Delphi, Dice and Dominos

Techniques to Understand and Mitigate Technical Risk

David L. Gallop

It is no secret that our programs are becoming increasingly complex and interdependent. With that complexity and interdependency come both opportunity and technical risk. There are three simple techniques any program can use to understand and mitigate technical risk.

These techniques take us beyond the common (yet important) "risk cube." They can be used together or separately. They are designed to avoid three common logic traps: failing to account for additional perspectives; failing to account for uncertainty in data; and failing to account for interdependencies. The techniques are

Gallop *is a professor at the Defense Systems Management College, Fort Belvoir, Virginia. He is a Project Management Professional and a Certified Systems Engineering Professional.*

Consensus is not our goal. Our goal is to understand the uncertainty. This requires diverse perspectives.

commonly used in medicine, finance and manufacturing. The metaphors we use are Delphi, dice and dominos.

Delphi

"Delphi" refers to the Oracle of Delphi, where in ancient Greece someone (perhaps ancient program managers) went to receive prophecies about the future. Program success often depends on how well we forecast the future. We often rely on teams or committees to provide forecasts. While we know groups perform better than their best member, groups have their weaknesses. Powerful individuals (personality or position) can dominate and therefore limit or bias a forecast. The Delphi method seeks to aggregate independent perspectives from a diverse group. You can find Delphi practiced in a variety of fields such as environmental management, tourism, education and marketing. Program managers can use Delphi in situations of complexity, uncertainty and where no hard facts exist. Delphi can be used whenever the program manager needs to analyze disparate perspectives on a subject—contractor incentives, task durations, nonmateriel impacts, etc. In the following example, we will use Delphi to help us evaluate a Technology Readiness Level (TRL).

TRLs are an important part of the technical risk assessment process. TRL scores measure the maturity of a technology on a scale of 1 through 9 (with 9 being the most mature). TRLs should be based on objective evidence instead of opinions. However, even the best available evidence can be incomplete and subject to interpretation. The Delphi technique can help us increase our knowledge and provide different perspectives. The classic Delphi technique involves using multiple rounds of feedback to drive the group toward consensus. This can be time and resource intensive. Classic Delphi also has been criticized for achieving consensus at the expense of the best ideas. We will use Delphi's power to extract knowledge from experts while avoiding the mentioned disadvantages. Figure 1 summarizes our modified Delphi process.

Figure 1. Modified Delphi Process

Steps	Actions
1	Select the problem or topic for investigation.
2	Assemble the evidence/information file for the Delphi panel members.
3	Select the Delphi panel members.
4	Send the Delphi panel members evidence/information file to be reviewed independently.
5	Receive and analyze the findings from the Delphi panel.

Once the problem or topic area is identified, assemble the evidence file. This includes the objective data that will underpin the TRL score. The file should also contain information regarding the operational and systems context. To avoid confirmation bias, the file should not include an expected TRL score. The file should have plenty of "white space" to elicit information and opinions from the panel members. Possible questions include the following: What additional information should we seek about the subject technology? Where and how can we get that information? What laws, regulations and policies must we consider? What stakeholders will have an interest in this technology? What are the pitfalls and unintended consequences of this technology? What are some materiel and nonmateriel alternatives to this technology? What new ideas and opportunities will this technology offer? Finally, the file should ask the panel members to provide a TRL score and its justification.

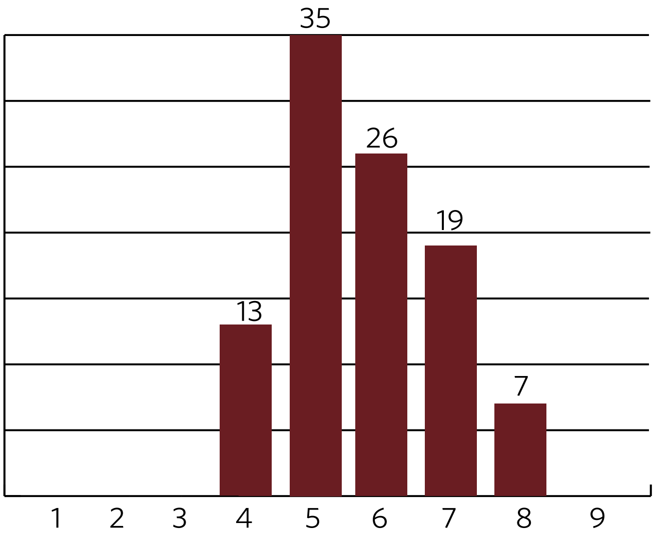
Continuing our TRL evaluation example, electing the Delphi panel members and distributing the evidence files to those members are the next steps. Conventional wisdom would have us seek out the most brilliant minds in the domain of the subject technology. This would be the "perfectly, perfect" panel and help drive panel members toward a consensus. But consensus is not our goal.

Our goal is to understand the uncertainty. This requires diverse perspectives. We should seek the "perfectly, imperfect" panel. Panel members should have collective expertise in the operational environment, defense acquisition process, technology development and the specific technical domain.

Figure 2. TRL Scores

	TRL-1	TRL-2	TRL-3	TRL-4	TRL-5	TRL-6	TRL-7	TRL-8	TRL-9
# of Delphi Panel Members	0	0	0	2	6	14	7	1	0

Figure 3. Monte Carlo Simulation



Delphi panels range in size from 10 to more than 1,000. The Delphi process does not call for the panel size to be representative samples for statistical purposes. However, 30 panel responses may be optimal in most cases. Each panel member responds individually and independently. If we were seeking consensus, we would perform the Delphi in many rounds. However, two rounds (initial feedback and follow-up clarification) should suffice. The process allows panel members the freedom to think reflectively and propose alternative viewpoints.

Finally, collect and analyze the findings. You are looking for new information, risks, stakeholders, ideas and opportunities. The increased knowledge and perspective from the Delphi method can lead to technical excellence and innovation.

Dice

Program managers frequently must assess quantifiable data—i.e., numbers. They often reach conclusions and make decisions without considering the uncertainty in the numbers. A Monte Carlo method typically runs a simulation many times to obtain the distribution of an unknown probabilistic entity. The Monte Carlo technique can be used in many situations such as cost estimates, corrective maintenance task durations or risk ratings. Our example uses the TRL scores from our Delphi process to assess the uncertainty.

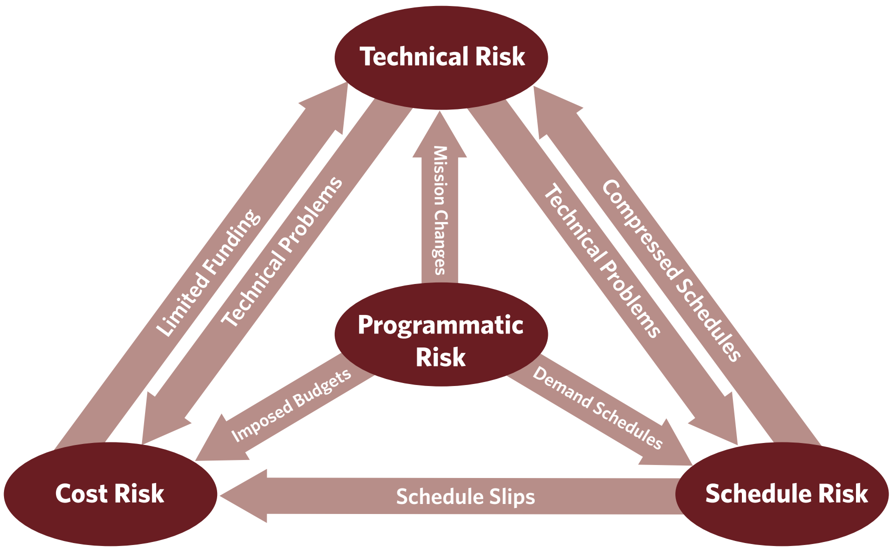
Figure 2 is a table of the TRL scores from our notional Delphi panel. We can see that the scores range from a pessimistic TRL-4 to an optimistic TRL-8. The most common score is TRL-6. To arrive at a single score, we could simply take the average of the scores. The problem is that decisions made based on the average are wrong on average ("The Flaw of Averages" by Sam L. Savage, John Wiley & Sons publishers). Since we are interested in understanding the uncertainty, we should consider the probability distribution. It's time to roll the dice using a Monte Carlo simulation.

There are many powerful commercial software packages that will perform Monte Carlo simulations. These tools can be expensive and often require training. Program managers need not spend a great deal of money on tools, training or consultants. Excel is widely available and has built-in features to create a simulation. The Web has a number of articles and videos that provide detailed instructions on how to construct a Monte Carlo simulation in Excel. Thomas and Linda McKee's article, "Using Excel to Perform Monte Carlo Simulations," in the December 2014 issue of *Strategic Finance,* is a great resource.

We created a simulation in Excel that ran 100 iterations (Figure 3). In that simulation, we assigned random probabilities that the actual TRL lies somewhere between the minimum (TRL-4) and maximum (TRL-8), with a greater probability around the mode—or most frequent—(TRL-6). Many random processes follow a normal (bell-shaped) distribution, but some do not. Since we have a minimum, maximum and a mode, we used a triangular distribution. The McKee article mentioned earlier describes the steps to create several common probability distributions in Excel.

This simulation reveals a strong probability that the technology maturity actually is TRL-5. If we assume that the program manager expected the maturity to be TRL-6, there should be mitigation commensurate with the risk that the maturity actually is TRL-5. The mitigation could include gathering more information or exploring alternative solutions. So far, we have

Figure 4. Risk Interrelationships



Adapted from the INCOSE SE Handbook *3.3.2, page 222.*

Our goal is to keep all the dominos standing ... but we never have the resources to monitor and control everything equally.

used the Delphi method to consider perspectives and the dice (Monte Carlo simulation) to consider uncertainty. Our next tool will allow us to consider interdependencies.

Dominos

Consider all the risks in a program. We can group those risks into four categories—cost, schedule, technical and programmatic. Figure 4, adapted from page 222 of the International Council on Systems Engineering's 2011 *Systems Engineering Handbook* (INCOSE-TP-2003-002-03.2.2), shows the typical relationships among the risk categories. Visualize these risks as groups of dominos standing on end. One domino falling may trigger a cascade of issues across the program. Our goal is to keep all the dominos standing ... but we never have the resources to monitor and control everything equally. It also is difficult to predict what other dominos will fall if a domino tips over. If only we had a tool that would help program managers focus their resources and predict where downstream issues could occur from a falling domino. An N-squared chart can help.

An N-squared chart can map interdependencies of functions, components, documents, organizations and budget lines ... just about anything that can be decomposed into smaller units and where those smaller units have exchanges. To build an N-squared chart, put the elements (to be designated by the letter "E" and given identifying numbers) in the diagonal blocks of a matrix. The exchanges between the two elements appear in the intersection of the corresponding row and column. In the generalized example in Figure 5, E1 receives an external input and an input from E2. E1 provides an external output and an output to E2.

Let's return to our domino analogy. If the E2 domino falls, the E1 domino could fall as well. The N-squared chart's predictive power increases as we provide more detail on the interdependencies. From Figure 6, we can infer that E1 and E4 deserve our attention. E1 is the interface with an external system, outputs data for three elements, and receives inputs from five elements. E4 receives four inputs from three elements. We also can see that three elements depend on Data Item A. Data Item A also deserves our attention. The fact that E1 outputs Data Item A underscores the importance of the E1 domino remaining upright. As a predictive tool, a performance shortfall in E6 would predict a performance shortfall in E1 (and in turn shortfalls in E2, 3, 4, and the external system). All the dominos are important ... but some dominos are more important than others.

Conclusion

Understanding and mitigating technical risk requires casting a wide net for information and perspectives. The Delphi technique is useful in complex situations where there may be no clear choice. Monte Carlo simulation highlights the uncertainty in data. Knowing this uncertainty allows communication of the confidence in the data. The N-squared chart maps the interdependencies so we focus our monitoring and control.

The author can be contacted at ***david.gallop@dau.mil****.*

Figure 5. The Elements Interact

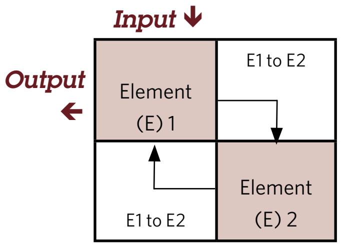


Figure 6. N-Squared Chart

Input ↓

Output ←

Element 1	Data Item A Data Item B	Data Item A	Data Item A		
Data Item C	Element 2		Data Item D		
Data Item L		Element 3	Data Item E Data Item F		
Data Item K			Element 4	Data Item E	
Data Item J				Element 5	Data Item H
Data Item I					Element 6