An Evaluation of the Optimality of Frequent Verification for Vertically Integrated Systems

Dr. Aditya U. Kulkarni (aditya88@vt.edu)
Grado Department of Industrial and Systems Engineering, Virginia Tech

Dr. Alejandro Salado (asalado@vt.edu), INCOSE Regular Individual Member
Grado Department of Industrial and Systems Engineering, Virginia Tech

Peng Xu (xupeng@vt.edu), INCOSE Student Member
Grado Department of Industrial and Systems Engineering, Virginia Tech

Dr. Christian Wernz (cwernz@vcu.edu)
Department of Health Administration, Virginia Commonwealth University

Abstract

Verification activities, such as inspection, testing, analysis, and demonstration, improve one's confidence in the system meeting the system requirements during the development process. Frequent verification is often advocated as a strategy that minimizes costs of rework over the entire design process, where frequent verification involves verifying after any change in the design. However, this strategy is yet to be validated. In this paper, we develop a belief-based model of verification in systems design to determine the conditions under which frequent verification is an optimal strategy for a vertically integrated organization. Our model uses belief distributions to capture the organization's dynamic confidence in the system design meeting a requirement of

interest during the development process. It also captures the organization's dynamic confidence in the correctness of its development activities (or design process) as a function of past verification activities and current system maturity. The analysis of our model shows that frequent verification is a cost-minimizing strategy for any level of belief in satisfying the requirement only when the organization has high confidence in the correctness of its design activities and the expected cost to rework a faulty design is greater than the costs to set up the verification activities throughout the development process. Otherwise, strategies with infrequent verification are superior. Our work contributes to the growing body of literature on the theoretical foundations of systems engineering and engineering design and seeks to provide practitioners with a means to determine optimal verification strategies.

Keywords: Belief-based model, verification strategy, dynamic programming

1 Introduction

Verification activities are used to determine if the system meets the system requirements that are set at the start of the design process [1]. That is, verification activities are the means by which engineers check if the system is being built "right" [2]. Furthermore, since system requirements are derived from stakeholder needs, verification activities during the development process also contribute to improve the organization and the customer's confidence in the system design satisfying stakeholder needs [3,4]. Furthermore, since requirements are often used as contractually binding agreements in the design of complex systems, such as satellites, verification activities become the means to demonstrate fulfillment of the contract from an engineering standpoint.

Monetarily, verification activities minimize rework costs by identifying design errors early in the development process [5]. Since verification activities are cost and time intensive [6], implementing

verification strategies that balance the cost of executing verification activities with the risk of undetected errors is an integral part of minimizing overall project costs [7]. The importance of identifying optimal verification strategies is further emphasized when one considers that verification activities are often planned and contracted upon early in the system development [8]. In this regard, frequent verification throughout the design process has been advocated for in both industry and the research literature [9-11]. However, the advocacy for frequent verification is based more on subjective experience than an analytical or theoretical foundation, and following this strategy without due consideration can increase the likelihood of an organization misallocating limited resources to verification activities [12].

In order to determine the conditions under which frequent verification is optimal, we study a belief-based model of verification of an organization developing a vertically integrated system. We abstract the problem space as a single system requirement, referred to as the requirement of interest. In this paper, we use the term *requirement* broadly and consider it to be an expected or desired attribute or set of attributes of the system under development, which can be formulated as a binary state or a direction on a scale.

Our model captures two high-level uncertainties [13] in the systems design process: 1) does the system baseline at the current development stage (e.g., system design, actual product, etc.) satisfy the requirement of interest, and 2) will the current set of development activities ensure that the system will continue to meet the requirement of interest? The first type of uncertainty represents the organization's confidence in the true state of the overall system as the development matures. The second type of uncertainty represents the organization's confidence in the correctness of its development activities. Furthermore, we consider two high-level costs associated with verification over the development process in our model: the cost to set up verification activities and the

expected cost to rework a faulty design. The optimal verification strategy is then defined as the strategy that minimizes the organization's expected verification costs (including the cost impact of deploying a faulty system) over the entire design process based on its confidence, or belief, in the correct state of the system design.

In addition to presenting an algorithm to solve our model numerically, we explore the parameter space of our model to characterize the scenarios for which frequent verification of the system is optimal. Our analysis shows that frequent verification is an optimal strategy for all belief levels in the correct state of the system design only when two conditions are met throughout the development process: (1) the organization's confidence in the correctness of its design activities (that is, its design process) is high, and (2) the expected cost to rework a faulty design is greater than the cost to set up verification activities. Our model expands prior work on mathematical models of verification that use belief distributions to model the organization's confidence in the true state of its design. Furthermore, our work contributes to the growing literature on studying the scientific foundations of systems engineering and engineering design.

The remainder of this paper is organized as follows. In Section 2, we briefly discuss literature related to our work. In Section 3, we develop the belief-based model of verification for an organization focused on determining verification strategies for a single system requirement. The parameter space of the belief-based model is explored in Section 4, and the conditions under which frequent verification is optimal are explored. An application example is provided in Section 5 to show how our model maps verification in practice. In Section 6, we present a discussion on validation aspects, and we conclude by summarizing the results and insights in Section 7.

2 Background and motivation

Recent literature in engineering design has acknowledged that system design is rife with epistemic uncertainty [14-16], and engineers make design decisions based on subjective beliefs about the true state of the system design [17-19]. Unlike aleatory uncertainty, where the uncertainty arises from physical variations in the underlying process, epistemic uncertainty arises due to a lack of knowledge about the current state of the design, or other aspects of the design process [20,21]. Since verification activities reveal the current state of the system design, it follows that verification activities improve an organization's knowledge in the current state of its design and design process [22]. That is, verification activities minimize the epistemic uncertainty in the design process for an organization. However, the majority of the literature on verification in systems design relies heavily on the traditional aleatory interpretations of probability [4-8,12,23-35].

With respect to epistemic uncertainty in systems design, previous works on verification in systems engineering can be broadly classified into conceptual approaches [8,28,29,33], empirical approaches [5,31,36-38], and probabilistic models of verification [7,34]. Conceptual approaches to verification espouse guidelines, industry standards, and best practices derived from personal experience and data from past projects. Empirical approaches to verification usually take the form of case studies. Both conceptual and empirical approaches to verification focus more on deriving best practices without addressing the scientific foundations of uncertainty in the system design process. This drawback is significantly reduced in probabilistic models of verification that specifically quantify the uncertainty and risk in systems design. However, probabilistic models of verification assume that all uncertain variables in the design process can be modeled using known stochastic models or processes. This bases all probabilistic models of verification on a strong assumption about the nature of the design process, thereby limiting their applicability.

To address these limitations, we derive optimal verification strategies using a belief-based approach to model the system development process. To the best of our knowledge, only the recent works by Salado et al. [3,39-41] have adopted the approach of capturing the epistemic uncertainty in the design process by using belief distributions to model verification strategies. In their work, verification strategies are derived based on the organization's changing belief in the system design meeting the system requirements. Using a belief-based approach to derive verification strategies is advantageous because beliefs better represent an organization's knowledge in the current state of its design. This results in a more accurate representation of the risk vs reward tradeoff in determining optimal verification strategies. We build upon this concept of an organization's knowledge in the state of its design in this paper.

The model and results in this paper expand our previous work [42], where we used a belief-based model to determine the optimal verification strategy for a single organization focused on a single system requirement, or the requirement of interest. In that work [42], we showed that frequent verification was not, in general, an optimal verification strategy. However, the belief-based model used in that paper assumed that the organization's confidence in its design activities not resulting in an error in the system design was constant throughout the design process. In this paper, we relax this assumption and derive more comprehensive and general results on when frequent verification is an optimal verification strategy.

Our decision to model the organization's confidence in the correctness of its design activities is motivated by the observation that design decisions are often made under considerable uncertainty [43,44]. This uncertainty can be broadly classified as exogenous, which is caused for example by the market and the environment, and endogenous, which is caused by the activities in the design process itself [13,45]. System verification contributes to reduce endogenous uncertainty.

Our work models the endogenous uncertainty in a design process as the organization's confidence level in whether the design activities will result in the system being built right [46]. For example, in the conceptual phase of an aircraft design, the organization's confidence level in whether the technology and capability choices made will satisfy system requirements may be low [47,48]. Similarly, when computational models of the aircraft are simulated and studied, the organization may be highly confident that the models suitably capture the proposed operating conditions [49]. Finally, when the aircraft is prototyped, the organization may be uneasy as to whether the prototype was constructed correctly. In each of these examples, the organization's confidence in the correctness of its design activities in each phase of the system development affects the organization's belief in the correct state of the system, which in turn will influence the optimal verification strategy. Thus, in order to determine the optimal verification strategy for an organization, we argue that it is necessary to account for the effect of the organization's dynamic, i.e., time-changing, confidence in the correctness of its design activities on the organization's dynamic belief in the true state of its design.

3 Model

3.1 Model environment

We consider a vertically integrated system (hence, developed by a single organization) and abstract the problem space as a single system requirement to: 1) determine the optimal verification strategy when there is no external pressure from other organizations participating in the project, and 2) to avoid confounding effects from a set of correlated requirements on the optimal verification strategy. Thus, in our model, the optimal verification strategy is derived based on the value of the system design meeting a single system requirement. The system development process is considered to progress through multiple development phases until the system is deployed (e.g.,

from conceptual design to preliminary design to detailed design and so forth [50]). In line with prior verification literature [51], we model the design process as a series of development phases for generality. We consider that the development process moves to the next development phase when the system design either changes in design attributes or design maturity. Here, design maturity refers to the level of implementation, or realization, of the system design, with concept of operations and block diagrams signifying low levels of design maturity and functional prototypes signifying high levels of design maturity.

In each development phase, the organization will execute design and verification activities to further develop the system, and so we divide each development phase into two periods: the design period and the verification period. In the design period, activities such as modeling, tradespace studies, and construction of mock-ups and prototypes are carried out, whereas in the verification period, activities such as testing, inspection, demonstration, and analysis are executed. In our model, we assume that the design period precedes the verification period in each development phase. Furthermore, we assume that in any given development phase, design activities are executed for certain, whereas verification activities are only executed when the verification strategy specifies it.

We assume that design choices and activities are fixed, and hence we normalize the cost of all design activities to $0 (since design activities are assumed to be executed in each development phase). For verification activities, we consider two high-level costs in each development phase: set-up cost for verification activities and the expected cost to rework a faulty design when verification reveals an error. The set-up cost for verification activities include the costs of executing the verification activities as well. We assume that if the organization chooses to verify the system design in a development phase, then it will incur the set-up cost for certain, whereas

the expected cost to rework is incurred only if verification reveals the system design does not meet the requirement of interest. In addition to the two aforementioned verification costs, we also consider the expected cost of project failure, which is incurred by the organization if the system design does not meet the requirement of interest at the end of all development phases.

We assume that both the set-up cost for verification activities and the expected cost to rework a faulty design will increase as the system development progresses. The latter is supported by empirical research [52,53]. The former is meaningful within the context of this paper, as we capture a general situation in which analyses become more refined as the design matures, and tests become more engaged as the system is realized.

As mentioned before, we consider two types of uncertainties for the organization in the design process: 1) does the current system design meet the requirement of interest, and 2) will the current set of design activities result in a system that meets the requirement of interest. The first type of uncertainty mentioned above is epistemic in nature [21]. Hence, we use belief distributions to model the organization's confidence in the correct state of the system design. The second type of uncertainty mentioned above can contain both aleatory and epistemic components. For example, the aleatory component would be the probability of a particular design activity being carried our correctly, whereas the epistemic component would be the probability of the design activity resulting in the system meeting the requirement of interest. Hence, we model the second type of uncertainty by a factor, whose value is in the range $[0,1]$, referred to as the belief retention factor, which affects the organization's belief in the correct state of the system design every time design activities are executed.

In our previous work [42], the belief retention factor was assumed to be stationary. This stationarity assumption ignores the possible correlation between the current design maturity and

past verification activities on the belief retention factor. For example, design decisions for new technological capabilities in aircraft design are challenging since designers do not know a priori if their design decisions will result in a system that meets all requirements [48]. In this scenario, the organization's confidence in the correctness of its design activities is both a function of design maturity and past verification activities. Thus, in this paper, we relax the stationarity assumption on the belief retention factor, and the belief retention factor is now a function of the current development phase (a proxy for design maturity), and past verification activities.

When no verification activities are carried out, it is logical for the organization's belief in the system meeting the requirement of interest to reduce with each development phase, as there is always a chance to introduce an error in the design process. Similarly, it is logical for the organization's belief in the system meeting the requirement of interest to increase after executing verification activities. This increase results from either obtaining successful verification results or from performing corrective actions when verification activities reveal errors in the system. In our model, we assume that the organization's belief is transformed by the belief retention factor after the design activities are executed in each phase, whereas verification activities reveal all errors in the system design and lead to the highest possible belief value in the system design meeting the requirement of interest.

Our assumptions on the increase and decrease in the organization's belief based on design and verification activities imply that the organization can be confident about its design after verification in a development phase, but this confidence can reduce in future development phases due to future design activities. To motivate this assumption, consider the landing gears on an aircraft with the requirement of interest being the gears deploy completely within a certain time limit. Early verification can ensure the gear controller design is correct and the right torque is

produced by the gear motors to deploy the gears within the time limit. However, as the design matures, other factors such as the amperage provided to the gear motors by the power supply or operation of the landing gear doors may negatively affect the aircraft meeting the requirement of interest. Furthermore, it is possible for the gear design to be correct, but the assembly to be carried out incorrectly. To account for such scenarios, we assume that even after the design is verified in a certain development phase, the organization's belief can be reduced by future design activities.

3.2 Model parameters

For ease of discussion, we will henceforth refer to *development phases* simply as *phases*. The number of phases in the design process is denoted by N, and a generic phase is denoted by $n \in \{1,\ldots,N\}$. We assume the organization broadly classifies the true state of the system design as either meeting the requirement of interest or not meeting the requirement of interest. We say the system design is in the *ideal* state when it *meets* the requirement of interest, and it is in the *non-ideal* state when it *does not meet* the requirement of interest. Restricting the state space to two states simplifies the communication of key insights without limiting the generality of the model. The model could be extended to a larger state space, though the characteristics and general properties of the findings for the two-state model would still apply.

We denote the organization's belief in the ideal state of the system design *at the start* of phase *n* by $\beta_n \in [0,1]$, and so the organization's belief in the non-ideal state of the system design at the start of phase *n* is equal to $1-\beta_n$. As per our assumptions, the organization's belief value β_n is reduced after the design period. The degree to which this belief value is reduced will be influenced by the culture of the organization. Thus, to model this reduction in belief, we say that design activities in phase *n* reduce the organization's belief value β_n by a factor $\varepsilon_{n,l} \in [0,1]$ to $\beta_n \varepsilon_{n,l}$.

Here, $\varepsilon_{n,l}$ is the belief retention factor, with the subscripts *n* and *l* denoting the current phase and the last phase in which the organization verified the design, respectively. It follows that $l < n$ for all $n \in \{1,\ldots,N\}$. Furthermore, we denote the scenario where the organization has not verified the design in any of the previous phases by $l = 0$. If the organization verifies the system design in phase *n,* then as per our assumptions, $\beta_{n+1} = 1$. However, if the organization does not verify the system design in phase *n,* then the organization's belief is unchanged after the design period and hence $\beta_{n+1} = \beta_n \varepsilon_{n,l}$.

As mentioned before, we assume that design activities and choices are fixed, and hence we normalize design costs to \$0. The influence of design decisions in verification decisions [54] is incorporated in our model by the belief retention factor $\varepsilon_{n.l}$, which captures the influence of the design activities on the organization's belief, which in turn affects the optimal verification strategy. Thus, our model accounts for the effect of design activities on the verification strategy.

In this paper, for ease of discussion, we only analyze those scenarios where the organization's confidence in the correctness of its design activities decreases linearly with each consecutive phase in which no verification is carried out. Specifically, we assume $\varepsilon_{n,l} = k_1\left(1 - \frac{(n-l)}{k_2 * N}\right)$. Parameter k_1 represents the organization's baseline confidence in the correctness of its design activities with respect to the requirement of interest immediately after the design has been verified. A low value of k_1 implies the organization has low confidence in the correctness of its design activities even after it verifies the system design. This could either be due to the organization believing the design activities *related* to the requirement of interest will result in a design error, or the design activities *unrelated* to the requirement of interest will change the system design in a manner that leaves it in

the non-ideal state. Whereas, a high value of k_1 implies the organization has high confidence in its design activities *not* resulting in a design error.

The parameter k_2 models the rate of decay in the organization's confidence in the correctness of its design activities with each consecutive phase for which the organization does not verify the system design. A low value of k_2 implies that a large number of changes in either design attributes or design maturity occur with each passing phase, and without verification, the organization's confidence in the correctness of its design activities decays rapidly. Similarly, a high value of k_2 implies that the number of changes in either design attributes or design maturity are few, and thus the organization's confidence in the correctness of its design activities decays slowly.

In phase *n,* we denote the organization's decision by $d_n \in \{v, -v\}$, where $d_n = v$ means that the organization decides to verify the system design, and $d_n = -v$ means that the organization decides not to verify the system design. The set-up cost of verification activities is denoted by c_n, and the expected cost to rework a faulty design is denoted by r_n. Furthermore, we denote the expected cost of project failure by r_F. Figure 1 shows the evolution of the organization's belief in a given phase *n*.

To concisely present out analysis and results, we will use a vector notation for the remainder of this paper. We denote the vector of beliefs for the organization by $b_n = (1-\beta_t, \beta_t)$. Furthermore, let $b_{n,0} = 1-\beta_n$ denote the first element of vector b_n and let $b_{n,1} = \beta_n$ denote the second element of

vector b_n. To represent the transformation of b_n into b_{n+1}, we define $\mathbb{T}_{-v,n,l} = \begin{pmatrix} 1 & 0 \\ 1-\varepsilon_{n,l} & \varepsilon_{n,l} \end{pmatrix}$ and $\mathbb{T}_v = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}$. Then,

$$b_{n+1} = \begin{cases} b_n \mathbb{T}_{-v,n,l} & d_l = v, d_{l+1} = -v, \ldots, d_n = -v \\ b_n \mathbb{T}_v & d_n = v \end{cases} . \quad (1)$$

We denote the organization's vector of costs associated with its decisions to verify and not verify the system design by $f_{n,v}$ and $f_{n,-v}$, respectively. The organization's cost vector for verifying the system design in phase *n* is defined by $f_{n,v} = (c_n + r_n, c_n)'$, where the apostrophe denotes the vector transpose. As per our model assumptions, the organization incurs no cost for *not* verifying the system design in development phases $n < N$, and so for all $n < N$, $f_{n,-v} = (0,0)$. At the end of the final phase and if the system design is in the non-ideal state, the organization incurs the cost of project failure. We capture this by defining the organization's cost vector for not verifying the system design in the final phase *N* as $f_{N,-v} = (-r_F, 0)'$. Table I summarizes the notation presented so far.

Table I: Summary of key notation

Notation	Description
N	Number of development phases
n	Generic development phase
β_n	Organization's belief in the ideal state of the system design at the *start* of development phase *n*
$b_n = (1-\beta_n, \beta_n)$	Vector of beliefs

$\varepsilon_{n,l}$	Belief retention factor: organization's confidence in the correctness of its design activities in phase *n* given the design was last verified in phase *l*
k_1	Organization's baseline confidence in its design activities
k_2	Rate of decay of $\varepsilon_{n,l}$
$d_n \in \{v, -v\}$	Organization's decision to verify (*v*) or not verify (-*v*) in phase *n*
$\mathbb{T}_{-v,n,l}$	Belief transformation matrix if the organization does not verify the design in phase *n* given the organization last verified in phase *l*
$\mathbb{T}_v$	Belief transformation matrix in any phase when the organization verifies its design
c_n	Set-up cost of verification activities in phase *n*
r_n	Expected cost to rework a design in non-ideal state in phase *n*
r_F	Expected cost of project failure
$f_{n,v}$, $f_{n,-v}$	Cost vectors associated with the organization's decision to verify or not verify, respectively

3.3 Belief-based optimal verification strategy

We now present the method to determine the organization's optimal verification strategy using the model presented above. To begin, note that in phase *n,* the organization's decision d_n cannot affect the costs the organization incurred in phases $1,\ldots,n-1$. However, d_n determines the organization's immediate costs in phase *n* and it also affects the costs the organization can expect to incur over phases $n+1,\ldots,N$. This is so since d_n determines the transformation of b_n into b_{n+1}, which in turn affects the organization's decision d_{n+1}, the organization's immediate costs in phase $n+1$ and the transformation of b_{n+1} into b_{n+2} through the decision d_{n+1}, and so on until the end of all phases. In this regard, the optimal decision for the organization in phase *n* is the one that minimizes the organization's expected costs over phases $n, n+1,\ldots N$ given the organization's

belief vector at the start of phase *n* is b_n. This line of reasoning can be extended all the way back to phase 1, where for any given initial belief vector b_1, there is a vector of decisions $(d_1,\ldots,d_N) \in \{0,1\}^n$ that minimizes the verification costs for the organization over all phases. Hence, we define *optimal verification strategy* for the organization as the function $D^*(n,b_1) \in \{v,-v\}$ which, for a given initial belief vector b_1, specifies a decision, v or $-v$, for each phase *n* such that the organization's expected costs over phases $n, n+1,\ldots N$ is minimized.

To determine $D^*(n,b_1)$, the organization must know the expected costs over phases $n,\ldots N$ resulting from its decision d_n for any possible belief vector b_n. To this end, we define $V_{n,l}(b_n,d_n)$ as the lowest possible cost the organization can expect to incur over phases $n, n+1,\ldots, N$ for decision d_n and a belief vector b_n, given that the last phase in which the organization verified the system design was phase *l*. We refer to $V_{n,l}(\cdot)$ as the organization's optimal cost function.

At this point, it is important to note that we have defined the optimal cost function $V_{n,l}(\cdot)$ to address a single objective (cost). This has been a common methodological approach in verification literature [6, 55] and is consistent with current literature on engineering decisions that suggests that all decision criteria can be converted to a cost and/or financial reward equivalent [55-58]. While doing so, our model parameters implicitly account for at least three major objectives associated with verification activities: 1) risk mitigation (belief vectors and belief retention factor), 2) quality assurance (expected cost of rework and expected cost of failure), and 3) compliance to standards (expected cost of failure) [6,59]. Yet, our modeling approach can be extended to those scenarios where the organization desires to use a multi-objective vector instead.

The optimal cost function $V_{n,l}(\cdot)$ for all phases can be determined using backward induction [60] as follows. In the final phase N, the organization only needs to consider the immediate costs of its decision. Hence, given $l < N$, the organization's optimal cost function for its decision to verify the system design is defined by

$$V_{N,l}(b_N, v) = b_N \mathbb{T}_{-v,N,l}\, f_{N,v} + b_N \mathbb{T}_v f_{N,-v} = b_N \mathbb{T}_{-v,N,l}\, f_{N,v}, \tag{2}$$

since $b_N \mathbb{T}_v f_{N,-v} = 0$, and the organization's optimal cost function for its decision to not verify the system design is defined by

$$V_{N,l}(b_N, -v) = b_N \mathbb{T}_{-v,N,l} f_{N,-v}. \tag{3}$$

In phase $N-1$, given $l < N-1,$ the optimal cost function $V_{N-1,l}(\cdot)$ must consider the organization's immediate cost in phase $N-1$ from decision d_{N-1} and the minimum cost the organization can expect to incur in phase *N* for the belief vector b_N, which results from the transformation of b_{N-1} through the decision d_{N-1}. Since equations (2) and (3) define the minimum cost the organization can expect to incur in phase *N* for each of its decisions, in phase $N-1$, the organization only needs to compute its immediate cost resulting from decision d_{N-1}, determine the transformation of b_{N-1} into b_N through the decision d_{N-1}, and then use b_N to determine the minimum expected cost in phase *N*. Hence, given $l < N-1,$ in phase $N-1$, the organization's optimal cost function for its decision to verify the system design is defined by

$$V_{N-1,l}(b_{N-1}, v) = b_{N-1} \mathbb{T}_{-v,N-1,l} f_{N-1,v} + \min\{ V_{N,N-1}(b_{N-1}\mathbb{T}_v, v), V_{N,N-1}(b_{N-1}\mathbb{T}_v, -v)\}, \tag{4}$$

and its optimal cost function for the decision to not verify the system design is defined by

$$V_{N-1,l}(b_{N-1},-v)=\min\{V_{N,l}(b_{N-1}\mathbb{T}_{-v,n,l},v),V_{N,l}(b_{N-1}\mathbb{T}_{-v,n,l},-v)\}. \quad (5)$$

The optimal cost functions for all remaining phases can be determined using the same procedure as the one presented above for phase $N-1$.

With the knowledge of $V_{n,l}(\cdot)$ for all $n\in\{1,\ldots,N\}$, the organization can then determine the optimal verification strategy $D^*(n,b_1)$. We will illustrate the procedure to determine $D^*(n,b_1)$ for a single set of belief vectors $\{\hat{b}_n\}_{n\in\{1,\ldots,N\}}$, where $\hat{b}_n$ denotes the organization's belief vector in phase *n* that results from the organization following the optimal verification strategy in phases $1,\ldots,n-1$. In phase 1, the organization's belief vector is $\hat{b}_1$, $l=0$, and hence

$$D^*(1,\hat{b}_1)=\arg\max_{d_1\in\{v,-v\}}\{V_{1,0}(\hat{b}_1,d_n)\}. \quad (6)$$

If $D^*(1,\hat{b}_1)=-v$, then $\hat{b}_2=\hat{b}_1\mathbb{T}_{-v,1,0}$ and *l* remains set at 0. Else, if $D^*(1,\hat{b}_1)=v$, then $\hat{b}_2=\hat{b}_1\mathbb{T}_v$ and $l=1$, which reflects the organization verified the system design in phase 1. Similarly, in phase 2, for the known belief vector $\hat{b}_2$ and known value of l, we know

$$D^*(2,\hat{b}_1)=\arg\max_{d_2\in\{v,-v\}}\{V_{2,l}(\hat{b}_2,d_2)\}. \quad (7)$$

Proceeding in a manner similar to the one presented above, in phase *n,* the value of *l* and $\hat{b}_n$ are known beforehand since $D^*(1,\hat{b}_1),\ldots,D^*(n-1,\hat{b}_1)$ are determined beforehand. It then follows that in phase *n*

$$D^*(n,\hat{b}_1)=\arg\max_{d_n\in\{v,-v\}}\{V_{n,l}(\hat{b}_n,d_n)\}. \quad (8)$$

The formulation presented above to determine $D^*(n, b_1)$ is similar in structure to partially observable Markov decision processes (POMDPs) [61]. There are two notable differences between our formulation and POMDPs: 1) POMDPs explicitly account for observations from a decision, whereas our model implicitly accounts for the observations in the design process, and 2) POMDPs use Bayes' rule to update beliefs based on observations, whereas in our model beliefs are transformed based on the organization's decision. The optimal verification strategy $D^*(n, b_1)$ in our model can be determined by using standard POMDP solution algorithms [62], after adjusting for the differences between POMDPs and our model. We present one such algorithm to numerically determine the optimal verification strategy $D^*(\cdot)$ in the Appendix.

4 Analysis

4.1 When is frequent verification optimal?

In our model, frequent verification being an optimal strategy is equivalent to the organization verifying the system design in all phases for all valid initial belief vectors b_1. It is mathematically intractable to derive an analytical closed form expression for $D^*(n, b_1)$ that does not contain the maximum function. Instead, we explore the parameter space of our model to determine the conditions under which frequent verification is an optimal strategy for the organization. To this end, we will assume that the organization verifies the system design with respect to the requirement of interest in all phases and then proceed to analytically determine the necessary conditions that parameters r_F, c_n and r_n must satisfy for our assumption of frequent verification to remain valid.

We begin with parameter r_F. Consider the final phase. Per our assumption, the organization has verified in phase $N-1$. Then, the organization will verify the design in the final development phase only if $V_{N,N-1}(b_N, v) > V_{N,N-1}(b_N, -v)$

$$\Rightarrow c_N + r_N(1-\beta_N\varepsilon_{N,N-1}) < r_F(1-\beta_N\varepsilon_{N,N-1}) \Rightarrow \beta_N < \frac{1}{\varepsilon_{N,N-1}}\left(1-\frac{c_N}{r_F-r_N}\right).$$

Since $\beta_N \le 1$, the above condition implies that the organization will verify the system design in the final development phase for all belief values if

$$1 < \frac{1}{\varepsilon_{N,N-1}}\left(1-\frac{c_N}{r_F-r_N}\right) \Rightarrow r_F > \frac{c_N + r_N(1-\varepsilon_{N,N-1})}{(1-\varepsilon_{N,N-1})} \Rightarrow r_F > c_N + r_N + \frac{c_N\varepsilon_{N,N-1}}{(1-\varepsilon_{N,N-1})}\ . \quad (9)$$

Hence, it is optimal for the organization to verify the design in the final phase, for any belief value β_N, when the expected cost of failure is greater than the maximum possible cost of verification in the final phase, $c_N + r_N$, by a margin of at least $c_N\varepsilon_{N,N-1}/(1-\varepsilon_{N,N-1})$.

We now consider the parameters c_n and r_n for phases $n = 1,\ldots,N-1$. Per our assumption, the organization has verified the design in phase $n-1$ and it will verify the design in phase $n+1$. Then, verification in phase *n* is an optimal strategy only if $V_{n,n-1}(b_n, v) > V_{n,n-1}(b_n, -v)$

$$\Rightarrow c_n + r_n(1-\beta_n\varepsilon_{n,n-1}) + c_{n+1} + r_{n+1}(1-\varepsilon_{n+1,n}) < c_{n+1} + r_{n+1}(1-\beta_n\varepsilon_{n,n-1}\varepsilon_{n+1,n-1})$$

$$\Rightarrow \beta_n \le \frac{r_{n+1}\varepsilon_{n+1,n} - r_n - c_n}{(r_{n+1}\varepsilon_{n+1,n-1} - r_n)\varepsilon_{n,n-1}}.$$

Since $\beta_n \le 1$, the above condition implies that verification in phase *n* is optimal for all belief values when

$$1 \le \frac{r_{n+1}\varepsilon_{n+1,n} - r_n - c_n}{(r_{n+1}\varepsilon_{n+1,n-1} - r_n)\varepsilon_{n,n-1}} \Rightarrow r_{n+1} \ge \frac{c_n + r_n(1-\varepsilon_{n,n-1})}{\varepsilon_{n+1,n} - \varepsilon_{n+1,n-1}\varepsilon_{n,n-1}}. \quad (10)$$

The necessary condition for the organization to verify in phase *n* is that the expected cost to rework the design in phase $n+1$ be greater than the lower bound defined by condition (10), with the lower bound being a function of the set-up and rework costs in phase *n*.

Conditions (9) and (10), in effect, base the organization's current optimal strategy in the comparison of future expected costs of rework, or failure, to a scaled value of the current costs of verification set up and rework. This is in line with verification literature that advocates for verification in order to avoid expensive rework in later design stages [6,23]. However, conditions (9) and (10) also prove that frequent verification is not an optimal strategy in general. It is not enough that future costs of design rework, or failure, be simply greater than the current costs of verification set up and rework. Indeed, they must be greater by a margin, defined by conditions (9) and (10) for frequent verification to be optimal for the organization.

4.2 Effect of the belief retention factor on optimal strategy

The results presented in Section 4.1 generalize our previous work [42]. In this section, we explicitly define the effect of $\varepsilon_{n,l}$ and *N* on frequent verification being an optimal strategy. Indeed, condition (10) is an implicit function involving $\varepsilon_{n+1,n}$, $\varepsilon_{n+1,n-1}$ and $\varepsilon_{n,n-1}$, with $\varepsilon_{n,l}$ in turn being a function of *N*. To derive additional qualitative insights on the effects of $\varepsilon_{n,l}$ and *N* on the optimal verification strategy, we will now study our model numerically.

For brevity, we will restrict our attention to those cases where c_n and r_n either increase linearly or superlinearly, while ignoring the scenarios where c_n and r_n increase sublinearly. We will refer

to each possible combination of the type of increase in c_n and r_n as a case. The 4 possible cases are defined in Table II. We will begin with an initial set of values for all parameters. The parameter values will then be systematically changed to bring out the effect of k_1, k_2 and N on the optimal verification strategy.

Table II: Rate of increase in costs for each case

Case	*Increase in* c_n	*Increase in* r_n
1	Linear	Linear
2	Linear	Superlinear
3	Superlinear	Linear
4	Superlinear	Superlinear

4.2.1 Case construction

The initial set of parameter values for each case are listed in Table III. These parameter values are notional, with the ordinal relationship between the parameters being the purpose of our analysis. Specifically, for this initial set of parameter values, we have captured the following relations: 1) the expected cost to rework a design in its non-ideal state is always greater than the set-up cost for verification activities in all phases, 2) the expected cost of failure satisfies condition (9) for all cases, 3) $k_1 = 0.95$ implies that the organization's baseline confidence in the correctness of its design activities (that is, on its design process) is high, and 4) $k_2 = 5$ implies that the organization's confidence in the correctness of its design activities decays at a relatively low rate when it does not verify the design for multiple consecutive phases.

Table III: Initial set of parameter values

Case	c_n	r_n	r_F	k_1	k_2	N

1	$\$5000n$	$\$100{,}000n$	$\$10^8$	0.95	5	8
2	$\$5000n$	$\$100{,}000n^2$	$\$10^8$	0.95	5	8
3	$\$5000n^2$	$\$100{,}000n$	$\$10^8$	0.95	5	8
4	$\$5000n^2$	$\$100{,}000n^2$	$\$10^8$	0.95	5	8

The different sets of changes in the initial parameter values, henceforth referred to as iterations, are listed in Table IV. As shown in Table IV, we begin by increasing the number of phases to determine the effect of N on the optimal verification strategy in iteration 1. This is followed by the change in the organization's baseline confidence in the correctness of its design activities in iteration 2. In iteration 3, we reset the value of the organization's baseline confidence to the original value and change the decay rate of the organization's confidence in the correctness of its design activities. Finally, we change both the organization's baseline confidence and the decay rate of its confidence in iteration 4.

Table IV: List of parameter perturbations to be explored

Iteration	*Previous parameter(s) value*	*New parameter(s) value*
1	$N = 8$	$N = 16$
2	$k_1 = 0.95$	$k_1 = 0.7$
3	$k_2 = 5,\ k_1 = 0.7$	$k_2 = 40,\ k_1 = 0.95$
4	$k_1 = 0.95,\ k_2 = 40$	$k_1 = 0.99,\ k_2 = 1$

4.2.2 Results of parameter changes

Figure 2 graphs the results from numerically solving our model for the initial set of parameter values listed in Table III. In each graph, the initial belief vector $b_1 = (1-\beta_1, \beta_1)$ is described by the value of β_1 on the x-axis, and the optimal strategy is plotted for each phase on the y-axis. Since

condition (9) is satisfied for all cases, we see that verifying the system design in the final phase for all beliefs is optimal for all cases. However, we observe that frequent verification is an optimal strategy only for case 2. This is due to condition (10) being violated in phase 3 for case 1, in phase 2 for case 3, and phase 7 for case 4.

Consider now the results of iteration 1, where we increase the value of *N* from 8 to 16, graphed in Figure 3. We observe that the structure of the optimal verification strategy has not changed for cases 2 and 3. In fact, frequent verification is still optimal for case 2, and our model still suggests that the design be verified in phases 1 and 3 for case 3. However, increasing the number of phases changes the structure of the optimal verification strategy for cases 1 and 4. Specifically, in case 1, verification is more irregular when the number of phases increase. In case 4, previously our model suggested that the organization not verify only in the penultimate phase. When *N* is increased to 16, our model suggests verification be avoided in every alternate phase after the 3rd phase in case 4.

A potential reason for the increase in the number of phases affecting the structure of the optimal verification strategy in cases 1 and 4 is that the rate of change in c_n and r_n is the same, with both costs increasing linearly in case 1 and superlinearly in case 4. However, this does not sufficiently explain the number of consecutive phases with no verification in between two phases where verification is suggested for both case 1 and 4. Hence, we hypothesize that increasing the number of phases leads to a change in $\varepsilon_{n,l}$, which also affects the optimal verification strategy for cases 1 and 4.

In iteration 2, we reduce the organization's baseline confidence in the correctness of its design activities to $k_1 = 0.7$. Figure 4 graphs the results for iteration 2. As illustrated in Figure 4, the

optimal verification strategy for the organization, in all cases, is to verify the design only in the final phase. A low baseline confidence in the design activities should result in frequent verification being optimal for the organization. Yet, our model suggests otherwise. We reason that this is due to the two-state feature of our model. If the organization's confidence in its design activities is low, and the design is either correct or incorrect, then it is rational for the organization to verify the design only in the final phase when k_1 is low. If the organization verified in any other phase, the benefits of verification would be reduced by the next phase's design activities, after which the organization would once again not be confident in the correct state of the design.

We conjecture that frequent verification could be optimal, when the organization's baseline confidence in the correctness of its design activities is low, if the possible states of the system design are more than 2. In this multi-state representation of the system design, one state would be the ideal state of system design, one state would be the non-ideal state of the system design, and there would be one or more degraded states where the system design meets the requirement of interest in a degraded manner. Furthermore, in this scenario, verification activities must ensure that if the system design is in the non-ideal state, then it is in at least a degraded state after verification. We conjecture that in the scenario described above, frequent verification would be optimal when the organization's baseline confidence in design activities is low. Such scenario is not included in the scope of this paper, but is left for future work.

In iteration 3, we increase the organization's baseline confidence in the correctness of its design activities back to $k_1 = 0.95,$ while increasing $k_2 = 40$. By increasing $k_2 = 40$, we reduce the rate at which the organization's confidence in the correctness of its design activities decays with each consecutive phase the organization does not verify the design. Figure 5 graphs the optimal verification strategy for each case for the new set of parameter values. Comparing Figure

4 with Figure 5, we see that the optimal verification strategy for cases 2 and 3 is unaffected by the rate of decay in the organization's confidence in the correctness of its design activities. Whereas, for cases 1 and 4, verification becomes more infrequent when the organization's confidence in the correctness of its design activities decays slowly.

The results of iteration 3 lead us to conclude that a low decay rate, or a high value of k_2, impedes frequent verification being an optimal strategy. Consider, for example, the requirement of installing appropriate control software on a utility tractor. Due to the repeated nature of the task over multiple design versions, the organization can be confident of the steps involved in the installation of the software. Here, k_2 will be high, implying a low decay rate, and our model would suggest that the organization verify whether the control software has been installed intermittently, instead of continuously. In contrast, consider the design of the fuel ignition system on a hypersonic missile [63]. Igniting the fuel at hypersonic speeds is difficult and the design of the fuel ignition system on a hypersonic missile is a challenging problem [64]. In this scenario, the organization's confidence in its design activities decays rapidly due to the epistemic nature of the problem itself, and whether the design activities actually result in an ignition system that works correctly at hypersonic speeds. In this scenario, k_2 will be low, and our model suggests that the organization verify frequently to confirm the correctness of its design.

Based on the results of iterations 1-3, we conjecture that a high baseline confidence in the correctness of the design activities and a high decay rate in this confidence is required for frequent verification to be an optimal strategy This is illustrated by iteration 4, where we set $k_1 = 0.99$ and $k_2 = 1$. Figure 6 graphs the results of iteration 4. As shown in the figure, frequent verification is

an optimal strategy for cases 1, 2 and 4. It is not so for case 3 since case 3 violates condition (10) in the first phase.

5 Application example

We now discuss how our model can be applied in practice. For this example, we borrow the case of verifying an optical instrument in a satellite presented in the work of Salado and Kannan [39]. The system is an optical instrument that consists of a telescope, a spectrometer, and a camera. The optical instrument must satisfy 3 system level requirements when verified. For this discussion, we will restrict our attention to the Modular Transfer Function (MTF) requirement. The MTF of an instrument is a measure of contrast and resolution capabilities [65]. In Salado and Kannan's example, the MTF requirement states that *the optical instrument's MTF be greater than 0.65* [39].

Salado and Kannan [39] present a verification plan for all system requirements on both the system level and the component (telescope, spectrometer and camera) level for three key decision points: i) preliminary design review (PDR), ii) critical design review (CDR), and iii) qualification review (QR). In this example, we restrict our attention to the system level design and verification activities. We expand on Salado and Kannan's example this to bring out a more detailed view of the design process *on the system level*. This detailed view is shown in Figure 7, which presents the design steps in the optical instrument's design process between the key decision points and the associated verification activity for each step that we will use for this discussion.

As shown in Figure 7, there are three major design steps we consider. In the 1st design step, all components in the optical instrument (telescope, spectrometer, and camera) are designed and their mathematical models integrated at instrument level. Here, the MTF of the optical instrument is verified with an analysis (activity v_7). In the 2nd design step, component prototypes are

manufactured and integrated into a prototype of the optical instrument. To verify the MTF of the optical instrument prototype, the MTF is measured on the center point of the prototype (activity v_2). Finally, in the 3rd design step, all actual components are fabricated and integrated into the final optical instrument design. The verification activity on the final design involves measuring the MTF on the center point of the final design (activity v_5).

We map the optical instrument example described above to our model as follows. The state of the optical instrument is broadly classified as either the instrument has an MTF value greater than 0.65, or not. If the optical instrument has an MTF value greater than 0.65, then we denote it as S_1, the ideal state, and S_0, the non-ideal state, otherwise. The design steps correspond to the phases in our model, and hence $N = 3$. In turn, this implies that in each design step, or phase, the organization's possible decisions are to execute the verification activity in that design step, or not. For example, in design step 2, or phase 2, the organization's possible decisions are to either measure the MTF on the center point of the prototype, or not. Furthermore, we will assume that the optical instrument is a critical component of the satellite. Thus, the expected cost of project failure, r_F, will be equal to the penalty the organization will have to pay the customer if the optical instrument's MTF falls below 0.65 before the end of its operational life.

To determine the value of $\varepsilon_{n,l}$, we require the values of k_1 and k_2 in addition to N. The parameter k_1 can either be set using historical data or with the help of subject matter experts. For example, if the organization has high confidence on the maturity/predictability of its design process, then k_1 can be set to a high value (> 0.85). Else, k_1 may be set to a low value. In comparison, we argue that only a low value of k_2 is suitable in this scenario. We say so since the MTF of the optical instrument heavily depends on integration factors such as misalignment of the

components, or mirrors in the spectrometer and telescope, or dust accumulated during integration. Hence, if the organization skips verification for even one phase, its confidence in the system design being in the ideal state will reduce significantly. This is captured by setting $k_2 = 1$ in our model.

The value of the parameter c_n is the total cost of executing the verification activities in phase *n.* This includes the cost of using the testing equipment and the cost of executing all verification activities in phase *n*. In comparison to the set-up cost c_n, the expected rework cost r_n will necessarily be an estimate since the cost of rework cannot be known before the system is verified. The expected rework cost r_n in phase *n* will be the organization's estimate of the labor and material costs associated with potential repairs in phase *n.* This will include estimates of the costs of disassembling the system design for rework when verification reveals an error.

The organization only needs to quantify its belief in the ideal state of the system design at the end of the PDR, or β_1, to utilize our model. One possible value of β_1 would be the organization's subjective assessment of the probability of the system design having an error at the end of the PDR.

We have graphically depicted our model for the optical instrument example discussed above in Figure 8. The intention of this figure is to facilitate the understanding of how our model maps to a verification planning problem in practice.

6 Model validity

We have developed a normative decision-theoretic model of verification in this paper. Our model was not developed using a dataset obtained from the industry and is theoretical in nature. Hence, a data-driven validation process is not applicable for our work. Instead, we validate our model with

the intention of providing a potential user with more confidence in its applicability. In this regard, hypotheses validity and logical validity are two qualitative validation methods frequently used on decision-theoretic models [66,67]. We discuss both below.

6.1 Hypothesis validity

Hypothesis validity checks if the model has adequately reproduced the connections between the elements of the *subject* being modeled [67,68]. In the context of our model, the subject is the development process for a system, particularly in terms of verification decisions throughout the development process. The evolution of the development process is dependent on the decisions made during the process. These decisions are the inputs to our model. We measure the manner in which this development process evolves with quantifiable metrics, such as time, cost, and meeting requirements. These metrics are the observable outputs from our model. Thus, the connections between the elements of the subject, in the context of our model, are the relationships between the input decisions to the model and the output metrics observed from the model.

In our normative model, the decision to execute the verification activity, or not, in each design phase is the input, while the observable attributes of the development process with respect to verification are the outputs. There are multiple output attributes from the development process with respect to verification: risk mitigation, costs, quality assurance, etc. In this paper, we have chosen three high-level costs associated with the verification activities: set up costs for verification activities, the expected cost of reworking a faulty design, and the expected cost of deploying a design with errors. The hypotheses validation step then requires us to check if we have adequately captured the relationship between the input and the outputs in our model.

We say that the organization's confidence in the correctness of its design is what connects the input of our model to its outputs. Our argument is as follows. The development process generates rich data in the form of design discussions, logs of activities, observations, and demonstrations, for example. This rich data influences the organization's understanding of the state of its design. Since the true state of the design is unknown prior to verification, the organization's understanding of the state of its design is subjective. That is, the organization does not *know* the true state of its design but can be thought of as being confident in the correctness of the design. The organization will make verification decisions based on this confidence. Since the costs of verification are set by the organization's decision, it then follows that adequately modeling the organization's confidence in the correctness of its design activities is sufficient to connect our model input to its outputs.

There are two aspects to modeling the organization's confidence: 1) quantifying the confidence, and 2) modeling the change in this confidence. To quantify the organization's confidence in the correctness of its design activities, we use belief distributions. The organization's confidence is changed by the actions the design activities. However, these activities have been abstracted away in our model. Thus, we need a parameter that adequately represents the manner in which design activities vary the organization's belief in the correctness of its design. This function is accomplished by the belief retention factor $\varepsilon_{n,l}$. To provide an additional granularity that models the organization's varying confidence in the maturity/capability of its design activities in different phases, we define the belief retention factor in terms of the baseline confidence parameter k_1 and the day of decay parameter k_2.

6.2 Logical validity

Logical validity checks if a model has been correctly converted into a numerical computer model that produces solutions [66]. There is no standard methodology for determining logical validity, but qualitative inspections have been used in the past [66]. To the best of our knowledge, the results of our model are numerically correct. However, we do contend that numerical accuracy does not necessarily imply applicability in reality. In this regard, our model makes two assumptions that leads to numerically correct but inapplicable results in those scenarios where the organization's baseline confidence in the correctness of its design maturity/capability is low throughout the design process: 1) the system design either meets the requirement of interest or not, and 2) when the system is verified, the belief in the correct state of the system design becomes absolute.

The two assumptions mentioned above, together, overlook the possibility of the system design being in more granular states during the design and verification process. Still, our model does derive a numerically correct strategy for those scenarios where the organization's baseline confidence in the correctness of its design activities is low – no verification in any phase but the last. This is so since our model suggests that even if the system is verified, the confidence of it being in the correct state will be low throughout the process, and hence it is best not to waste monetary resources on the same. However, in reality, the organization would prefer to verify its design if its baseline confidence in the correctness of its design activities is low. We conjecture that this issue can be resolved by expanding the size of the state space and by allowing a more granular increase in belief after verification activities.

7 Conclusion

Verification activities often consume a significant portion of the project budget. When verification activities are planned correctly, they help minimize rework costs by identifying errors in the system design early in the design process. However, overuse of verification activities can lead to

misallocation of limited resources, resulting in cost overrun which verification itself aimed to prevent. In this regard, frequent verification of the system design has been advocated in industry and the research literature. In this paper, we used a belief-based model to characterize the conditions under which frequent verification is optimal for a single organization that is considering the verification of a single system requirement.

The analysis of our model showed that frequent verification is often not an optimal verification strategy. Necessary conditions must be met for frequent verification of a single system requirement to be optimal. The necessary conditions are:

- The expected cost of failure must be strictly greater than the maximum possible verification cost in the final phase.
- The expected cost of rework in the next phase must be greater than the maximum possible verification cost in the current phase.
- The organization's baseline confidence in the correctness of its design activities must be high throughout the design process.
- The organization's baseline confidence must decay rapidly when the organization does not verify the design over consecutive phases.

The scenarios alluded to by the conditions above have been observed in the design of complex systems, such as aircrafts and satellites, which take the form of large scale systems engineering projects [69]. For such projects, the expected cost of failure is high, and complex correlations between various system requirements cause the expected cost of design rework to increase rapidly as the design matures. Furthermore, the organization's confidence in the correctness of its design

activities for the requirement of interest is often influenced by other design activities or related requirements.

A significant advantage of our model is that it is built on fundamental building blocks of a design process, that is, phases that consist of design and verification activities. This enables our model to be applicable for all system development process models (such as V model, waterfall model, spiral model, etc.). Though we used our model to determine the conditions under which frequent verification is optimal, our model can be used to determine optimal verification strategies for any feasible belief values during the design process.

There are several limitations to our model. First, we assumed a vertically integrated system. Vertically integrated systems are rare in practice. However, we have adopted it to use it as an approximation for systems that are mainly developed and integrated by a single organization (even if the organization purchases some parts, raw material, or components externally). Furthermore, by exploring a model with the vertical integration assumption, our work now provides a baseline with which the results of future works that relax the vertical integration assumption, can be compared to.

Another limitation of our model, as revealed in the analysis, is that two states are not sufficient to accurately model those scenarios where the organization's confidence in the correctness of its design activities is low throughout the design process. We conjecture that a multi-state model could conclude – in contrast to our results – that frequent verification is optimal when the organization's confidence in the correctness of its design activities is low. Another limitation of our model is that it is normative in nature and considers high-level parameters. This restricts the applicability of our model to analysis before the system developments begins, where verification activities have to be planned and resources allocated. In such conditions, our model can still aid engineers in

determining which system requirement requires frequent verification and which does not. However, once the project begins, it would be necessary to use a model of verification that dynamically accounts for the information generated by design and verification activities in the project. Our work is not unique in possessing these limitations, with the majority of mathematical models on verification sharing this limitation.

Taken together and despite the aforementioned limitation, our work makes important contributions to the growing body of literature on scientific foundations of systems engineering and engineering design. It introduced a normative model of belief-based decision-making in verification of system design, a conceptual and mathematical foundation which can be built upon in future research.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. CMMI-1762883 and CMMI-1762336.

Appendix

We now present an algorithm to numerically determine $D^*(\cdot)$ given that the parameter values c_n, r_n, r_F and $\varepsilon_{n,l}$ are known beforehand. In equation (2), let $\phi_{v,N,l} = \mathbb{T}_{-v,N,l} f_{N,v}$ and let $\phi_{-v,N,l} = \mathbb{T}_{-v,N,l} f_{N,-v}$. Then, in phase *N,* given $l < N$, the organization's optimal cost functions can be defined as

$$V_{N,l}(b_N, v) = \min\left(b_N \phi_{v,N,l}\right) \text{ , and} \tag{11}$$

$$V_{N,l}(b_N, -v) = \min\left(b_N \phi_{-v,N,l}\right). \tag{12}$$

Here, the minimum function returns the lowest value of its vector argument. The minimum function is redundant in the phase *N* since the dot product between b_N and $\phi_{.,N,l}$ results in a vector with a single element, but the above formulation will prove useful for remaining phases.

In phase $N-1$, given $l < N-1$, the organization's optimal cost function for its decision to verify can be defined as

$$V_{N-1,l}(b_N, v) = b_{N-1}\mathbb{T}_{-v,N-1,l} f_{N-1,v} + \min\{V_{N,N-1}(b_{N-1}\mathbb{T}_v, v), V_{N,N-1}(b_{N-1}\mathbb{T}_v, -v)\}$$

$$\Rightarrow V_{N-1,l}(b_N, v) = \min\{b_{N-1}\mathbb{T}_{-v,N-1,l} f_{N-1,v} + b_{N-1}\mathbb{T}_v\, \phi_{v,N,l} \;,\; b_{N-1}\mathbb{T}_{-v,N-1,l} f_{N-1,v} + b_{N-1}\mathbb{T}_v\, \phi_{-v,N,l}\}$$

$$\Rightarrow V_{N-1,l}(b_N, v) = \min\{b_{N-1}\cdot(\mathbb{T}_{-v,N-1,l} f_{N-1,v} + \mathbb{T}_v\, \phi_{v,N,l}) \;,\; b_{N-1}\cdot(\mathbb{T}_{-v,N-1,l} f_{N-1,v} + \mathbb{T}_v\, \phi_{-v,N,l})\}.$$

Define the matrix $\phi_{v,N-1,l} = [(\mathbb{T}_{-v,N-1,l} f_{N-1,v} + \mathbb{T}_v\, \phi_{v,N,l}) \;,\; (\mathbb{T}_{-v,N-1,l} f_{N-1,v} + \mathbb{T}_v\, \phi_{-v,N,l})]$. Then, the organization's optimal cost function for its decision to verify can be defined as

$$V_{N-1,l}(b_{N-1}, v) = \min\,(b_{N-1}\, \phi_{v,N-1,l}). \tag{13}$$

Similarly, define the matrix $\phi_{-v,N-1,l} = [\mathbb{T}_{-v,N-1,l}\, \phi_{v,N,l} \;,\; \mathbb{T}_{-v,N-1,l}\, \phi_{-v,N,l}]$. It then follows

$$V_{N-1,l}(b_{N-1}, -v) = \min\,(b_{N-1}\, \phi_{-v,N-1,l}). \tag{14}$$

Proceeding in the same manner as the one presented above for phase $N-1$, assume that the matrices $\phi_{v,n+1,l}$ and $\phi_{-v,n+1,l}$ have been previously computed for all $l < n+1$. Then, given $l < n$, define the matrices $\phi_{v,n,l} = [(\mathbb{T}_{-v,n,l} f_{n,v} + \mathbb{T}_v\, \phi_{v,n+1,l}), (\mathbb{T}_{-v,n,l} f_{n,v} + \mathbb{T}_v\, \phi_{-v,n+1,l})]$ and

$\phi_{-v,n,l} = [\mathbb{T}_{-v,n,l}\, \phi_{v,n+1,l} \;,\; \mathbb{T}_{-v,n,l}\, \phi_{-v,n+1,l}]$. It then follows that the organization's optimal cost functions in phase *n* can be defined as

$$V_{n,l}(b_n, v) = \min\left(b_n\, \phi_{v,n+1,l}\right), \text{ and} \tag{15}$$

$$V_{n,l}(b_n, -v) = \min\left(b_n\, \phi_{-v,n+1,l}\right) . \tag{16}$$

Once the matrices $\phi_{v,n,l}$ and $\phi_{-v,n,l}$ have been computed for all valid pair of values of *n* and *l,* determining the optimal verification strategy $D^*(\cdot)$ is straightforward. Consider again the single set of belief vectors $\{\hat{b}_n\}_{n\in\{1,\ldots,N\}}$, where $\hat{b}_n$ denotes the organization's belief vector in phase *n* that results from the organization following the optimal verification strategy in phases $1,\ldots,n-1$. Since l and $\hat{b}_n$ can be determined beforehand by using $D^*(1,\hat{b}_1),\ldots,D^*(n-1,\hat{b}_1)$, it follows that

$$D^*(n,\hat{b}_1) = \arg\min_{d_n\in\{v,-v\}}\{\min(\hat{b}_n\phi_{d_n,n,l})\} . \tag{17}$$

The set of all possible initial belief vectors b_1 is uncountable. Hence, $D^*(n,b_1)$ must be computed for a finite set of belief vectors $\psi = \{b_1^1,\ldots,b_1^M\}$ that reasonably discretizes the space of all possible belief vectors b_1. Table V outlines the solution algorithm to numerically determine $D^*(\cdot)$ assuming the set of initial belief vectors ψ is already known.

Table V: Algorithm to determine optimal verification strategy

Initialize c_n, r_n, r_F, $\varepsilon_{n,l}$, $f_{n,v}$, $f_{n,-v}$ and ψ	
Set l = 0 and iterate until l < N	
1	Set $\phi_{v,N,l} = \mathbb{T}_{-v,N,l} f_{N,v}$ and $\phi_{-v,N,l} = \mathbb{T}_{-v,N,l} f_{N,-v}$
2	Set $l = l + 1$
Set n = N – 1 and iterate until n > 0	
3	*For each update on value of n, set l = 0 and iterate until l < n*
3a	Set $\phi_{v,n,l} = [(\mathbb{T}_{-v,n,l} f_{n,v} + \mathbb{T}_v\, \phi_{v,n+1,l}), (\mathbb{T}_{-v,n,l} f_{n,v} + \mathbb{T}_v\, \phi_{-v,n+1,l})]$ and $\phi_{-v,n,l} = [\mathbb{T}_{-v,n,l}\, \phi_{v,n+1,l}, \mathbb{T}_{-v,n,l}\, \phi_{-v,n+1,l}]$

3b	Set $l = l + 1$. If $l < n$, then return to 3a, else proceed
3c	Update $n = n - 1$ and return to check condition on n
Do for each $b_1^x \in \psi$	
4	Set $b_1 = b_1^x$, $l = 0$
5	*Set n = 1 and iterate until n = N*
5a	Set $b_n = b_1$
5b	Set $D^*(n, \hat{b}_1) = \arg \min_{d_n \in \{v, -v\}} \{\min(\hat{b}_n \phi_{d_n, n, l})\}$
5c	Set $b_{n+1} = \begin{cases} b_n \mathbb{T}_{-v,n,l} & \text{if } D^*(n, b_1) = -v \\ b_n \mathbb{T}_v & \text{if } D^*(n, b_1) = v \end{cases}$
5d	If $D^*(n, b_1) = v$, then set $l = n$.
5e	Set $n = n + 1$ and return to 5.

The algorithm presented above was implemented using MATLAB©. However, it can also be implemented in general purpose programming languages, such as Python™. We suggest the algorithm be implemented in languages that have libraries to support matrix operations. This could significantly reduce the effort required to code the algorithm.

References

1. Lake JG. 4 V & V in Plain English. Paper presented at: INCOSE International Symposium1999.
2. INCOSE. *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities.* version 4.0 ed. Hoboken, NJ, USA: John Wiley and Sons, Inc.; 2015.
3. Salado A, Kannan H. A mathematical model of verification strategies. *Systems Engineering.* 2018;21:583-608.
4. Shabi J, Reich Y. Developing an analytical model for planning systems verification, validation and testing processes. *Advanced Engineering Informatics.* 2012;26(2):429-438.
5. Shabi J, Reich Y, Diamant R. Planning the verification, validation, and testing process: a case study demonstrating a decision support model. *Journal of Engineering Design.* 2017;28(3):171-204.
6. Engel A. *Verification, validation, and testing of engineered systems.* Vol 73: John Wiley & Sons; 2010.

7. Barad M, Engel A. Optimizing VVT strategies: a decomposition approach. *Journal of the Operational Research Society.* 2006;57(8):965-974.
8. Nagano S. Space systems verification program and management process: Importance of Implementing a Distributed-Verification Program with Standardized Modular-Management Process. *Systems Engineering.* 2008;11(1):27-38.
9. Chang T-f, Danylyzsn A, Norimatsu S, et al. " Continuous verification" in mission critical software development. Paper presented at: Proceedings of the Thirtieth Hawaii International Conference on System Sciences1997.
10. Klingstam P, Olsson B-G. Using simulation techniques for continuous process verification in industrial system development. Paper presented at: 2000 Winter Simulation Conference Proceedings (Cat. No. 00CH37165)2000.
11. Maropoulos PG, Ceglarek D. Design verification and validation in product lifecycle. *CIRP annals.* 2010;59(2):740-759.
12. Loch CH, Terwiesch C, Thomke S. Parallel and sequential testing of design alternatives. *Management Science.* 2001;47(5):663-678.
13. Chalupnik MJ, Wynn DC, Clarkson PJ. Approaches to mitigate the impact of uncertainty in development processes. Paper presented at: DS 58-1: Proceedings of ICED 09, the 17th International Conference on Engineering Design, Vol. 1, Design Processes, Palo Alto, CA, USA, 24.-27.08. 20092009.
14. Agarwal H, Renaud JE, Preston EL, Padmanabhan D. Uncertainty quantification using evidence theory in multidisciplinary design optimization. *Reliability Engineering & System Safety.* 2004;85(1-3):281-294.
15. Samson S, Thoomu S, Fadel G, Reneke J. Reliable design optimization under aleatory and epistemic uncertainties. Paper presented at: ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference2009.
16. Zhuang X, Pan R. Epistemic uncertainty in reliability-based design optimization. Paper presented at: 2012 Proceedings Annual Reliability and Maintainability Symposium2012.
17. Eifler T, Engelhardt R, Mathias J, Kloberdanz H, Birkhofer H. An assignment of methods to analyze uncertainty in different stages of the development process. Paper presented at: ASME 2010 International Mechanical Engineering Congress and Exposition2010.
18. Morse E, Dantan J-Y, Anwer N, et al. Tolerancing: Managing uncertainty from conceptual design to final product. *CIRP Annals.* 2018;67(2):695-717.
19. Wynn DC, Grebici K, Clarkson PJ. Modelling the evolution of uncertainty levels during design. *International Journal on Interactive Design and Manufacturing (IJIDeM).* 2011;5(3):187.
20. Schlosser J, Paredis CJ. Managing multiple sources of epistemic uncertainty in engineering decision making. *SAE Transactions.* 2007:1340-1352.
21. Sentz K, Ferson S. *Combination of evidence in Dempster-Shafer theory.* Vol 4015: Citeseer; 2002.
22. Tosney WF, Pavlica S. Satellite verification planning: Best practices and pitfalls related to testing. Paper presented at: Environmental Testing for Space Programmes2004.
23. Engel A, Barad M. A methodology for modeling VVT risks and costs. *Systems Engineering.* 2003;6(3):135-151.
24. Engel A, Last M. Modeling software testing costs and risks using fuzzy logic paradigm. *Journal of Systems and Software.* 2007;80(6):817-835.

25. Goel AL, Okumoto K. Time-dependent error-detection rate model for software reliability and other performance measures. *IEEE Transactions on Reliability.* 1979;28(3):206-211.
26. Ha AY, Porteus EL. Optimal timing of reviews in concurrent design for manufacturability. *Management Science.* 1995;41(9):1431-1447.
27. Hossain SA, Dahiya RC. Estimating the parameters of a non-homogeneous Poisson-process model for software reliability. *IEEE Transactions on Reliability.* 1993;42(4):604-612.
28. McGarry F, Page G. Performance evaluation of an independent software verification and integration process. *NASA Goddard, Greenbelt, MD, SEL Sill 0.* 1982.
29. Powell PB. Software validation, verification, and testing technique and tool reference guide. 1982.
30. Tahera K. *The role of testing in engineering product development processes*, The Open University; 2014.
31. Tahera K, Earl CF, Eckert CM. Integrating virtual and physical testing to accelerate the engineering product development process. *IJITM.* 2014;13(2/3):154-175.
32. Thomke S, Bell DE. Sequential testing in product development. *Management Science.* 2001;47(2):308-323.
33. Wallace DR, Fujii RU. Software verification and validation: an overview. *Ieee Software.* 1989;6(3):10-17.
34. Yamada S, Ichimori T, Nishiwaki M. Optimal allocation policies for testing-resource based on a software reliability growth model. *Mathematical and Computer Modelling.* 1995;22(10-12):295-301.
35. Yamada S, Ohba M, Osaki S. S-shaped reliability growth modeling for software error detection. *IEEE Transactions on Reliability.* 1983;32(5):475-484.
36. Cook TD, Reichardt CS. Qualitative and quantitative methods in evaluation. 1979.
37. Lee AS. A scientific methodology for MIS case studies. *MIS quarterly.* 1989:33-50.
38. McCutcheon DM, Meredith JR. Conducting case study research in operations management. *Journal of Operations Management.* 1993;11(3):239-256.
39. Salado A, Kannan H. Elemental patterns of verification strategies. *Systems Engineering.* 2019;22(5):370-388.
40. Salado A, Kannan H, Farkhondehmaal F. Capturing the Information Dependencies of Verification Activities with Bayesian Networks. Conference on Systems Engineering Research (CSER); 2018; Charlottesville, VA, USA.
41. Xu P, Salado A. A Concept for Set-based Design of Verification Strategies. Paper presented at: INCOSE International Symposium2019; Orlando, FL, USA.
42. Kulkarni AU, Salado A, Wernz C, Xu P. Is Verifying Frequently an Optimal Strategy? A Belief-Based Model of Verification. Paper presented at: ASME 2020 International Design Engineering Technical Conference & Computers and Information in Engineering Conference (IDETC/CIE 2020)2020; St. Louis, MO, (USA).
43. Beheshti R. Design decisions and uncertainty. *Design Studies.* 1993;14(1):85-95.
44. Xenakis I, Arnellos A. Reducing uncertainty in the design process: the role of aesthetics. Paper presented at: 8th International Conference on Design and Emotion2012.
45. De Weck O, Eckert CM, Clarkson PJ. A classification of uncertainty for early product and system design. Paper presented at: DS 42: Proceedings of ICED 2007, the 16th International Conference on Engineering Design, Paris, France, 28.-31.07. 20072007.

46. Kreye ME, Goh YM, Newnes LB. Manifestation of uncertainty-A classification. Paper presented at: DS 68-6: Proceedings of the 18th International Conference on Engineering Design (ICED 11), Impacting Society through Engineering Design, Vol. 6: Design Information and Knowledge, Lyngby/Copenhagen, Denmark, 15.-19.08. 20112011.
47. Daskilewicz MJ, German BJ, Takahashi TT, Donovan S, Shajanian A. Effects of disciplinary uncertainty on multi-objective optimization in aircraft conceptual design. *Structural and Multidisciplinary Optimization.* 2011;44(6):831-846.
48. Kirby MR, Mavris DN. Forecasting technology uncertainty in preliminary aircraft design. *SAE transactions.* 1999:1388-1399.
49. Padulo M. Computational engineering design under uncertainty: an aircraft conceptual design perspective. 2009.
50. Blanchard BS, Fabrycky WJ. *Systems engineering and analysis.* Vol 4: Prentice Hall New Jersey;; 1990.
51. Ahmadi R, Wang RH. Managing development risk in product design processes. *Operations Research.* 1999;47(2):235-246.
52. Arundachawat P, Roy R, Al-Ashaab A, Shehab E. Design rework prediction in concurrent design environment: current trends and future research directions. Paper presented at: Proceedings of the 19th CIRP Design Conference–Competitive Design2009.
53. Browning TR, Eppinger SD. Modeling impacts of process architecture on cost and schedule risk in product development. *IEEE transactions on engineering management.* 2002;49(4):428-442.
54. Salado A. Applying tradespace exploration to verification engineering: From practice to theory and back again. Paper presented at: Conference on Systems Engineering Research (CSER)2016; Huntsville, AL (USA).
55. Abbas AE, Cadenbach AH. On the Use of Utility Theory in Engineering Design. *IEEE Systems Journal.* 2018;12(2):1129-1138.
56. Hazelrigg GA. A Framework for Decision-Based Engineering Design. *Journal of Mechanical Design.* 1998;120(4):653-658.
57. Collopy P. Aerospace system value models: Survey and observations. Paper presented at: AIAA Space 2009 Conference2009; Pasadena, CA.
58. Collopy PD, Hollingsworth PM. Value-Driven Design. *Journal of Aircraft.* 2011;48(3):749-759.
59. Salado A, Kannan H. Properties of the Utility of Verification. IEEE International Symposium in Systems Engineering; 2018; Rome, Italy.
60. Bradley SP, Hax AC, Magnanti TL. Applied mathematical programming. 1977.
61. Littman ML. A tutorial on partially observable Markov decision processes. *Journal of Mathematical Psychology.* 2009;53(3):119-125.
62. Cassandra AR, Littman ML, Zhang NL. Incremental pruning: A simple, fast, exact method for partially observable Markov decision processes. *arXiv preprint arXiv:13021525.* 2013.
63. Baurle R, Mathur T, Gruber M, Jackson K. A numerical and experimental investigation of a scramjet combustor for hypersonic missile applications. Paper presented at: 34th AIAA/ASME/SAE/ASEE joint propulsion conference and exhibit1998.
64. Van Wie DM, D'Alessio SM, White ME. Hypersonic airbreathing propulsion. *Johns Hopkins APL technical digest.* 2005;26(4):430-437.

65. Hua F, Johnson P, Sazonova N, Lopez-Meyer P, Schuckers S. Impact of out-of-focus blur on face recognition performance based on modular transfer function. Paper presented at: 2012 5th IAPR International Conference on Biometrics (ICB)2012.
66. Gass SI. Decision-aiding models: validation, assessment, and related issues for policy analysis. *Operations Research.* 1983;31(4):603-631.
67. Mihram GA. Some practical aspects of the verification and validation of simulation models. *Journal of the Operational Research Society.* 1972;23(1):17-29.
68. McCarl BA. Model validation: an overview with some emphasis on risk models. *Review of Marketing and Agricultural Economics.* 1984;52(430-2016-31544):153-173.
69. Kozlak SJ, White ED, Ritschel JD, Lucas B, Seibel MJ. Analyzing Cost Growth at PROGRAM STAGES FOR DOD AIRCRAFT. *Defense Acquisition Research Journal: A Publication of the Defense Acquisition University.* 2017;24(3).

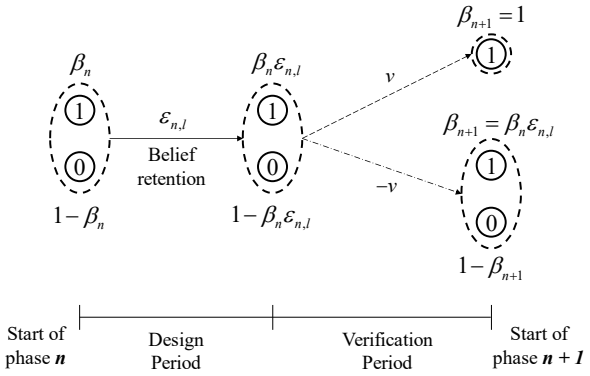

Figure 1: Change in organization's belief in phase n

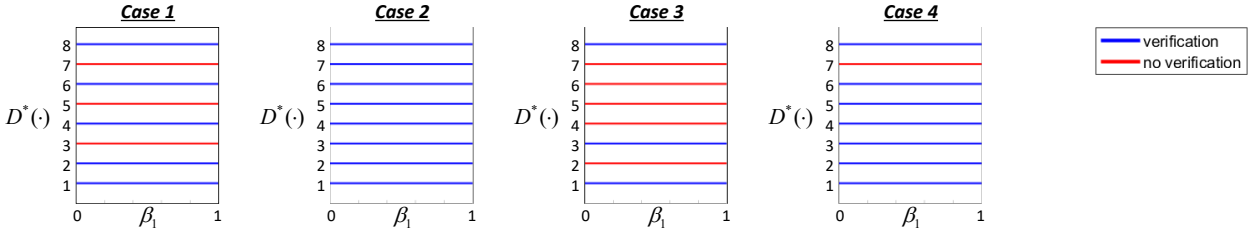

Figure 2: Optimal verification strategy for the initial set of parameter values

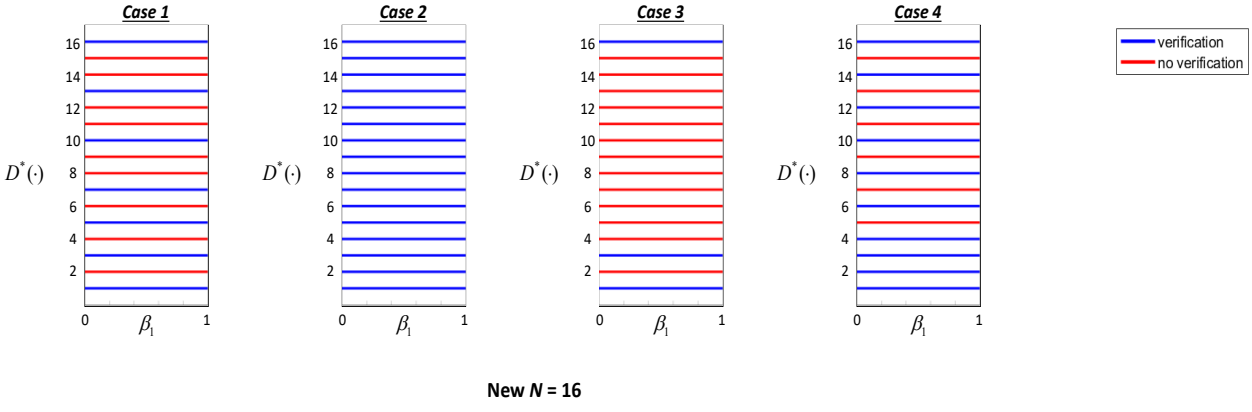

Figure 3: Optimal verification strategy for iteration 1

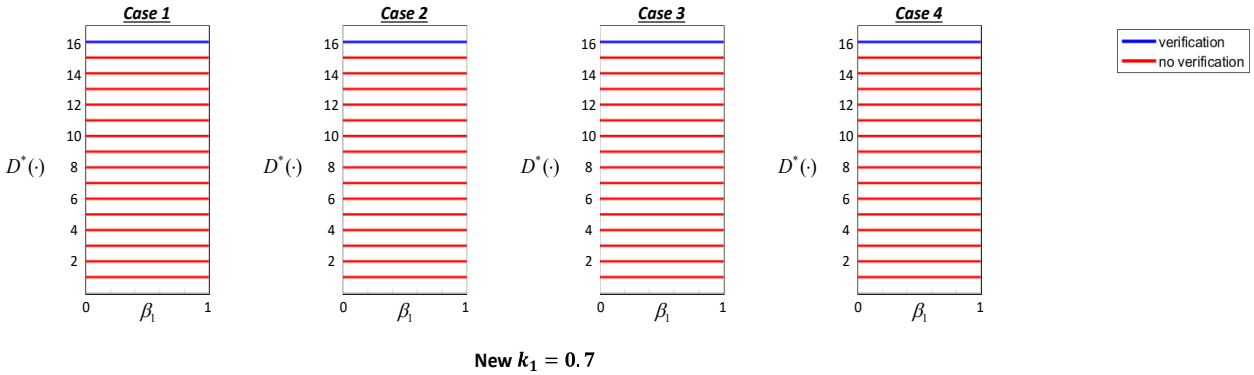

Figure 4: Optimal verification strategy for iteration 2

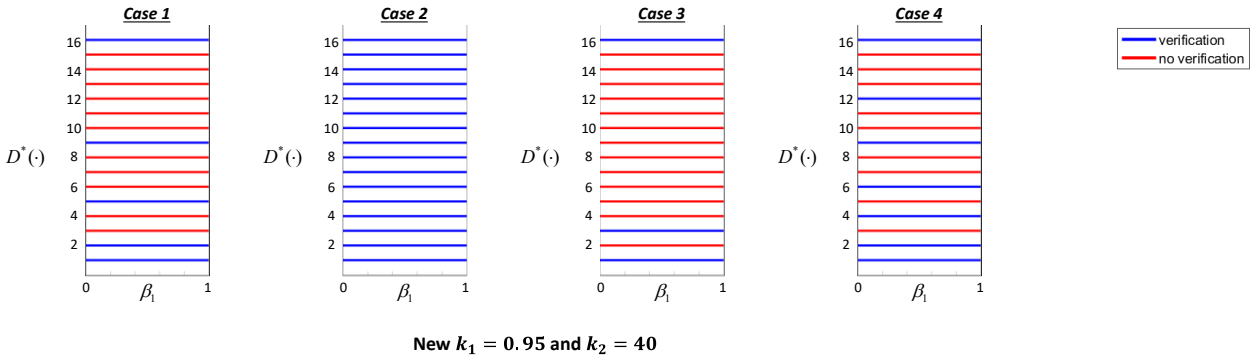

Figure 5: Optimal verification strategy for iteration 3

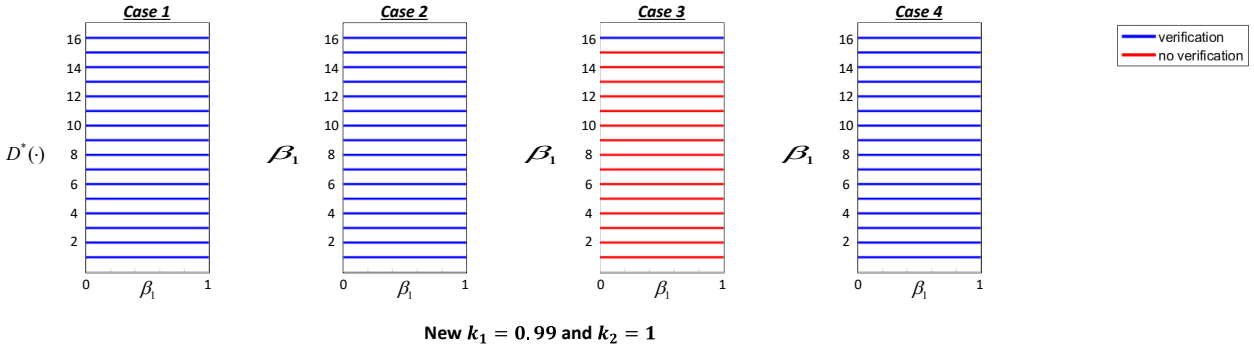

Figure 6: Optimal verification strategy for iteration 4

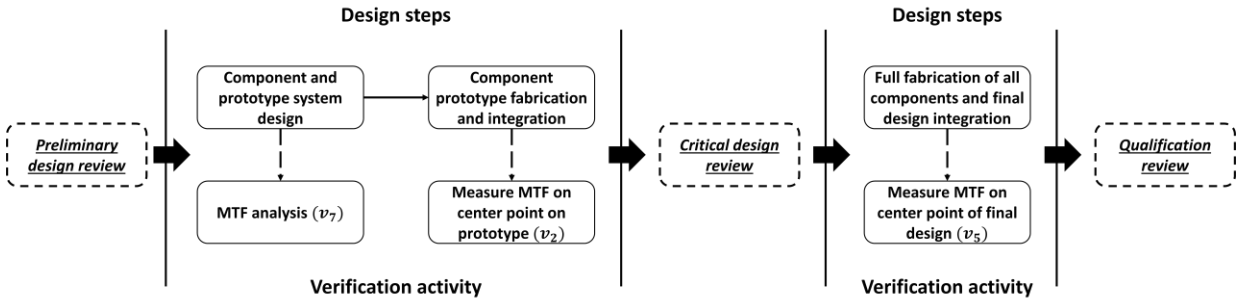

Figure 7: MTF related design and verification activities for the optical instrument

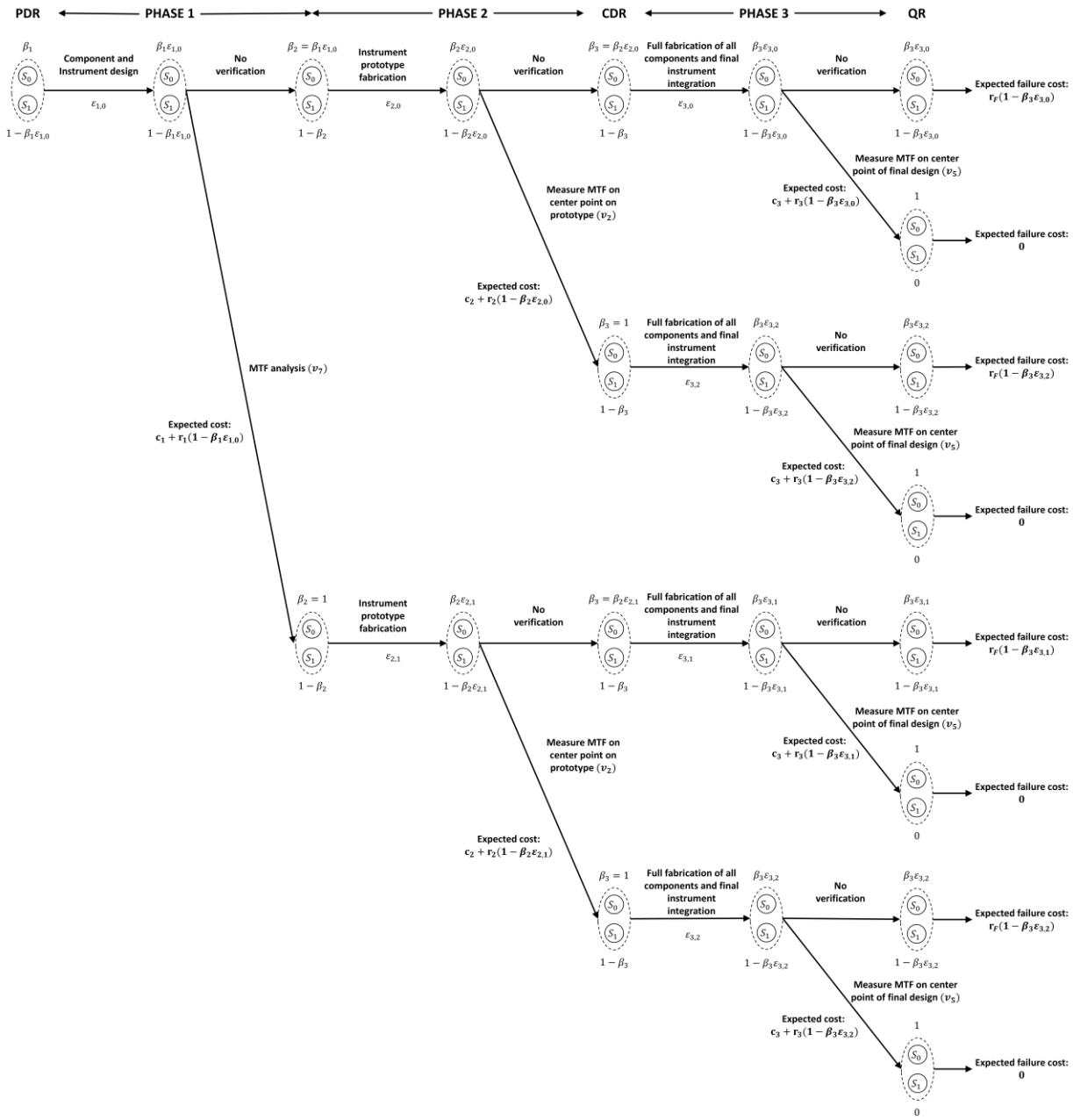

Figure 8: Graphical representation of the optical instrument example

Biographies

Dr. Aditya U. Kulkarni is a Postdoctoral Associate in the Grado Department of Industrial and Systems Engineering at Virginia Tech. He obtained his Ph.D. in Industrial Engineering from Virginia Tech in 2018. He holds an M.S. in Industrial Engineering from Virginia Tech (2012) and a B.Tech in Mechanical Engineering from National Institute of Technology, Karnataka (2010). Dr. Kulkarni's research focuses on leveraging stochastic modeling and game theory to solve research problems in in Systems Engineering. He's currently working on determining optimal verification strategies in systems engineering projects.

Dr. Alejandro Salado is an assistant professor in the Grado Department of Industrial and Systems Engineering and the Director of its Systems Engineering program at Virginia Tech. He conducts research in problem formulation, design of verification and validation strategies, model-based systems engineering, and engineering education. Before joining academia, Dr. Salado spent over 10 years in the space industry, where he held positions as systems engineer, chief architect, and chief systems engineer in manned and unmanned space systems of up to $1B in development cost. He has published over 75 technical papers, and his research has received federal funding from the National Science Foundation (NSF), the Naval Surface Warfare Command (NSWC), the Naval Air System Command (NAVAIR), and the Office of Naval Research (ONR), among others. He is a recipient of the NSF CAREER Award, the International Fulbright Science and Technology Award, the Omega Alpha Association's Exemplary Dissertation Award, and several best paper awards. Dr. Salado holds a BS/MS in electrical and computer engineering from the Polytechnic University of Valencia, a MS in project management and a MS in electronics engineering from the Polytechnic University of Catalonia, the SpaceTech MEng in space systems engineering from the

Technical University of Delft, and a PhD in systems engineering from the Stevens Institute of Technology. Alejandro is a member of INCOSE and a senior member of IEEE and AIAA.

Peng Xu is a Ph.D. candidate in the Grado Department of Industrial and Systems Engineering at Virginia Tech. He received his MS degree in Mechanical Engineering from National Cheng Kung University in 2015 and his Bachelor degree in Mechanical Engineering from Shandong University in 2013. His research interests include complex system diagnosis, sequential decision making, and engineering statistics.

Dr. Christian Wernz is an Associate Professor in the Department of Health Administration at Virginia Commonwealth University (VCU). Dr. Wernz earned his PhD in Industrial Engineering and Operations Research from the University of Massachusetts Amherst, and received his bachelor's and master's degrees in Business Engineering from the Karlsruhe Institute of Technology (KIT) in Germany. His research focuses on data and decision analytics in complex systems, such as healthcare and systems engineering. In healthcare, he studies systems across different levels, ranging from hospital operations to health policy, with a particular interest in health IT and medical technologies. In systems engineering, he focuses on verification and multi-firm interactions. He has developed and continues to work on the forefront of multiscale decision theory. His work has been funded by the National Science Foundation (NSF), the Agency for Healthcare Research & Quality (AHRQ), the Harvey L. Neiman Health Policy Institute, VCU Health, Carilion Clinic, Rolls Royce, and Dell, among others.