$22,000 loss.

But averages throw away information, so ignore that and let's look at the percentile curve on the right.

The first thing to note is that the worst case is a \$700,000 loss but the best case is profit of only \$400,000. That's the 3:1 odds and financial imbalance kicking in. The curve also tells us that breaking even is a coin toss, 50% either way.

We need to give ourselves some headroom. After trying some changes to the bid and checking the outcomes, we could end up with something like this:



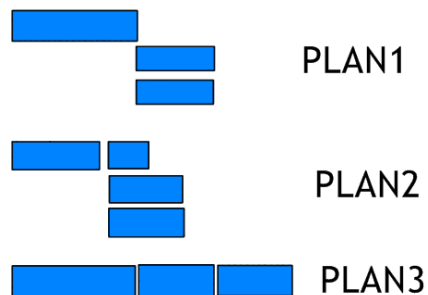
The outlook is decidedly brighter. Assuming we can negotiate the penalty down to \$750/day, the worst case and best case are balanced at \$500k. We're looking at a 75% probability of making a profit and a 50% chance of making as much as \$100k. We've also got a reasonably safe 80% probability of making the time commitment.

A real project would probably have some more input variables, and other relevant outputs, but the display shouldn't be any more complex if there's only one plan contemplated.

Explore alternate plans

The construction plan we used here isn't the only plan possible.

For every project, there are many possible plans, mostly



differentiated by the amount of concurrency in the workflow.

Doing more at once, we finish faster, but with greater risk and cost. This is another risk management decision that should be available to our stakeholders.

Planning a project should include exploring alternate workflows. The kind of modeling tools described here makes it relatively easy.

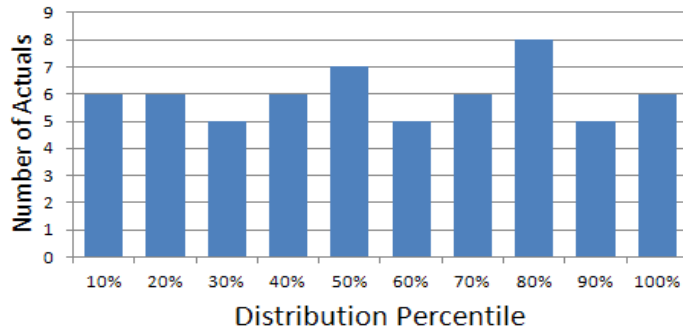
Using a DIST library to input global assumptions, we can model each workflow separately. Then, once again using DISTs, we can aggregate and compare the results in a management dashboard.

5 CALIBRATING ESTIMATORS

No process is complete without some quality feedback.

There is, in general, a measure of the quality of sample distributions: extending Hubbard's calibration approach to distributions, the frequency of events should match the predictions. In other words, the things that the distributions predicted would happen 10% of the time happened 10% of the time, the things that the distributions predicted would happen 50% of the time happened 50% of the time, and so on.

For each actual value, note its percentile rank in the corresponding sample distribution. As the percentile values accumulate, they should be evenly distributed; a histogram of the list should be fairly flat.



If it's not flat, the shape tells us something about the quality of our estimates. If there's a lump on the right, our estimates are too low. A lump on the left means they're too high. A lump in the middle says we lack confidence and a depression in the middle is a sign of over-confidence. This information can be fed back into our estimating approach and used to get more representative sample distributions.

Is that enough? Do you feel lucky? Is this competitive in today's residential construction market? Those are questions the computer can't help us with.

In any given context, there are many considerations that go into developing a budget and schedule for a project. The model output makes it clear that there's a resource/risk tradeoff, and quantifies it. Having a clear indicator of the likelihood of success is vital to an informed decision.

Probability Management is how we manage the odds of success.

If we want our stakeholders to make sound risk management and resourcing decisions, we should show them sample distributions and make sure they understand what they mean.

The Bottom Line

The probability of success – the chance of failure – is a choice you make.

Probability Management can inform that choice, and give you an opportunity to set attainable targets for your projects.

The current version of this document and the accompanying Excel workbook can be found at <http://smpro.ca/ProbMan> .

The Flaw of Averages. Sam Savage
<http://www.flawofaverages.com>

How to Measure Anything. Douglas Hubbard

Uncertainty. M. Granger Morgan and Max Henrion

Probability Management. Savage, Scholtes & Zweidler, OR/MS Today, February 2006, <http://www.orms-today.org/orms-2-06/frprobability.html>

The DIST standard:
<http://probabilitymanagement.org/Dist.htm>

The Flaw of Averages in Project Management. PMI Virtual Library <http://www.pmi.org/en/Knowledge-Center/Knowledge-Shelf/Project-Estimating.aspx>

The Art of the Plan. <http://goodplan.ca>

Marc Thibault

Marc Thibault is an independent consultant with a twenty-year practice focused on technical analysis, design and planning. His clients have included a large fraction of the Canadian federal government's departments and a variety of high-tech companies.

His earlier experience includes over a decade of marketing and technology roles at Xerox, and senior management in two high-tech startups. He has a physics degree from Loyola College in Montreal.

Author of the 'Art of the Plan' blog at goodplan.ca, he's on a mission to fix project planning and the systemic errors that result in high-risk plans and unattainable targets.

Marc Thibault, marc@smpro.ca

Text or phone: 613-724-9442

Skype: marc.thibault256

<http://smpro.ca>