INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

Intl. Trans. in Op. Res. 20 (2013) 515–531
DOI: 10.1111/itor.12023

Integrating probabilistic design and rare-event simulation into the requirements engineering process for high-reliability systems

Kristin Graham Saling[a] and K. Preston White, Jr.[b]

[a] *Department of Systems Engineering, United States Military Academy, West Point, NY 10996, USA*
[b] *Department of Systems and Information Engineering, University of Virginia, Charlottesville, VA 22902, USA*
E-mail: kcg9g@virginia.edu [Saling]; kpwhite@virginia.edu [White]

Received 14 March 2013; received in revised form 20 March 2013; accepted 21 March 2013

Abstract

Early in a program, engineers must determine requirements for system reliability and availability. We suggest that existing techniques gathered from diverse fields can be incorporated within the framework of systems engineering methodology to accomplish this. Specifically, adopting probabilistic (Monte Carlo) design techniques allows the designer to incorporate uncertainty explicitly into the design process and to improve the designer's understanding of the root causes of failures and how often these might realistically occur. In high-reliability systems in which failure occurs infrequently, rare-event simulation techniques can reduce the computational burden of achieving this understanding. This paper provides an introductory survey of the literature on systems engineering, requirements engineering, Monte Carlo simulation, probabilistic design, and rare-event simulation with the aim of assessing the degree to which these have been integrated in systems design for reliability. This leads naturally to a proposed framework for the fusion of these techniques.

Keywords: simulation; reliability; engineering; uncertainty modeling; risk analysis; performance evaluation

1. Introduction

Advances in computing technology have led to a dramatic shift in the way engineers view the development and behavior of systems. Computer simulation has enabled us to better study the behavior of complex problems in depth. At the same time, it offers a vast array of new network problems, queuing systems, reliability and failure studies, and other problems. A multitude of overlapping yet distinct areas have been developed from this research, including probabilistic design (PD), probabilistic requirements, rare-event analysis, discrete-event simulation (DES), and rare-event simulation, each of which includes a variety of subareas and methodologies.

These methodologies, if used separately, have enabled systems engineers to take a closer look at many elements of more complex problems than previous methodologies. If used in combination, these tools and techniques can be even more powerful. Systems engineers are already using these techniques in tandem, but there exists no cohesive methodology for applying multiple techniques from PD and rare-event analysis to solve complex problems.

This leads us to the following questions: (1) To what degree have engineers used their understanding of rare events and rare-event analysis to improve their understanding of system failures in high-reliability systems? (2) To what degree can we use simulation, rare-event analysis, and PD to improve the requirements engineering process, individually or in conjunction with design of experiments (DOE) and supercomputing?

In order to address these questions, we first review the foundations of the systems engineering process, PD, and rare-event analysis. We then demonstrate where in the systems engineering process both techniques can be applied and how this application can improve the understanding of the system and its behavior.

2. Existing literature

The existing literature covering systems engineering, requirements engineering, DES, PD, and rare-event simulation is very diverse in nature. Although numerous papers within these fields refer to techniques used in the other areas, no cohesive literature currently ties all these processes together. As suggested in Fig. 1, the various disciplines and the literature within these processes overlap in some areas while leaving significant gaps in others. Although many of the sources address several areas, no existing source provides a cohesive flow of the entire process. Significant literature outlining the PD process exists in the mechanical engineering realm, integrating some reliability engineering concepts. The works of Safie (Safie, 1992, 2002; Safie and Weldon, 2001) and his colleagues in NASA are presently the most comprehensive publication, combining elements of PD, risk analysis, simulation, and reliability into probabilistic risk analysis. However, only a few of the papers by this group address the impact of engineering by probability distributions on variance, confidence, and requirements, and none of these address how to identify and analyze high-reliability/availability systems in which failure can be characterized as a rare event.

The rare-event papers presently available address identification of rare events in simple stochastic systems. Although some of these papers include methods of variance reduction, these incorporate little PD methodology and do not address the propagation of error inherent in a system with multiple stochastic inputs. The most comprehensive source was *An Introduction to Rare-event Simulation* by Bucklew (2004) that covered elements of simulation including Monte Carlo processes, methods of analyzing rare events, and a short segment on applications in reliability engineering.

None of the literature reviewed analyzed the impact of PD and rare-event analysis early in the process on systems requirements, and very few of these frame PD and rare-event analysis in the realm of systems engineering processes. Although these elements are necessary skills for systems engineers, PD and requirements analysis are addressed in the International Council on Systems Engineering (INCOSE, 2007) *Systems Engineering Handbook* only in the abstract, and no mention is made of how to deal with these tools in the context of writing actual systems requirements.

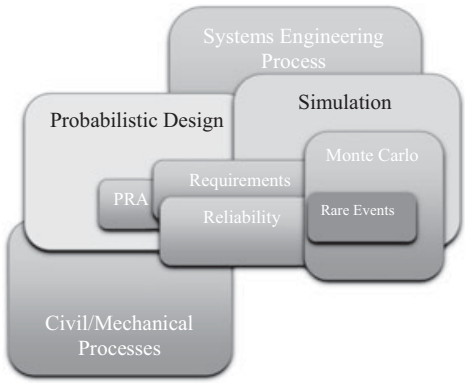

Fig. 1. Notional view of the existing literature.

The study of probabilistic requirements itself is an emerging field in engineering disciplines, but has not yet seen broad use. NASA design engineers have begun using it as a method for writing materials and structures requirements (White et al., 2009), and this is the closest methodology needed by systems engineers to write and verify system requirements. However, NASA team specialists cite critical weaknesses in the methodology in terms of its performance regarding high-reliability systems. For high-reliability systems with low-risk levels, the technique "requires thousands of runs" (White et al., 2009). This kind of brute force method can be costly and time consuming, requiring the use of complex mathematical search techniques or acceptance sampling (White et al., 2009). As this method has much in common with importance sampling and other methods of rare-event simulation, it is a logical conclusion that if these methods apply, a systems engineer can directly employ methods of rare-event simulation such as importance sampling and splitting to the problem.

In synthesizing this literature, this research proposed an iterative methodology for the analysis of high-reliability problems beginning with the basic systems design process, and then integrating the disparate elements of requirements engineering, PD, risk analysis, DES, and rare-event simulation. The resulting methodology enables a systems engineer to design a high-reliability system, to develop requirements for that system analytically, and to determine the sensitivity of the system's behavior to changes in those requirements, with the end goal of determining requirements that best improve the system's return to a steady state after the introduction of a rare event. Fig. 2 depicts the flow of the process through the new methodology.

2.1. The basic building blocks

This process begins with the fundamentals of the systems design process. Gibson et al. (2007) define this as a six-phase process:

1. Determine goals of the system.
2. Establish criteria for ranking alternative candidates.
3. Develop alternative solutions.

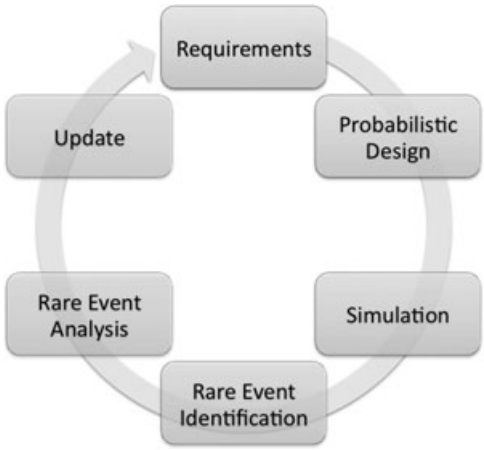

Fig. 2. An iterative process for specifying/verifying design requirements in high-reliability systems.

4. Rank alternative candidates.
5. Iterate.
6. Action.

This covers exactly what a systems engineer needs to do during the development and implementation of a system design. Using the above phases as a framework, we integrate the techniques a systems engineer needs when determining the requirements of a high-reliability system.

"Requirements engineering" is a methodology for determining the success of a system when that success is defined as how well the system does what it is supposed to do (Gibson et al., 2007). This process is in the first step of the systems design process in which we define the goals of the system. A system's initial requirements are written to govern how well the system will meet the goals assigned to it, and thus must be written carefully.

The requirements engineering process is best defined in the software engineering literature (Berenbach et al., 2009). The INCOSE defines various activities under this process (Haskins, 2007), which coincide with the Gibson–Scherer–Gibson Stakeholder Analysis portion of systems engineering (Step 1; Gibson et al., 2007). Berenbach breaks down the process in a similar fashion (Berenbach et al., 2009). Common activities in all sources covered under the process of requirements engineering are as follows:

1. Elicitation—communicating with customers and users to determine what their requirements are.
2. Analyzing and negotiation—determining whether the stated requirements are unclear, incomplete, ambiguous, or contradictory, and then resolving these issues.
3. Specification—recording and documenting these requirements.
4. Modeling—creating a model of the system in order to determine whether or not it will fulfill the desired functions.
5. Validation—confirming that the system will do what it is supposed to do in terms of the stated requirements and the client's desired end state.

6. Management—a variation on iteration, or ensuring that the system continues to do what it is supposed to do.

Some pitfalls in the existing system are the customers' difficulty in articulating their requirements and the engineers' difficulty in assessing exact requirements, as engineers do not know what effect certain properties of the system can have on their behavior. When the customers or the engineers do specify requirements, requirements lists and contracts can run to hundreds of pages, and it is virtually impossible to read such documents as a whole (Gibson et al., 2007), making it nearly impossible to understand these to design a system that can meet all such requirements. However, if these requirements can be estimated within a certain degree of confidence, PD techniques and rare-event analysis can help determine the sensitivity of these requirements and better help engineers structure them.

"Probabilistic requirements" are an extension of PD that allows engineers to assign a probability distribution to the performance of a component in the system or to the system itself, and then to assign a range of acceptance to that distribution (Lam et al., 2004). For example, if a requirement for a computer system component states that the component must not fail to the extent that a repair or replacement takes longer than 3 hours, a probabilistic requirement states that 95% of the time, a ship will not fail to the extent that a repair takes longer than 3 hours. In this case, that 95% metric becomes the acceptance probability.

Research conducted for NASA by White and Johnson (White et al., 2009) advances probabilistic requirements by introducing a means of verification. This integrates a probability of the system successfully falling within the acceptance probability and a "consumer's risk" (White et al., 2009). These two metrics are similar in calculation to the probabilistic materials and structures test methodology used by NASA.

The "PD" methods used in these tests deal with the effects of random variability on the performance of a system. Typically, the effects considered are those related to quality and reliability (Long and Narciso, 1999). Thus, PD is a tool that is frequently used in areas concerned with reliability, for example, product design, quality control, systems engineering, machine design, civil engineering, and manufacturing. Instead of using a standard safety factor that applies a set multiplicative factor to a measurement to account for variance, PD assumes probability distributions for the measurements in a design, as suggested in Fig. 3 (Long and Narciso, 1999). The risk of treating properties of a design deterministically instead of stochastically is a larger amount of uncertainty, and a design that totally ignores the upper tails of the distribution, which can stretch several standard deviations beyond the expected value (Long and Narciso, 1999).

When using a probabilistic approach to design, the designer no longer thinks of each variable as a single value or number. Instead, each variable is viewed as a probability distribution. From this perspective, PD predicts the flow of variability through a system (Long and Narciso, 1999). By considering this flow, a designer can make adjustments to reduce the flow of random variability and improve quality, and can see what combinations of variables among those in the range of possibility give the best performance.

Proponents of this approach contend that many quality problems can be predicted and rectified during the early design stages and at a much-reduced cost. The optimal design exhibits the smallest effects of random variability (Safie and Weldon, 2001). This could be the one design option out of several found to be the most robust. Alternatively, it could be the only design option available,

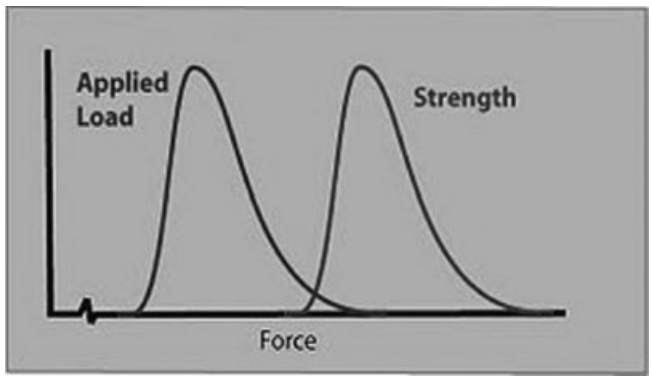

Fig. 3. Stress/strength diagram.

but with the optimum combination of input variables and parameters. This second approach is sometimes referred to as robustification, parameter design, or design for six sigma (Long and Narciso, 1999).

The process itself is very similar to the Gibson–Scherer–Gibson systems design process. It follows six general steps:

1. Identify potential failure nodes in a system or design—to accomplish this, the engineer must have a thorough understanding of the system. At this point, the design will have already passed through preliminary phases and with completion of initial testing.
2. Establish acceptable probabilities—to accomplish this, the engineer must establish criteria by which the performance of the system will be graded as acceptable or unacceptable.
3. Model the internal response of the system (outputs)—the engineer must develop models for the strength of components of the system through expert information, knowledge of the components, or simulation.
4. Define statistically random design variables (inputs)—here, the engineer creates a stochastic model for the process that will affect the model's behavior: stress, in the case of the materials model. These data are also obtained through expert information, knowledge of the components, or simulation.
5. Model the probability of total system failure as a function of individual failures—this begins with the mathematical formulation of the problem or a representation of the interaction of all the factors being analyzed.
6. Apply through simulation to determine adjustments to predetermined failure probabilities, and conduct sensitivity analysis (Long and Narciso, 1999).

To calculate each of these elements, the engineer can simulate the separate components, extract historical data, or consult additional experts. However, these processes are only as accurate as the research contributing to them. There is a base amount of uncertainty built into every model.

No matter how diligently a system is designed, all systems are subject to a certain amount of variability in their parameters. It does not make sense that the probabilistic analysis of system components should be confined to mechanical systems, so this process can be used effectively on

nonmechanical systems as well. DES accomplishes this to a certain extent in the DOE phase, and this method has found greater acceptance in the realm of systems engineering.

"Risk analysis" is an integral part of systems engineering. Risk is one of the primary reasons for analyzing events with very low probabilities, because the occurrence of certain events, even if they occur very infrequently, creates consequences significant enough to make those events worth examination (Haimes, 2009).

The idea of risk analysis has been around for a long time, since Pascal and Fermat established the roots of probability theory, purportedly over a game of dice. Not just satisfied with determining the likelihood of outcomes, Pascal and Fermat and their followers were interested in the outcomes with particularly significant consequences (Huang and Shahabuddin, 2003). The significant consequences of concern to engineers are those that bring about the system's inability to function as designed, or those that follow a system's failure to action.

"Probabilistic risk assessment" (PRA) has grown out of both PD and risk analysis. As a "systematic and comprehensive methodology to evaluate risks associated with every life-cycle aspect of a complex engineered technological entity" (Safie and Weldon, 2001), it was developed first within the nuclear power industry to address problems such as those resulted in the catastrophe at Three Mile Island (Bari, 2003). This technique asks and answers the three basic questions of risk as follows: (1) what can go wrong, (2) what are the consequences, and (3) how likely are the consequences? In Step 3, when determining the likelihood of the consequences, this process uses elements of PD to identify failure probabilities or consequence probabilities, thereby obtaining a better understanding of the system's behavior (Safie and Weldon, 2001). These assessments are applied to safety questions in areas varying from NASA's design of spacecraft to designing accident emergency response plans in environmental science and many other purposes.

"Extreme event theory" is another element of risk analysis that must be addressed, especially when taking into account low-probability events. Advanced first by Taleb, this theory focuses on the characteristics of disasters and why these became such an important part of study. Taleb cites three characteristics of a disaster: (1) they must be outlier events, not ones that fall within the normal range of predictable behavior; (2) they must carry an extreme impact; and (3) human nature needs explanation of these events in order to see predictable results (Taleb, 2007).

This theory has been quantified by Haimes and his colleagues (Bier et al., 1999) through the partitioned multiobjective risk method (PMRM) and the statistics of extremes (Haimes, 2009). Haimes (2009) first identifies "the fallacy of expected value." As noted with the factor of safety, the expected strength or value of a process only gives part of the information that a designer needs in order to understand the behavior of a process (Haimes, 2009). Stochastic processes having the same expected value can have very different extremes—without taking into account the tails of the distribution, a researcher will not be able to determine just how far the distribution can range. Haimes' (2009) statistics of the extreme are useful in discovering a brute force methodology for observing the effect of rare events on a system. This methodology will be discussed further in Section 4.

"Simulation" is a central component not only of PD but of systems engineering as a whole. Due to the probabilistic and random nature of complex systems and their responses, deterministic methods are ill-equipped to help a systems engineer understand the true behavior of a system. Simulation provides a methodology for conducting multiple experiments on the design of a system when actual

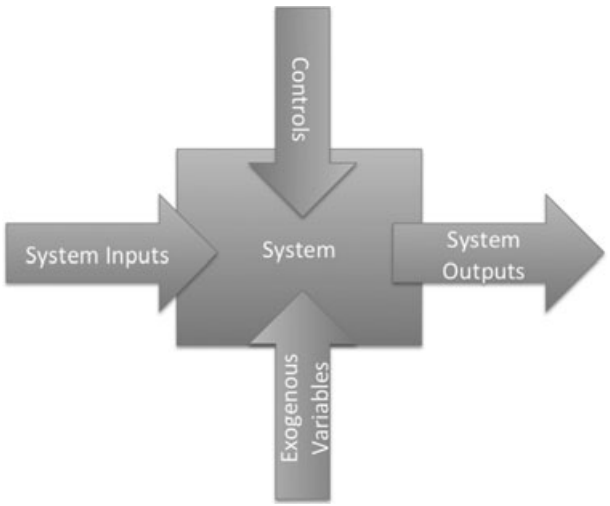

Fig. 4. System conceptual model.

experiments are too cost-prohibitive, time intensive, or dangerous to be conducted in a real-world scenario (Law, 2007).

Design is facilitated by the conceptual model depicted in Fig. 4. Outputs are the consequence of changes in the system state resulting from the action of inputs, controls, and exogenous variables. System inputs are values, or sequences of values, for which the engineers seek to determine the corresponding system outputs. These outputs are determined from experiments or expert opinion. Controls are elements that the systems engineer adds into the system in order to achieve requirements specified from the outputs. Exogenous variables are disturbances beyond the control of the engineer. In the case of Monte Carlo analysis, inputs are randomly drawn from input probability distributions and resulting the probability distributions for the outputs. Based on this conceptual model, the engineer can proceed to the design of the simulation, which closely follows the methodology for DOE.

In the case of a probabilistic model, it is useful to examine the model through the lens of DOE. In general, DOE refers to the "design of any information-gathering exercises where variation is present, whether under the full control of the experimenter or not" (Siddall, 1993). Much as we do with the systems design process, the designer using DOE is looking for the effect (outputs) of a certain process on some objects. By observing elements of DOE, we can better examine what variables and processes need to be taken into account in the systems design process to account for PD.

Most mathematicians agree that DOE has its roots in analysis of variance. Bayesian experimental design is based on Bayesian inference and the analysis of posterior probabilities to interpret the outputs of a system, or, in this case, the outputs of the experiment. Bayes' techniques allow designers to further analyze any uncertainties in the observation. The aim when designing an experiment is to maximize the expected utility of the experimental outcome, something we can understand when working with rare events, as we need to make every outcome of a rare-event count (Bertsekas and Tsitsiklis, 2008).

In terms of utility to a systems engineer, "response surface methodology" better outlines DOE than does classical experimental methodology. Response surface modeling is a technique that allows an engineer to approximate an unknown governing model or function of a system with an empirical model that creates a relationship between the response variables (outputs) and the explanatory variables (inputs). Using this methodology and the empirical model, the engineer can then predict the behavior of the system or model, given certain inputs, within a confidence interval (Myers and Montgomery, 1995). The engineer's level of success depends on how closely the response surface model approximates the actual behavior of the system, or the unknown underlying model.

In order to develop these response surface models, the engineer must have a working knowledge of experimental design in a statistical context. The typical practice is to develop a first-order approximation function, and then to progress from near optimum to a second-order function using one or several optimization techniques (Myers and Montgomery, 1995). The steps of this process are as follows:

1. Identify the factors (inputs and outputs) important to the system.
2. Conduct a screening experiment to eliminate the unimportant factors.
3. Determine the input variable values that result in the optimum response using a first-order equation as an approximation. This can be done via techniques such as line searches or method of steepest ascent.
4. When the process is near optimum levels, develop a second-order approximation to the model.
5. Define a "region of interest" or "region of experimentation" around the last best point in the search algorithm (Myers and Montgomery, 1995). This region is usually either defined by a cube or a sphere.
6. Iterate (Myers and Montgomery, 1995).

An engineer may conduct this process many times until he or she has arrived at an optimal approximating function. In theory, optimization problems with multiple parameters and dimensions can be solved through this method, but, in practice, it is far more efficient to restrict the process to as few variables as necessary to describe the behavior of the system (Box and Draper, 1987).

In designing a simulation as an experiment, it is extremely helpful to look at techniques being used in standard "DES." DES is a methodology that represents a system in a series of asynchronous chronological events. Normally, these events cause a change in the state of the system (Banks et al., 1999). The basic components of a DES model are as follows:

1. Clock—The clock must keep track of the current simulation time. The units are generally uniform and should be appropriate to the system being modeled.
2. Events list—A list of the events that will occur in the system, also known as pending events. These will occur according to a set of parameters. The list of events is typically sorted into a priority queue, in which events will occur in a strictly chronological order.
3. Random number generators—The system must generate random numbers for the event occurrences in keeping with the event parameters, and this is typically done using random number generators.

4. Statistics (tracking mechanism)—The simulation needs to have a mechanism for tracking measures of system performance. These are typically identified during the requirements analysis process.
5. Ending condition—Unless catastrophic failure occurs in the system, the designer must specify an ending condition to keep it from running forever. This can either be time based or event based. It is usually based on one of the tracking statistics being used (Banks et al., 1999).

The methodology used in DES design is remarkably similar to the end methodology in this research. However, the techniques used in DES, when applied directly to high-reliability systems composed of multiple stochastic processes, require extensive and sometimes prohibitive computation. The process relies heavily on techniques such as bootstrapping to create robustness in the statistics, and the probabilities being manipulated in a high-reliability system are, by definition, small.

This is the motivation behind "rare-event simulation" techniques. Mathematicians and practitioners have argued about the exact definition of rare events. Some define rare events as any event occurring with a probability as small as 10^{-9}, but practitioners ask why they should bother studying something that has a probability of occurring only once in a trillion time units (Bucklew, 2004). Some argue that these events should be discounted completely from the analysis of the system, because the cost accrued to mitigate their occurrence or effect would almost never be realized. (Taleb, 2007). However, many events thought to be rare enough to discount have had considerable negative impacts: the Challenger incident, the British Petroleum oil spill in the Gulf of Mexico, Three Mile Island, the 9/11 attacks, and the Haiti earthquake, just to name a few in recent history (Taleb, 2007).

When looking at events in the context of PD, probability distributions having extremely long tails occur occasionally, and sometimes the events of interest will be in those tails. These events are, by nature, rare events, but because of the risk context, these are important to consider in the comprehensive analysis of the system. Fortunately, "somewhat remarkable simple techniques exist that permit a very accurate estimation of such small probabilities within a matter of minutes on even modest-sized workstations" (Heidelberger, 1993).

Two of the most widely used methods for the deeper analysis of rare events are importance sampling and splitting. Importance sampling modifies the base probability distribution of a system so that rare events occur more frequently when simulated. Splitting is a methodology in which the simulation starts in an identified intermediate state to generate more occurrences of the rare event (Garvels, 2000).

"Importance sampling" is a simulation technique "based on a modification of the underlying probability distribution in such a way that the rare-events occur much more frequently" (Heidelberger, 1993). Using this approach, a region is selected to "hit the rare-event of interest more often" or to ensure that "the more likely or higher probability regions of [the target event] are hit more often during the simulation than the lower probability or less likely regions of [the probability distribution]" (Garvels, 2000). Changing the nature of the stochastic process governing the behavior of the system under study, in essence of creating a new system, avoids the propagation of error brought about by techniques such as bootstrapping.

For example, suppose that we want to estimate the probability p_t of the event $X \geq t$, where X is a continuous random variable with distribution $F(x)$ and probability density function $f(x)$. Let $Y(x)$ be an indicator random variable for this event of the function, such that

$$p_t = E(Y) = \int_{-\infty}^{\infty} y(x) f(x)\, dx.$$

Normally, we estimate p_t as

$$\frac{1}{n} \sum_{i=1}^{n} y(x_i),$$

where $\{x_i\}$ is a random sample generated from $F(x)$. Instead, suppose we sample from the distribution $F^*(x)$ with density $f^*(x)$, where $f^*(x) > 0$ whenever $y(x)f(x) \neq 0$. Then

$$E(Y) = \int_{-\infty}^{\infty} y(x) W(x) f^*(x) dx = E^*[y(x) W(x) dx],$$

where $W(x) = f(x)/f^*(x)$ is called the likelihood ratio. Then

$$\frac{1}{n} \sum_{i=1}^{n} y(x_i) W(x_i)$$

is an unbiased estimator of $E(Y)$.

Determining the likelihood ratio is critical and, in most practical cases, exceptionally difficult. The wrong choice of likelihood ratio in a Monte Carlo system could make the behavior of the new process even less accurate than simply running the Monte Carlo process *ad infinitum* (Garvels, 2000). Another problem with this method is that, for complex systems, the optimal change may exist analytically, but cannot always be found analytically. Even if found, the system's behavior can be so unpredictable that it might not improve the simulation's behavior at all, and in the worst case, will increase the variance to a point where true results are almost indistinguishable from experimental noise (Kroese and Rubenstein, 2004).

"Splitting" appears to be a more useful approach in the context of a system with multiple stochastic inputs. Splitting was introduced in the 1970s in the analysis of a simple discrete Markov chain and has been extended by multiple experimenters (Glasserman et al., 1999) to more complex problems (Garvels, 2000).

In the splitting method, we are again observing an event X, a function of process $X = X_t$, of which we take a sample of set S. We want to estimate the probability that event X_*, our rare event, occurs within sample set S, or $X_* \subset S$ occurs. This probability is p_* (Garvels, 2000).

Splitting assumes that $X = X_t$ is a Markov process. Since we can incorporate the history of the process into the system state, we can make any time-homogenous stochastic process obey the Markov conditions; thus, the Markov property is not really a limitation of the assumptions necessary for the proper use of the method. The fewer the variables necessary to describe the system as a Markov system, the better the method will work. Otherwise, as with so many other processes, we inherit the problems of multiple covariances, variable interference, and the curse of dimensionality (Berenbach et al., 2009). The method also relies on the identification of intermediate states in the system called "indicator states" that lead to the rare event so that the paths reaching the rare event can be selected

and replicated more easily. This can be done in a Markov process by identifying all the states leading to the rare-event state with nonzero probabilities (Bucklew, 2004).

The "basic idea of the splitting method is to consider the rare event as the intersection of a nested sequence of events. The probability of the rare event is thus the product of conditional probabilities" (Garvels, 2000). As in DES, those simulations using the splitting method must have a starting or a stopping state, or else the simulation could conceivably run forever. The stopping state can be as simple as reaching the target event, or the simulation can run until the statistics show it has returned to steady-state operations. Typically, the stopping criteria for a split are time dependent, which is one of the drawbacks of the method. It also raises the issue of dependency. Paths that are started in the same state are dependent because they share a common history. After the split, the paths will progress according to probabilities, but they will be influenced by the common starting state (Bucklew, 2004).

To begin identifying states for the splitting process, we denote T_A as the first time process X enters the rare-event set X_*, and T_B as an arbitrary stopping time in the process. Using the extreme events method, we can calculate what that time needs to be (demonstrated in Section 4). The rare event can then be expressed as $X_* = (T_A < T_B)$, the event in which set A is hit before the stopping time. We are then looking for an estimate of the probability $p_*(T_A < T_B)$, or the probability that the rare event will occur in the given time frame.

In Monte Carlo simulation, when the probability p_* of our rare event is extremely small, this is a difficult problem to attack. The Monte Carlo method consists of simulating the system without making any changes to its behavior; as a result, very few samples will actually hit the rare event. Analytically, we can show the difficulty in this by expressing samples $(X_1, \ldots, X_n)$ of the n random observations in which event A may occur. The value of a random variable X_i equals 1 when the rare event is seen in the ith attempt, and equals zero otherwise. This makes the Monte Carlo estimator expression:

$$\hat{p}_* = \frac{1}{n} \sum_{i=1}^{n} X_i.$$

3. Literature fusion

The result of analyzing these various techniques is that we see that integration points in the systems design process will improve the engineer's understanding of the system's behavior in order to determine requirements and whether or not changing the requirements has any effect on the system's ability to avoid rare or catastrophic events, or its ability to recover from those events. The systems process, with the integration of probabilistic and rare-event analysis techniques, is described as follows:

1. Define the problem—in this phase, the engineer sets base requirements drawn from expert advice and stakeholder analysis.
2. Simulate for probabilistic requirements—here, the engineer increases the robustness of the statistical data involving requirements through the use of simulation and defines acceptability intervals.

3. Refine the model—using probabilistic requirements and PD for key variables of the system, the engineer develops the base simulation model.
4. Identify the rare/failure event to be studied—engineers cannot anticipate all failure events, but there are risk analysis techniques that will allow an engineer to select an event by which to test the system's behavior and the effect of the requirements. In the case of a reliability system, an engineer could choose to have several components fail simultaneously.
5. Rare-event analysis—here, the engineer applies one of the many techniques for rare-event analysis. He or she can use analytic methods, importance sampling, or splitting as defined here to explore the behavior of the system before, during, and after the rare event in question.
6. Rare-event simulation—the engineer validates the rare-event analysis by determining whether or not the simulation accurately produces more rare events. Once the engineer understands how to replicate these rare events, further analysis can be conducted.
7. Sensitivity analysis—existing optimization methods can help the engineer determine optimal requirements. The engineer can also conduct a sensitivity analysis on the range of his or her probabilistic requirements to determine if adjusting this range affects the behavior of the system, or the occurrence of the rare event.
8. Iterate—the engineer tests various catastrophic scenarios based on the scope of the system in question, and finally determines the optimal range of the system's requirements.

The integration of probabilistic modeling and rare-event techniques into the systems design process allows the engineer to gain a greater understanding of the effect of the system's requirements on its performance in a risk or failure situation. There are two basic methods for implementing these various techniques. One is a brute force method that takes advantage of computational power and statistics of the extreme in order to compute state transition probabilities. The other is a technique that implements splitting into the analysis process to increase the probability of encountering a rare event in simulation.

4. Rare-event analysis in a PD

In order to observe the occurrence of effects of rare events, a systems engineer can take one of the two approaches identified through this research. The first is the brute force method mentioned previously using the statistics of the extreme.

The equations used in the statistics of extremes are derived from "the study of the largest or smallest values of random variables ... specifically concerned with the maximum or minimum values of sets of independent observations" (Haimes, 2009). These produce mathematical estimates for (1) the largest value anticipated in n number of observations, (2) the "dispersion" or variance of that value, and (3) the return time of that value, or how frequently that value will be observed in a system.

Suppose, we have identified a target "largest" value in the system under analysis. This could be the longest anticipated length in a queue, the largest anticipated cost of a failure, or any number of pieces of information. If we have enough historical data or data obtained from simulation to estimate a probability for this event, we can use this to determine just how long a simulation must run in order for us to observe the target event.

The statistics of the extremes states that the return time, or how frequently that value will be observed in a system, is $t = 1/(1 - p)$, where p is the exceedance probability $(1 - F(t))$ at the desired point, where $F(t)$ is the cumulative distribution function for t. Using the resultant value for t, we can advance the simulation time clock to that point and observe the effects of the target event. This is useful as we can "brute force" the occurrence of a rare-event in the simulation.

Although this is useful to a systems engineer, it is not as useful as being able to directly observe the effects of a rare event in the system, or being able to induce an extremely rare event repeatedly. At this point, existing rare-event simulation techniques such as splitting become useful.

As discussed previously, the splitting method relies on the hidden Markov properties of a model and the identification of transition states that lead to a target event. In the case of a simple queuing model, we know that a queue length of 2 must be preceded by a queue length of 1, that a queue length of 3 must be preceded by a queue length of 2, and so forth, even if there are two nearly simultaneous arrivals in the queue. By using the batch means method of simulation, we can estimate transition probabilities for a Markov transition matrix (White et al., 2009). However, this method is still computationally intensive, and easier methods for inducing rare events exist.

Splitting the model allows for the calculation of transition probabilities using a starting state and setting a stopping criteria. In the case of the queuing model example, we initialize the model with a certain queue length. The simulation stops when the queue length transitions states (i.e. there is a new arrival or a customer leaves the queue). By running these simulation criteria a large number of times, we develop a new transition matrix, one with much greater probabilities of seeing a rare event in the simulation, and thereby being able to study its effects.

5. Conclusions

Verifying design requirements for a high-reliability system implies a modest modification of systems methodology that integrates elements of DES, PD, rare-event simulation, and other existing techniques in order to give the systems engineer a breadth of tools with which to analyze the problem. When accompanied by proper sensitivity analysis and variance reduction techniques, advances in computing can provide greater understanding of the behavior of a complex system under various levels of stress.

In terms of computation and time, the extreme event method for generating rare events is the easiest. This essentially brute force method can show a systems engineer the time frame in which the rare event will occur, allowing him or her to advance the time clock on a simulation and observe the effect of the rare event, and allowing him or her to formulate basic decision strategies and high-level requirements. For more in-depth analysis, the splitting method is a more efficient process than importance sampling. If the systems engineer can accurately identify transition states or thresholds leading to the rare event in question, the Markov properties this method utilizes make it relatively easy to generate more occurrences of the rare event. This method is more efficient than the extreme event method as well. If an event occurs once every 10,000 time intervals normally and the splitting method can increase that to one every thousand, 10,000 time intervals will only produce one event with the extreme event method, but will produce 10 events utilizing splitting.

There are many applications for a technique that allows a systems engineer to easily analyze failures in high-reliability systems. One is in the analysis of the U.S. Government's natural disaster

response strategy. In California, when responding to wildfire and earthquake incidents, federal agencies must allocate resources from various parts of the state with relatively short notice, and must often activate members of the National Guard (Friel and Singer, 2005). This problem could be better analyzed through the systems lens by employing simulation and rare-event analysis techniques. By inducing natural disasters in a simulation and analyzing response times, especially when two or more natural disasters occur simultaneously in different parts of the state, systems engineers could gain a better understanding of the effectiveness of the response element.

Long distance transportation companies could also benefit from simple versions of this method in determining how to respond when trucks carrying either hazardous or perishable material break down. These trucking systems must have a high level of reliability in order to prevent risk to their cargo, and they must have redundancies to cover malfunctioning vehicles. A highly reliable system such as this could be modeled so that induced failures demonstrate the effectiveness of disaster response plans or whether or not the specified truck reliabilities in use are sufficient to maintain the level of reliability needed to minimize risk and loss to the owning company.

All these problems demonstrate the practical applicability of systems engineering with integrated elements of PD, requirements analysis, and rare-event simulation. Making these techniques more accessible can greatly benefit these areas and the field of systems engineering.

Acknowledgments

The authors thank Professors William T. Scherer and Gerard P. Learmonth of the Department of Systems and Information Engineering, University of Virginia for their assistance in the completion of this research.

References

Ang, A., Tang, W., 2006. *Probability Concepts in Engineering: Emphasis on Applications to Civil and Environmental Engineering*. Wiley & Sons, New York.

Asmussen, S., Rubenstein, R., 1983. The transform likelihood ratio method for rare-event simulation with heavy tails. *Queuing Systems: Theory and Applications* 46, 3, 317–351.

Au, S.K., 2005. Reliability-based design sensitivity by efficient simulation. *Computers and Structures* 83, 14, 1048–1061.

Banks, J., Carson, J.S., et al., 1999. *Discrete-Event System Simulation*. Prentice Hall, Upper Saddle River, NJ.

Bari, R., 2003. *Handbook of Reliability Engineering*. Springer, New York.

Berenbach, B., Paulish, D., Katzmeier, J., Rudorfer, A., 2009. *Software and Systems Requirement Engineering: In Practice*. McGraw-Hill Professional, New York.

Bertsekas, D., Tsitsiklis, J.N., 2008. *Introduction to Probability* (2nd edn). Athena Scientific, Belmont, MA.

Bhagwan, R., Savage, S., Voelker, G.M., 2003. Understanding availability. International Workshop on Peer-To-Peer Systems, Berkeley, CA, USA, pp. 256–267.

Bier, V.M., Haimes, Y.Y., Lambert, J.H., Matalas, N.C., Zimmerman, R., 1999. A survey of approaches for assessing and managing the risks of extremes. *Risk Analysis* 19, 1, 83–94.

Box, G.E.P., Draper, N.R., 1987. *Empirical Model-Building and Response Surfaces*. John Wiley & Sons, Hoboken, NJ.

Bucklew, J.A., 2004. *Introduction to Rare-event Simulation*. Springer, New York.

Clemens, P.L., 2011. Combinatorial failure probability analysis using MIL-Std822C. Available at www.fault-tree.net/clemens-comb-fail-prob-analysis.pdf (accessed 24 January 2011).

Cottrell, M., Fort, J.C., Malgouyres, G., 1983. Large deviations and rare-events in the study of stochastic algorithms. *IEEE Transactions on Automatic Control* AC-28, 9, 907–920.

Dec, J., Mitcheltree, R., 2002. Probabilistic design of a mars sample return earth entry vehicle thermal protection system. Proceedings of the 40th Aerospace Sciences Meeting & Exhibit, Reno, NV, pp. 14–17.

Desaulniers, G., Desrosiers, J., Dumas, Y., Solomon, M., Soumis, F., 1997. Daily aircraft routing and scheduling. *Management Science* 43, 6, 841–855.

Dezfuli, H., Stamatelatos, M., Maggio, G., Everett, C., Youngblood, R., 2010. *NASA Risk-Informed Decision Making Handbook*. Office of Safety and Mission Assurance, NASA HQ.

Frank, M.V., 2011. Case studies of uncertainty analysis in reliability and risk assessment. Available at www.fault-tree.net/frank-tutorial-uncert.pdf (accessed 25 January 2011).

Friel, B., Singer, P., 2005. Gaps remain in government strategy for handling natural disasters. *National Journal*. Available at www.govexec.com/dailyfed/1005/102805nj1.htm (accessed 9 March 2011) .

Garvels, M.J.J., 2000. The splitting method in rare-event simulation. Thesis Proceedings, Center for Telemetry and Information Technology, Twente University.

Gibson, J., Scherer, W.T., Gibson, W.F., 2007. *How to Do Systems Analysis*. Wiley & Sons, Hoboken, NJ.

Glasserman, P., Heidelberger, P., Shahabuddin, P., Zajic, T., 1999. Multilevel splitting for estimating rare-event probabilities. *Operations Research* 47, 4, 585–600.

Grieve, D.J., 2002. Probabilistic design. Available at http://www.tech.plym.ac.uk/sme/tsoc302/reliable1.htm (accessed 12 January 2011).

Haimes, Y.Y., 2009. *Risk Modeling, Assessment, and Management*. Wiley & Sons, Hoboken, NJ.

Haskins, C. (ed.), 2007. *INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*. OSE.

Heegaard, P., Helvik, B., Andreassen, R., 2005. Application of rare-event techniques to trace driven simulation. Proceedings of the 2005 Conference on Winter Simulation, 4–7 December 2005, Orlando, FL, pp. 509–518.

Heidelberger, P., 1993. Fast simulation of rare-events in queuing and reliability models. *ACM Transactions on Modeling and Computer Simulation* 5, 1, 43–85.

Henry, M., Haimes, Y.Y., 2009. Assessing uncertainty in extreme events: application to risk-based decision making in interdependent infrastructure sectors. *Reliability Engineering and Systems Safety* 94, 819–829.

Higgens, J.J., Keller-McNulty, S., 1995. *Concepts in Probability and Stochastic Modeling*. Duxbury Press, Pacific Grove, CA.

Homem-de-Mello, T., Rubenstein, R., 2002. Rare-event simulation and combinatorial optimization using cross entropy: estimation of rare-event probabilities using cross entropy. Proceedings of the 2002 Conference on Winter Simulation, 8–11 December 2002, San Diego, CA, pp. 310–319.

Huang, Z., Shahabuddin, P., 2003. New simulation methodology for risk analysis: rare-event, heavy-tailed simulations using hazard function transformations, with applications to value-at-risk. Proceedings of the 2003 Conference on Winter Simulation, 7–10 December 2003, New Orleans, LA, pp. 276–284.

INCOSE, 2007. *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*. INCOSE: International Council on Systems Engineering (ed. by Cecilia Haskins), Aug 2007.

Kappas, J., 2010. Review of risk and reliability methods for aircraft gas turbine engines. Technical Report, Airframes and Engines Division, Australian Government Department of Defense, Canberra. Available at http://dspace.dsto.defense.gov.au/dspace/handle/1947/4270 (accessed 28 July 2010).

Kelling, C., 1996. A framework for rare-event simulation of stochastic petri nets using 'RESTART.' Proceedings of the 1996 Winter Simulation Conference, 8–11 December 1996, Coranado, CA, pp. 317–324.

Kroese, D., Rubenstein, R., 2004. The transform likelihood ratio method for rare-event simulation with heavy tails. *Queuing Systems: Theory and Applications* 46, 3/4, 317–351.

Krystul, J., Blom, H.A.P., 2005. Sequential Monte Carlo simulation of rare-event probability in stochastic hybrid systems. Preprint: 16th IFAC World Congress, 4–8 June 2005, Prague, Czech Republic.

Labeau, P.E., 1996. Probabilistic dynamics: estimation of generalized unreliability through efficient Monte Carlo simulation. *Nuclear Energy* 23, 17, 1355–1369.

Lam, K.Y., Cheng, R., Liang, B., Chau, J., 2004. Sensor node selection for execution of continuous probabilistic queries in wireless sensor networks. Proceedings of the ACM 2nd International Workshop on Video Surveillance & Sensor Networks, New York, pp. 63–71.

Laplante, P.A., 2007. *What Every Engineer Should Know About Software Engineering*. CRC Press, Redmond, WA, 2007.
Law, A.M., 2007. *Simulation Modeling and Analysis* (4th edn). McGraw-Hill Higher Education, Boston, MA.
Learmonth, G.P., Lewis, P.A.W., 1973. *Naval Postgraduate School Random Number Generator LLRANDOM*. Naval Postgraduate School, Monterey, CA.
Long, M.W., Narciso, J.D., 1999. Probabilistic design methodology for composite aircraft structures. Final Report, June 1999, US DOT, Federal Aviation Administration (FAA).
McConnell, S., 1996. *Rapid Development: Taming Wild Software Schedules*. Microsoft Press, Redmond, WA.
Myers, R.H., Montgomery, D.C., 1995. *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. John Wiley & Sons, New York.
Neu, C.W., Byers, C.R., Peek, J.M., 1974. A technique for analysis of utilization-availability data. *The Journal of Wildlife Management* 38, 3, 541–545.
Pate-Cornell, E., Dillon, R., 2001. Probabilistic risk analysis for the NASA space shuttle: a brief history and current work. *Reliability Engineering and System Safety* 74, 3, 345–352.
Pham, T.Q., Neubert, H., Kamusella, A., 2008. Design for reliability and robustness through probabilistic methods in COMSOL multiphysics with OptiY. Proceedings of the 2nd European COMSOL Conference, Hanover, GE.
Rabelo, L., Sepulveda, J., Compton, J., Turner, R., 2006. Simulation of range safety for the NASA space shuttle. *Aircraft Engineering and Aerospace Technology* 78, 2, 98–106.
Randhawa, R., Juneja, S., 2004. Combining importance sampling and temporal difference control variants to simulate Markov chains. *ACM Transactions on Modeling and Computer Simulation* 14, 1, 1–30.
Rogers, J., Safie, F., Stott, J., 2002. *Application of Probabilistic Risk Assessment (PRA) During Conceptual Design for the NASA Orbital Space Plane (OSP)*. NASA/MFC.
Rubenstein, R., 1997. Optimization of computer simulation models with rare-events. *European Journal of Operations Research* 99, 89–112.
Rubenstein, R., 2007. *Simulation and the Monte Carlo Method.* Wiley & Sons, West Sussex, UK.
Rubino, G., Tuffin, B., 2009. *Rare-event Simulation Using Monte Carlo Methods*. Wiley & Sons, West Sussex, UK.
Safie, F. M., 1992. Use of Probabilistic Design Methods for NASA Applications. NASA STI/Recon Technical Report A 93, 50522, 1992.
Safie, F., 2002. *An Overview of Quantitative Risk Assessment of Space Shuttle Propulsion Elements*. NASA/MFC.
Safie, F., Weldon, D., 2001. Use of Probabilistic Engineering Methods in the Detailed Design and Development Phases of the NASA Ares Launch Vehicle. NASA/MFC, 2001.
Sage, A.P., Armstrong, J.E., 1992. *Introduction to Systems Engineering*. John Wiley & Sons, New York.
Siddall, J.N., 1993. *Probabilistic Engineering Design: Principles and Applications*. CRC Press, New York.
Sobkiw, W., 2008. *Sustainable Development Possible with Creative System Engineering*. CassBeth, Cherry Hill, NJ.
Srninivasan, R., 2002. *Importance Sampling—Applications in Communication and Detection*. Springer-Verlag, Berlin.
Stamatelatos, M., 2002. Probabilistic risk assessment: what is it and why is it worth performing? NASA Office of Safety and Mission Assurance.
Stellman, A., Greene, J., 2005. *Applied Software Project Management*. O'Reily Media, Cambridge, MA.
Systems Engineering Fundamentals. Defense Acquisition University Press, 2001.
Taleb, N., 2007. *The Black Swan: The Impact of the Highly Improbable*. Random House, New York.
Ushakov, A.A., Mishulin, I., Pankov, A., 2002. Probabilistic design of damage tolerant composite aircraft structures. DOT/FAA/AR-01/55 Technical Report.
Weigers, K.E., 2003. *Software Requirements* (2nd edn). Microsoft Press, Redmond, WA.
Weiss, G.M., Hirsch, H., 1998. Learning to predict rare-events in event sequences. Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining. AAAI Press, pp. 359–363.
White, K.P., Jr., Johnson, K.L, Creasey, R.R., Jr., 2009. Attribute acceptance sampling as a tool for verifying requirements using Monte Carlo simulation. *Quality Engineering* 21, 2, 203–214.
Youn, B.D., Choi, K.K., 2004. Selecting probabilistic approaches for reliability-based design optimization. *AIAA Journal* 42, 1, 124–131.