Boeing Technical Journal

Monte-Carlo Simulation of large-scale complex life-cycle product patterns for Program Management

*Scott D. Button, Joshua W. Brown, Tek D. Kim, and Thomas E. Sherer

***Abstract* –Monte-Carlo analysis of threads enables functional managers to understand the expected completion distribution of their function, in the context of the overall program. This provides the means for program managers and engineering personnel to understand the role of variation and uncertainty in the completion distribution for large-scale complex life-cycle product patterns for Program Management. This understanding is central to meeting the flow-time and cost improvement necessary to remain competitive.**

***Index Terms* – Affordable Complex Systems Integration (ACSI), Critical Chain Project Management (CCPM), Process Based Program Planning (PBPP), Lean+ Systems Integration Management (LSIM), Theory of Constraints (TOC), and Monte-Carlo Methods.**

I. Introduction

Cognitive bias unfavorably affects how we make decisions under uncertainty (Kahneman, 2011). This is exacerbated as the scale and complexity of the underlying system increases. Methods are needed to provide decision analysis support for program managers, resource managers, and other engineering personnel faced with allocating scarce resources for large-scale, complex product development. The approach described in this work extracts a life-cycle thread from an overall product development network, uses Monte-Carlo analysis to simulate completion distributions for selected resource allocation decisions options, then summarizes the trade-offs between the resource allocation choices. Expected benefits are reductions in the cost and makespans of programs which contain significant product development.

II. Background

Lean+ Systems Integration Management (LSIM) is a set of tools and methods that have increased productivity and improved affordability of Product Development. LSIM is when modelers work with Subject Matter Experts (SMEs) to develop analytical data-driven models of program work statement. These models are produced using the Constructing Processes around Data (CPAD) software which enforces rigorous rules of integration, and then are exported to Critical Chain Project Management software for execution by program managers and resources.

The principle objective of LSIM is to perform the right work in the right order which will reduce unplanned and out-of-sequence rework. This model-based approach improves the

quality of deliverables, increases throughput, facilitates cross-functional integration, and provides management a predictive model, with regular feedback on focus areas requiring their attention. For further detail on LSIM, see (Button, Garcia, Guy, Holt, & Olenginski, 2014) and (Button S. D., 2017).

In 2015, we completed an Internal Application Development (IAD) project to develop life-cycle threads from our integration models. The context of a thread is the life-cycle maturity of information for a few functions, e.g. how Flight Controls and Stability and Control interact to mature information from conception to deliverable. These thread features enable the use of statistical clustering to extract a life-cycle pattern of products and inputs, and to apply this life-cycle pattern to other products to establish similar patterns of inputs, without manually selecting these inputs. This has the potential to significantly speed up our modeling process.

In 2016, we enhanced these features to provide network diagram plots for use in SME reviews. Features include the ability to add to a thread from a tree view of the product hierarchy, produce a network diagram with nodes and edges colored according to responsible organization, and to populate the nodes with additional information including remaining duration, reviewer name, and notes. In 2017, we added a Monte-Carlo analysis of these threads. This enables functional managers to understand the expected completion distribution of their function, in the context of the overall program. This provides the means for program managers and engineering personnel to understand the role of variation and uncertainty in the completion distribution for large-scale complex life-cycle product patterns for Program Management. This understanding is central to meeting the flow-time and cost improvement necessary to remain competitive.

Monte-Carlo Analysis is a simulation method named after the famous Mediterranean gambling resort. To conduct a Monte-Carlo simulation, a sample is drawn from a defined distribution representing each task. Each sample from the defined distribution constitutes a trial, and 100-1,000,000 trials are typically evaluated to determine the results. The mainstream approach to simulation of project duration is by assuming a shape for the task distributions, then aggregating the tasks according to the project network to form a project completion distribution.

Analytical methods to estimate completion distribution are available as well. Dodin showed how to find the upper bound for project completion time from below (Dodin, Jul. - Aug., 1985). Sahner and Trivedi showed an approach combining directed acyclic graphs hierarchically (Sahner & Trivedi, 1987). Chu et. al. showed a modified label-correcting tracing algorithm to determine Critical Path in stochastic networks (Weng-Ming Chu, 2014). Button, Sherer and Kim showed a static analysis using Kirchhoff's circuit laws (Button, Sherer, & Kim, Method and Systems for Determination of Criticality Index For The Development of a Structural Product, 2016).

These analytical methods can reduce the time and effort needed to develop and run an analysis. However, as discussed in Chu (Weng-Ming Chu, 2014), a Monte-Carlo is considered the gold standard and is used to validate analytical methods. The need to validate the authors' analytical methods is what drove the development of this particular Monte-Carlo simulation implementation.

III. METHODOLOGY

Malcom, Roseboom, Clark, and Fazar (D.G. Malcom, 1959) used the beta distribution to represent the probability distribution of the time expected to perform an activity. They reasoned that the needed distribution would have a peak (the most probable time estimate) with small probabilities associated with an optimistic and pessimistic time estimate. This required gathering three numbers from the estimator, the optimistic, pessimistic, and most likely. Hagstrom (Hagstrom, 1990) used a three-point distribution, with a probability of 0.6 at the mean, 0.2 a selected random distance above the mean, and 0.2 for being the same selected random distance below the mean. Hoel and Taylor (Hoel, 1999) assumed a normal distribution for activity times, stating that "...the normal probability distribution is both familiar and reasonable." The (Project Management Institute, 2000) states that common distribution types for risk assessment include the uniform, normal, triangular, beta, and log normal. (Leach, 1997) stated that the log normal distribution often describes distributions of activity times. For this Monte-Carlo Analysis, we have selected the log normal distribution to use as the shape of the distribution for task duration. We favor the log normal distribution as it has the right shape as described by Malcom, Roseboom, Clark, and Fazar, and only requires the resources to estimate the average duration and validate our assumption that the safe duration should be two standard deviations longer than the average duration. This allows us to form the curve for the distribution of task durations with minimal burden on our resources.

To draw a sample from the lognormal distribution, we use the following from Wikipedia: (Log-normal distribution, 2017)

"Given a log-normally distributed random variable X and two parameters μ and σ that are, respectively, the mean and standard deviation of the variable's natural logarithm, then the logarithm of X is normally distributed, and we can write X as

$$X = e^{(\mu + \sigma Z)} \qquad \text{(Equation 1)}$$

with Z a standard normal variable.

In Excel, the Z normal can be computed as

=NORMINV(RAND(),0,1). (Equation 2)

This computed Z normal in Equation 2 may be substituted into the Equation 1 for each sample of duration.

The probability density function for μ=0 and various values of σ can be seen in Figure 1, taken from (Log-normal distribution, 2017). There is an interesting discussion in (Ginos, 2009) which shows for values of σ less than 0.3, the lognormal shape is very close to the normal distribution shape.

Additional PDF's of log-normal, along with a comparison of methods for parameter estimation may be found in (Ginos, 2009). Figure 1 was helpful for the authors to understand how to adjust location and scale parameters to obtain the desired Probability Density Functions to represent the completion distributions for tasks.

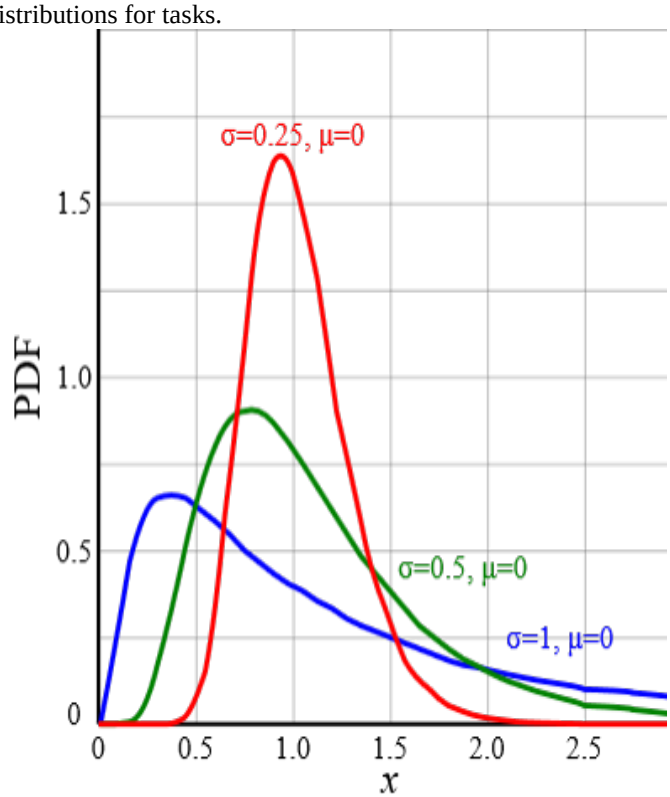

Figure 1, Log-normal Distribution

An obstacle to simulating a task in a project is understanding how to adapt the log-normal distribution with the appropriate location μ and scale σ parameters to represent a task in our project with an estimated duration. Given that the location μ is the mean of the natural log of the variable, and σ is the standard deviation of the natural log of the variable, this is not straightforward.

These location and scale parameters may be computed as described in (Log-normal distribution, 2017), excerpted as Equation. 3. "m" is the mean, and the variance "v" is the square of the standard deviation. Equation 3 is very helpful to convert between normal and log-normal space.

$$\mu = \ln\left(\frac{m}{\sqrt{\left(1+\frac{v}{m^2}\right)}}\right), \sigma = \sqrt{ln\left(1+\frac{v}{m^2}\right)}. \qquad \text{Equation 3}$$

IV. Thread SME Review Calculations:

Criticality Index, or "criticality" is the probability that a task will be on the critical path (Ghomi, 2002). The Thread Monte-Carlo calculates criticality with a location parameter, μ of the natural log of the estimated remaining duration, and a scale parameter, σ of 0.3125. This scale factor was selected by the authors to provide a reasonable distribution for duration.

Using Java, we assign a log-normally distributed random value of duration to each task in the thread. For those who would like to reproduce our results, the Java command is as follows: (Sherer, 2017)

monteCarloProduct.setDuration(Math.exp(monteCarloProduct.getnaturalLogOfDuration() + (sigma * random.nextGaussian())));

In pseudocode, this is:

$Task\ Duration = e^{(\mu + \sigma Z)}$, where "μ" is the natural log of the duration and "z" is the next Gaussian. The next Gaussian is the next normally distributed double-precision floating point value with mean 0.0 and standard deviation 1.0 from the random number generator's sequence (Knuth, 1998).

Figure 2 illustrates a network thread for the 787 Semi-Levered Landing Gear. This is a thread excerpted from a larger, airplane-level integration model. The curve above the first two tasks in the thread represents the log-normal distribution of durations for each task. For a full-size, 36-inch by 91-inch PDF of this figure, please see (Button S. D., 2017).

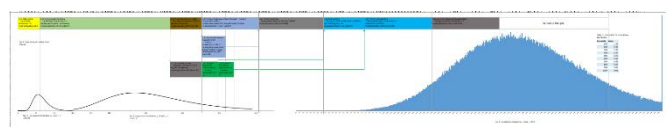

Figure 2, Semi-Levered Landing Gear Thread

Figure 3 shows the probability density function for task completion for an estimated task durations of 61 days. Note the log-normal shape (long right tail).

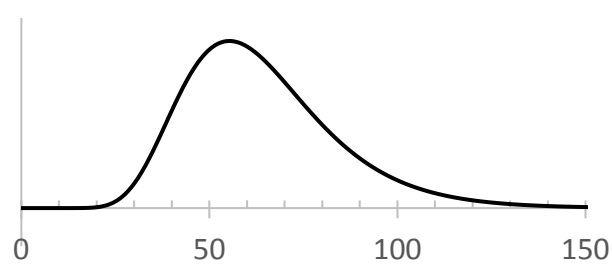

Figure 3, Log-normal Distribution, μ =ln(61), σ =0.3125

Figure 4 shows the probability density function for task completion for an estimated task durations of 353 days.

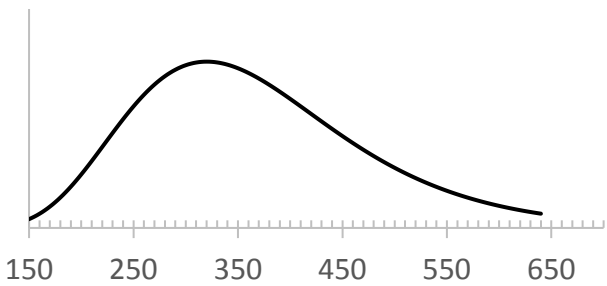

Figure 4, Log-normal Distribution, μ =ln(353), σ =0.3125

For each task in the network, a random sample is taken from the probability density function representing that task. A finish time is computed for each task, which is the sum of the sampled task duration and the finish time of the latest predecessor to that task. The finish time of the terminal product in the network represents the finish time for the thread, for one run of the Monte-Carlo analysis. This is repeated numerous times (1,000 to 100,000) to form the completion distribution of the thread. Figure 5 shows the histogram for the completion distribution of the terminal product in the thread. Note the completion distribution shifts towards a more normal distribution, due to effects predicted by the Central Limit Theorem.

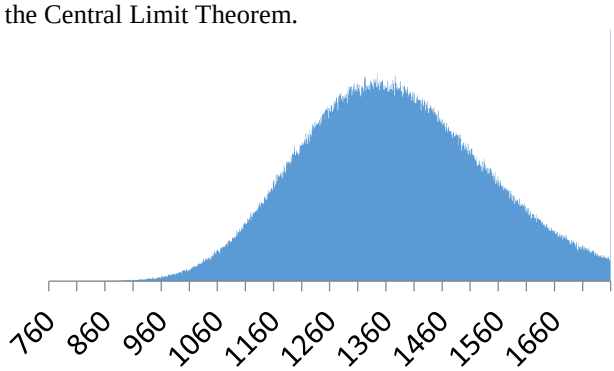

Figure 5, Completion Distribution, Mode = 1370

Table 1 shows the percentile values for the completion distribution of this simple project. Note that the apparent end of the project, provided by the sum of the durations along the critical path is at 1316 days, however the 50% probability duration is 1370 days. If estimates of duration for a program correspond to actual average durations, and the program has paths in parallel, the parallel paths act as a high-pass filter, causing the project completion distribution to shift to the right and shift towards log-normal. This effect is typically somewhat mitigated by resources giving safe, rather than average estimates of duration. For further discussion on these effects, see (Goldratt, 1997).

Duration estimates used directly to form the tasks described here adopt the view that resources are dedicated to completion distributions from which task finish times are sampled. For a multi-project view of simulating the periodic output of resources, then allocating the resource output to tasks across projects (Button S. D., 2003).

Table 1: Percentiles for Completion Distribution

10%	1169
20%	1234
30%	1283
40%	1327
50%	1370
60%	1414
70%	1463
80%	1523
90%	1612
100%	2951

Appendix 1 illustrates the level 4 network thread for the 787 Semi-Levered Landing Gear in the context of the larger Commercial Aircraft model. We selected a rather simple, high-level thread to clearly illustrate the trade-offs between resource consumption and project completion.

V. Process Improvement Trade Study Analysis

This study started with the premise that it was known, for three selected tasks, that it was possible to double the resources on the 1st task, which is expected to cut the duration in half. As a trade-off, resources on the 2nd and third task could be cut in half, doubling the duration. Such conditions may not be typical in reality, for example Brooks points out that "adding human resources to a late software project makes it later" (Frederick P. Brooks, 1995 [1975]). However, these trade-offs illustrate the overall project effect of reallocating finite resources to change individual task durations. Given those conditions, the question we wish to answer is: "What is the effect these trade-offs, in isolation and in combination?"

The analysis is performed by first forming all possible combinations of the tradeoffs. The possible combinations in the study are shown in the first column of the table in Appendix 1. The first three rows represent the three tasks taken in isolation. The next three rows represent the two-factor combinations, and the last row represents the combination of all three effects. For each row in the table, a Monte-Carlo analysis is run on the network with the conditions modified according to the row entries. Average thread duration is simulated with the Monte-Carlo, and used for comparison between different combinations. The optimal combination from this study is highlighted in Appendix 1 with blue ellipses.

The network diagram is shown as Figure 8: Landing Gear Network Diagram. The upper number is the URI identifier, the lower identifier is the duration in days. The three tasks involved in the trade-off are indicated with an ellipse.

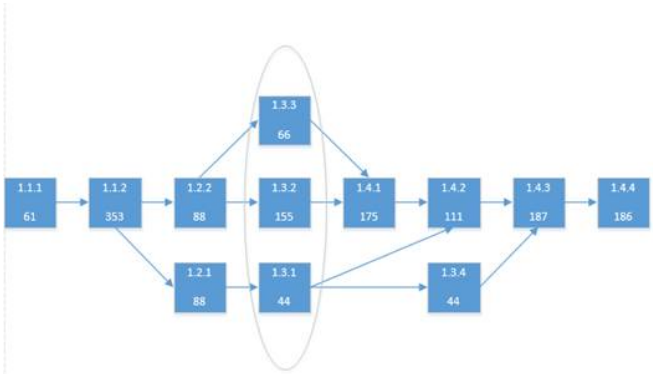

Figure 8: Landing Gear Network Diagram

We find it rather interesting that by moving resources from 1.3.1 to 1.3.2, the project duration is reduced approximately the same amount as if you doubled the resources on 1.3.2 in isolation. This is due to sufficient slack in the non-critical path feed from 1.3.1 downstream. Although this effect is somewhat obvious for this simple network, this technique will provide non-obvious results for more complex networks.

VI. Further Work

Replacing the Monte Carlo analysis with an analytical approach has long been a goal of the authors. This will improve the response time of the Process Trade Study Analysis tool, and make it feasible to evaluate more combinations and larger networks. The literature review conducted during the development of this article will help move that effort forward. The development of the large LSIM networks which are used as an input to the trade study analysis tool requires a lot of effort, modeling aptitude, and some time from busy SMEs for interviews and reviews. Improvements resulting in faster and cheaper development of LSIM networks are needed.

Another major area for development is to get better information around task estimates. In product development, there are many uncertainties which make it difficult to estimate the duration of a task. Often these estimates are required years in advance before the task will be performed. In addition, the actual conditions including task scope, resources assigned, skill levels, resource availability and quality of precedence inputs is unknown. In 2007, a task was identified to research accuracy with respect to task effort. In 2012, Reference Class Forecasting (Kahneman, 2011) was identified as a possible solution direction.

VII. Conclusion

Monte-Carlo Analysis methods are useful to demonstrate the trade-offs between time and resources for lifecycle threads in a commercial aircraft program. This provides the means for program managers and engineering personnel to understand the role of variation and uncertainty in the completion distribution for large-scale complex life-cycle product patterns for Program Management. This understanding is central to meeting the flow-time and cost improvement necessary to remain competitive.

Acknowledgments

The authors would like to acknowledge the LSIM modeler whom developed this network, Thanh-Thuy T. Huynh.

Also, James Karr, Ron Engelbeck, and Kumar Bhatia, who all asked questions related to the value of decreasing duration in an LSIM network.

The Associate Editor and anonymous peer reviewers from the Boeing Technical Journal provided excellent critical comments, which enabled substantial improvements to this manuscript. We gratefully and sincerely thank the reviewers for critically reading and commenting on this manuscript.

References

Button, S. D. (2003). Project duration prediction using a Monte Carlo simulation of the periodic output of the project resources. *Monte Carlo Methods and Applications (MCMA)* 9 (3): 217 - 225.

Button, S. D. (2017). Monte Carlo simulation of large-scale complex life-cycle product patterns for program management. *Boeing Technical Excellence Conference, 24th (BTEC 24), Track J: Exhibits, Charleston, South Carolina*.

Button, S. D. (2017). Critical chain at Boeing - We put the "+" in "Lean+". *Boeing Technical Journal 7 (2). (http://catalog.web.boeing.com/record=b2870311)*

Button, S., Garcia, J., Guy, T., Holt, S., and Olenginski, S. (2014). *Lean systems integration management (LSIM) concept of operations for the product development life cycle.* Boeing Company, D6-84932 (April 25, 2014).

Button, S., Sherer, T., and Kim, T. (2016). Method and systems for determination of criticality index for the development of a structural product. *Invention Disclosure 16-1256*. Everett, WA : Boeing Company Invention Disclosure.

Malcom, D. G., Roseboom, J. R., Clark, C. E., and Fazar, W. (1959). Application of a technique for research and development program evaluation. *Operations Research* 7: 646 - 669.

Dodin, B. (1985). Bounding the project completion time distribution in PERT networks. *Operations Research* 33 (4): 862 - 881.

Brooks, F. (1995 [1975]). *The Mythical man-month: essays on software engineering.* Boston: Addison-Wesley.

Fatemi Ghomi, S. M. T. and Teimouri, E. (2002). Path critical index and activity critical index in PERT networks. *European Journal of Operational Research* 141 (1): 147 - 152.

Ginos, B. F. (2009). *Parameter estimation for the lognormal*. M.S. thesis. Brigham Young University, Provo, Utah.

Goldratt, E. M. (1997). *Critical chain.* Great Barrington, MA: North River Press.

Hagstrom, J. N. (1990). Computing the probability distribution of project duration in a PERT network. *Networks: an international journal* 20 (2): 231-244 [237].

Hoel, K. and Taylor, S. G. (1999). Quantifying buffers for project schedules. *Production and Inventory Management Journal* 40 (2): 43 - 47 [43].

Kahneman, D. (2011). *Thinking, fast and slow.* New York: Macmillan.

Knuth, D. E. (1998). *The Art of computer programming, volume 3: seminumerical algorithms.* New York: Pearson Education.

Leach, L. P. (1997). *The Critical chain project managers fieldbook.* Idaho Falls, ID: Quality Systems.

Log-normal distribution. (n.d.) In *Wikipedia*. Retrieved January 8, 2017, from https://en.wikipedia.org/wiki/Log-normal_distribution.

Project Management Institute. (2000). *Guide to the project management body of knowledge*. Newtown Square, PA: PMI.

Sahner, R. A., and Trivedi, K. S. (1987). Performance and reliability analysis using directed acyclic graphs. *IEEE Transactions on Software Engineering* SE-13 (10): 1105 - 1114.

Sherer, T. E. (2017). *Threaded lifecycle editor Java code*. Everett, WA: Boeing Company.

Chu, W. M., Chang, K-Y, Lu, C-Y, Hsu, C-H, Liu, C-H, and Hsiao, Y-C (2014). A new approach to determine the critical path in stochastic activity network. *Mathematical Problems in Engineering* 2014: 1 - 13.

BIOGRAPHIES

Scott Button: A former US Navy officer, leadership consultant, and entrepreneur, Scott worked in the Boeing Interior Responsibility Center from 1989-2000, then re-joined Boeing in 2005.

He was selected as an Associate Technical Fellow in 2007, with his areas of expertise including Large Scale Systems Integration and Theory of Constraints.

His current assignment is in Airplane Level Engineering Integration (ALEI), contributing to Lean+ Systems Integration Management Core.

Josh Brown: Joined The Boeing Company in 2008, and has worked in roles in both BDS and BCA, including Electrical Installation, Affordability, Quality Engineering, and Systems Engineering. He has a B.S.E. in Aerospace Engineering from the University of Michigan, and an M.Eng. in Systems Engineering from Cornell University.

Tek Kim: A former entrepreneur, Tek hired into Lean Systems Engineering Management in 2007 as a Programmer/Analyst. He has a B.S. in Computer Science from the University of Washington, and is bilingual in English and Korean.

Tom Sherer: Joined The Boeing Company in 1996 as a Software Engineer, having previously developed engineering, econometric and scientific applications for the environmental, nuclear and utility industries. Tom received B.S. degrees in Geology, Mathematics and Computer Science from WWU in 1979, and an MS degree in Software Engineering from RPI in 1984.

APPENDIX

Appendix 1 – Tabulated Values of Monte-Carlo Process Improvement Trade Study.

Task URI	Criticality, (%)	Original Duration (Days)	Change in Cost (Resource Allocation)	Modified Duration (Days)	Task Duration Change (Days)	Original Project Duration (Days)	Modified Project Duration (Days)	Project Duration Change (Days)	Change in Resource Cost, $M	Project Duration Change (%)
1.3.2	97.32%	155	2X	77.5	-77.5	1382	1308	-74	$2X	-5.35%
1.3.3	2.68%	66	1/2X	132	66	1382	1398	16	$1/2X	1.16%
1.3.1	0.00%	44	1/2X	88	44	1382	1383	1	$1/2X	0.07%
combination [1.3.2, 1.3.3]						1382	1360	-22		-1.59%
combination [1.3.2, 1.3.1]						1382	1309	-73		-5.28%
combination [1.3.3, 1.3.1]						1382	1398	16		1.16%
combination [1.3.2, 1.3.3, 1.3.1]						1382	1361	-21		-1.52%