Received 9 April 2021; | Revised 27 July 2021; | Accepted 22 October 2021

DOI: xxx/xxxx

ARTICLE TYPE

Modeling Correction Activities in the Context of Verification Strategies

Peng Xu[1] | Alejandro Salado[2]

[1]Grado Department of Industrial and Systems Engineering, Virginia Tech, Blacksburg, VA, USA

[2]Department of Systems and Industrial Engineering, The University of Arizona, Tucson, AZ, USA

Correspondence

*Alejandro Salado, Department of Systems and Industrial Engineering, The University of Arizona, Tucson, AZ, USA Email: alejandrosalado@arizona.edu

Abstract

Correction activities, which can take the form of redesign, rework, or repair, are essential to system development. Whereas verification activities provide information about the state of the system, correction activities modify the state of the system to facilitate its correct operation. However, existing approaches to modelling and optimizing verification strategies take a simplistic approach to correction activities. Specifically, correction activities are modeled as an expected cost to achieve a desired confidence level after a verification activity has failed and are inherent to such verification activities. In this paper, we present a modeling paradigm based on Bayesian networks that captures the effects of different types of correction activities. This modeling paradigm allows for the integration of verification and correction decisions under a common framework. The modeling paradigm is illustrated in the notional case of a communication system.

KEYWORDS:

system verification, verification planning, rework, repair, redesign, uncertain evidence, Bayesian network, causal interaction

1 | INTRODUCTION

System verification is defined as the process that evaluates whether a system or its components fulfill their requirements[1]. Verification is also used to inform design choices and optimize performance. System verification is often planned and implemented as a strategy of verification activities (VA), which can be executed at different developmental phases and on different system configurations[2]. However, it is inadequate to only rely on verification for the successful development of a system because system verification can only identify design defects and errors–it cannot solve them. If these defects and errors reveal a fundamental system deficiency, the development team can perform correction activities (CA) to correct the defects and errors[1]. CAs usually take the form of rework, repair, or redesign, depending on whether the activity encompasses substitution of a faulty part, modification of the product, or modification of the product's design[3,1,4].

Given the uncertainty surrounding the development of large systems, the need for CAs is inevitable in practice[5,6,7]. These activities have a critical impact on project cost and schedule[8], potentially representing between 66 and 80% of the total development time[9,10], and between 30 and 65% of the total design hours on a project[11].

Despite the strong coupling between VAs and CAs, and the criticality of CAs, existing research focuses on the modeling and selection of just one type of activity, significantly simplifying or even ignoring the other activity. For example, CAs are treated as direct properties of VAs (not as independent activities) in research that aims to develop system verification models or to design methods to generate verification strategies[1,12,13,14,15,16]. Similarly, VAs are ignored in work that aims to better understand CA

models or analysis methods[10,17]. While these simplifications are useful for reducing the complexity of the research problem, they impose a significant limitation to the fidelity of the resulting models with respect to the practice of verification. As far as we know, a quantitative model to support the design of verification strategies that captures the interaction of VAs and CAs as independent activities is lacking.

To overcome this limitation, this paper presents a hybrid verification and correction framework that integrates CAs in verification strategies as stand-alone activities. First, as proposed in prior work[18], Bayesian networks (BN) and probabilistic causal interaction (PCI) models are used to capture the mutual dependencies between system parameters and verification activities. Second, we augment the prior modeling framework by incorporating different types of evidence modeling. This is used to capture the impact that different types of CAs have on the system parameters and the confidence of their state, as well as on the information expected to be acquired through the execution of verification activities. Third, we use a basic interaction loop (BIL) to integrate all VAs and CAs so that a generalized Bayesian inference process can be applied. This framework is applicable to both of the systems being newly developed (where iterative design changes and corrections informed by early prototypes often occur due to the availability of little or ill-defined data) and systems that are recurrently produced (where knowledge and data exist but performance of a specific unit is poor).

This paper extends work by Salado and Kannan, who proposed a mathematical model of verification strategies[2] and adopted the BN paradigm to construct patterns of verification strategies[18]. However, in none of those works do the authors model correction activities; they only model the relationships between VA and system parameters. The authors make a simplistic assumption about the effects of correction activities, assuming no influence on the structure of the BN. In the present paper, we extend such work by incorporating constructs to the BN that allows for modeling and incorporating the effects of correction activities.

The rest of this paper is organized as follows: Section 2 reviews the background of BN-based systems, uncertain evidence, and CAs in system development. Section 3 introduces BN and probabilistic causal interaction (PCI) models. Section 4 describes the proposed hybrid verification and correction framework to integrate VAs and CAs. Section 5 presents a demonstrative case to illustrate the application of the framework. Section 6 offers our conclusions and presents some future research directions.

2 | BACKGROUND

In this section, we review three topics related to our study, including modeling complex systems with BN, uncertain evidence in BNs, and CAs in system development.

2.1 | Modeling complex systems with BNs

When developing new systems, the emergence of errors or undesirable effects that lead to dissatisfaction of the requirements is closely related to the uncertainty surrounding system development[19,20]. Here, we do not distinguish if the source of the error is aleatory or epistemic, as both are significant in system development[21]. However, because verification activities primarily target epistemic uncertainty, that is, a lack of knowledge associated with whether the system meets its requirements[22], we focus on epistemic uncertainty in the context of system verification. Many approaches have been introduced to characterize epistemic uncertainty, including belief theory[23], fuzzy theory[24], possibility theory[25,26], and generalized p-boxes[27]. Because belief theory has effectively been applied to build epistemic uncertainty models in system development[28,29], we suggest that belief theory is a promising modeling paradigm to capture the subjective nature of how engineers interpret the information provided by verification activities.

Furthermore, it is necessary to capture the belief dependencies between elements of an engineered system, because the engineered systems of interest in this paper can be conceptualized as recursively formed by lower level elements[30,31]. That is, a system error or undesirable effect may be caused by an error in one of its elements, an error in the integration of the elements, or both. Various methods have been developed to capture this aspect, such as Bayesian networks (BN) [32], the transferable belief model[33,34], the Fuzzy cognitive map[35], or the belief rule-based method[36]. Among them, BNs have been predominant in dealing with error diagnosis and have been effectively used in a broad range of fields, such as economics, accident analysis, risk analysis, and cognitive sciences[37]. BNs can directly capture uncertainty about the system state as beliefs and all dependencies between system parameters and between system parameters and VAs as a network, make them promising as a fundamental method to model verification strategies [18].

A BN is a probabilistic graphical model that represents a set of random variables and their conditional dependencies via a directed acyclic graph[38]. Structurally, a BN consists of nodes and arrows between the nodes. Nodes represent the random variables, which we are interested in gaining knowledge about, and arrows between nodes represent the probabilistic dependencies between the nodes[39]. The basic BN has been extended to serve more sophisticated modeling needs, such as dynamic BN (which captures the change in beliefs as a time series model)[40,41], object-oriented BN (which allows large-scale systems to be described in terms of inter-related objects)[42], and subjective BN (which generalizes BNs to the theory of subjective logic)[43,44]. We leverage this flexibility in this work to incorporate various types of CAs to the verification strategy models developed in[2,18].

Even though BNs have been widely applied to update beliefs in complex systems, there is one limitation in the representation of BNs. As the conditional probabilities of a BN is generated by enumerating all possible state combinations of network parameters, it is hard to interpret sources of uncertainty about the parameters, especially when the parameter has many parent ones. This problem makes it hard to understand and update the prior probabilities of a parameter directly. Therefore, as shown in later sections, probabilistic causal interaction (PCI) models are leveraged to help interpret the sources of uncertainty in this study.

2.2 | Uncertain evidence in BNs

In BN, and Bayesian analysis in general, the term 'evidence' corresponds to new information about parameters of a distribution coming from the data[45]. This evidence plays a central role in Bayesian inference by triggering a belief update. In this paper, evidence captures the information provided by the results of verification activities[18].

Two types of evidence may be employed: hard (certain) evidence and uncertain evidence. We also distinguish between three types of uncertain evidence, including soft evidence, not-fixed probabilistic evidence, and virtual evidence[46]. Hard evidence refers to the instantiation of a set of variables to their values without uncertainty[46]. Soft evidence[47] (also called fixed probabilistic evidence) specifies the probability distributions of some variables. These distributions cannot be changed by further information. Not-fixed probabilistic evidence also specifies the distributions of variables, which is similar to soft evidence. However, these distributions can be modified by the propagation of new evidence. Virtual evidence[32] (also called likelihood evidence) is represented by a likelihood ratio (LR) that represents the strength of confidence toward the observed event[48]. While the two types of probabilistic evidence directly represent uncertainty as distributions, the uncertainty in virtual evidence comes from the unreliability or imprecision of the source of information[46].

Hard evidence is arguably the most common type in BN applications, but applications of the other three types of *uncertain* evidence have also been attempted. For example, Gowadia et al.[49] used soft evidence to develop a framework for intrusion detection. Subramanya and Bilmes[50] used virtual evidence to reduce the amount of supervisory training data for automatic speech recognition systems. Corradi et al.[51] used soft evidence to classify individuals for forensic purposes. However, to our knowledge, *uncertain* evidence has not been used in systems engineering research, particularly in relation to the design of verification strategies. In this paper, we will leverage *uncertain* evidence to capture the effects that different CAs may have on BN.

2.3 | CAs in system development

In this paper, we define CAs as those that correct errors or defects that are found during system development. We distinguish between three types of CAs: rework, repair, and redesign. Different definitions appear for these types of activities in the literature. For example, rework is defined in[3] as "an action on a non-conforming product or service to make it conform to the requirements," while repair is defined as "an action on a non-conforming product or service to make it acceptable for the intended use." In addition, many alternative definitions are also provided for CAs in previous work. In[52], rework is defined as the repetition of an activity at the same scope and abstraction level. In[53], rework is defined as "the design change whose implementation alters work that was previously done upstream and downstream," and in[54], rework is defined more generally as a task that needs to be redone because of a change. Therefore, dedicated definitions will be adopted for this paper, which will be presented later in Section 4.2.1.

We distinguish between two different threads in research related to CAs. The first one addresses the development of methods that help reduce or even prevent the need to execute CAs. These include, among others, the use of set-based practices[10], collective learning approaches[55], the use of design visualization models to facilitate early inspection[56], overlapping strategies to perform implementation tasks concurrently[57,58], early testing to avoid cascading effects and late correction (which is generally more expensive)[59,60], and the design structure matrix to isolate and mitigate rework by capturing interdependencies[61]. The second

thread considers that some CAs are inevitable and focuses on reducing the impact (e.g., cost) of CAs. Because uncertainty seem to be the major cause of the need for CAs[62,53,63], these methods primarily leverage modeling and optimization. For example, Browning and Eppinger[64] developed a stochastic model that uses a discrete event simulation to understand the impacts of process architecture on process cost, duration, and risk. Liao[65] proposed an economic production quantity model for maintenance, production, and free-repair warranty programs to minimize total production cost. Taleizadeh et al[66] developed an imperfect, multi-product production system with rework to minimize the joint total cost of the system. While these studies provide valuable insights to manage CAs, they mainly focus on the negative impact of CAs on cost and time during system development, leaving aside their positive effect of re-gaining confidence in the correct state of the system. This aspect is central to this paper.

3 | PRELIMINARIES

3.1 | Basic system verification model

We consider that a given system can be decomposed into a set of elements $E_i, i = 1, \ldots, I$, and assume that the objective of system verification is to verify relevant requirements for these elements.[1] We conceive a basic system verification model as a set of tuples of system parameters $P_i, i = 1, \ldots, I$, and the verification activities (VA) that provide information about such system parameters, denoting the resulting verification evidence by $V_j, j = 1, \ldots, J$.[2] Using the modeling framework presented in,[18] we build a basic system verification model as a directed acyclic graph $G = (N, E)$, where N is the set of nodes, each of which represents one system parameter P_i or V_j, and E is a set of directed edges that connect the different nodes. There are three types of directed edges: $P_i \Rightarrow P_j$, $P_i \Rightarrow V_j$, and $V_i \Rightarrow V_j$, which capture the information dependencies between system parameters, between system parameters and VAs, and between VAs, respectively. There are no directed edges from VAs to system parameters because an actual system parameter is not caused by the VA; however, the result of the VAs causally depends on the system parameter. In the resulting BN, nodes representing VAs will always be treated as observable nodes (those whose value states can be observed directly) and nodes representing system parameters will be treated as hidden nodes (those whose value states cannot be observed directly, but are inferred from the values of the observable nodes).

Because the interpretation of the information provided by VAs is subjective,[67] we capture the information about system parameters as beliefs. Without loss of generality, all nodes are assumed to be binary (i.e., two node states, such as pass/fail, true/false, or compliant/non-compliant). The nature of Bayesian analysis, and of BNs by extension, allows for easy removal of this restriction and use of any number of discrete values (e.g., also adding an inconclusive state to pass and fail) or even continuous belief distributions[68], which can be used to model verification activities aimed at informing design choices or supporting performance optimization. The specific beliefs of a network node are presented as a conditional probability table (CPT). Each CPT summarizes the dependency relationships between a node and all its parent nodes.

3.2 | Probabilistic causal interaction model

Probabilistic causal interaction (PCI) models, such as the noisy OR-gate model[32], were originally proposed to alleviate the elicitation burden of CPTs[69,70,71]. In PCI models, a CPT is decomposed into a set of cause factors and the interaction between these cause factors is represented by combination rules. This mechanism aligns well with the verification strategy models used in this paper, given the causal interactions in system verification. While several PCI models exist, we select two types of PCI models, one to derive belief distributions for system parameters and one for the information yielded by VAs separately, without loss of generality.

We apply the generalized noisy OR-gate model[72,73,70] to interpret the conditional distributions of system parameters. That is, for a system parameter P_i, the interaction of all its cause factors is assumed to be the noisy OR-type.[32] Applying this model consists of three steps:

1. For each system parameter P_i, if all its parent nodes have no faults, the belief in whether P_i passes or not is estimated as a cause factor (i.e., a random variable) X_0 with a distribution $F(X_0)$. This cause factor is called *independent factor* in

[1] We note, as indicated in the introduction, that verification may also be used to inform design choices and support performance optimization. The models presented here can be used for any of those purposes.

this study. In nature, an independent factor represents the belief that an error will occur in the absence of all parent nodes modeled explicitly.

2. For each system parameter, the dependency relationships of all its parent nodes are represented by a set of cause factors $X_1; X_2; \ldots X_K$, each of which has a distribution $F.X_k/$, where $_i$ has K parent nodes and the index follows the index sequence of the parent nodes. Each $F.X_k/$ represents the belief in whether passes or not if the kth parent node fails. They are called dependent factors in this study;

3. With the noisy OR-type assumption between those cause factors, the conditional distribution is given by the formula below:

$$F.\ _i\partial X_E/ = \begin{matrix} T \\ 1 * .1 * \ P.X_0 = Fail// \ . \pm \\ .1 * \ P.X_0 = Fail// \ \ \pm \end{matrix} \begin{matrix} _{k:\ X_k \ddot{E} X_E} .1 * \ P.X_k = Fail///; \ for \ _i = Fail \\ _{k:\ X_k \ddot{E} X_E} .1 * \ P.X_k = Fail///; \ for \ _i = Pass \end{matrix} \quad (1)$$

where X_E is the set of all cause factors that are false. The sum of each conditional distribution is 1.

Second, for a VA $_j$, its results are completely determined by its parent nodes. For simplicity, it is assumed that the causal interaction of cause factors $_j$ is of noisy AND-type. So the noisy AND-gate model[32] is used for the conditional distributions of VAs in this study. Its application consists of four steps:

1. When all parent nodes pass, the belief in whether $_j$ passes or not is estimated as a cause factor X_* with a distribution $F.X_*/$. This cause factor is called negative factor in this study.

2. When all parent nodes fail, the belief in whether $_j$ fails or not is estimated as a cause factor X_+ with a distribution $F.X_+/$. This cause factor is called positive factor in this study. The dependency of both negative and positive factors is only on the parent nodes because the results of a VA are determined by its parent nodes.

3. Otherwise, the belief in whether $_j$ passes or not is assumed to follow a uniform distribution $unif\ ^Fail; Pass`$ for simplicity. That is, the belief of each possible state is 0.5. In reality, the speci c distribution may be so complex that is a hybrid function of all parental states, which is left for the future work.

4. The conditional distribution of $_j$ is given by the formula below:

$$F.\ _j\partial N_K/ = \begin{matrix} h \\ n \\ l \\ n \\ j \end{matrix} \begin{matrix} F.X_*/; \ if \ N_k = Pass; \AA N_k \ddot{E} N_K; \\ F.X_+/; \ if \ N_k = Fail; \ \AA N_k \ddot{E} N_K; \\ unif\ ^Fail; Pass`; \ otherwise: \end{matrix} \quad (2)$$

where N_k is the set of all parent nodes of $_j$.

4 | HYBRID VERIFICATION AND CORRECTION FRAMEWORK

4.1 | Modeling of VAs

Using the basic system veri cation model presented in Section 3.1, the planning and execution of VAs is modeled as follows: (1) A BN is built based on an elicited network structure and estimated prior distributions; (2) A set of hard evidence is collected after executing some veri cation activities (i.e., observable nodes); and (3) The posterior distributions of the network nodes are updated by the Bayesian rule.

To illustrate this process, a basic system veri cation model about the speed of a computer product is provided as an example. The basic system veri cation model consists of four nodes, as shown in Fig. 1. We denote by $_1$ the speed of the computer's processor, and by $_2$ the overall speed of the computer. For each of these two parameters, if their speed values reach a certain threshold, these nodes are assigned with the value of compliant. Otherwise, they would be assigned the value of non-compliant. While these two parameters cannot be observed directly, the processor's speed and the overall computer speed may be inferred from the results of VAs (denoted by $_1$ and $_2$, respectively). Similar to the system parameters, the VAs are also assumed to be binary with their own thresholds. If the processor speed test indicates that it is larger than such a threshold for $_1$, it is assigned the value of pass; it is assigned the value of fail otherwise. The same reasoning applies to $_2$.

FIGURE 1 An example BN

Once the results of a VA become available (that is, an observable node is observed), the Bayesian inference method can b applied to update the belief in the system parameters (that is, in the state of the hidden nodes). For example, if the benchmar processor speed test turns out $= \text{Pass}$, the posterior marginal probability of $_2$ can be obtained with the Bayes rule:

$$P(\ _2 = \text{Pass}) = \frac{P(\ _2)P(\ _2 = \text{Pass}\mid\ _2)}{P(\ _2 = \text{Pass})} \quad (3)$$

In practice, when the results of multiple nodes are collected simultaneously, the posterior probabilities of the nodes can be obtained with o -the-shelf methods such as the variable elimination method and the junction tree algorithm. The speci c inference process can be referred in [74].

4.2 | Modeling of CAs

4.2.1 | Basic types of CAs

The purpose of CAs is to correct the defects and errors that are found by VAs. This study focuses on three basic types of CAs that are commonly used in system development: rework, repair, and redesign. We adopt the de nitions in Table 1. Once a CA is implemented according to its correction procedure, some corresponding system elements may be changed to meet syste requirements or improve system performance. As a result, the prior beliefs of these elements are in uenced after these correctio procedures.

In general, we can consider that VAs only observe or measure attributes of the system. That is, they shape the con dence of the state of the system but the system remains unchanged after a VA has been executed.[2] However, the execution of CA does not only shape the con dence of the state of the system but also, by de nition, it a ects the state of the system. This poses a challenge to the estimation of posterior beliefs because a BN represents a joint distribution and some marginal distributions nee to be reserved after a CA. Thus, the basic veri cation strategy model presented in the previous section needs to be extended t capture these e ects. These extensions are presented in the next three subsections.

As described in the introduction of this paper, these modeling constructs are applicable to both systems being newly develope (where iterative design changes and corrections informed by early prototypes often occur due to the availability of little or ill- de ned data) and systems that are recurrently produced (where knowledge and data exist but performance of a speci c unit is poor). Only two procedural di erences are noted. First, while determining priors for models of recurring systems likely rely on historical data, models of newly developed systems need to resort to subject matter expertise [75] or fusion techniques [44]. Second, while all kinds of CAs may be necessary for a newly developed system, redesign activities are typically unnecessary for recurring systems.

4.2.2 | Impact of rework

According to the de nition of rework in Table 1, when a rework activity is executed, an existing system element is directly replaced with a replica. We make three assumptions about the impact of rework. First, the two elements share the same caus factors as well as their distributions. This assumption is sensible because both elements are built using the same design, materia and processes. Second, the new element is assumed to be in a correct state where all other elements have no errors. This

[2] In practice, some VAs can actually a ect the state of a system, such as subjecting the system to vibration testing. However, these e ects are generally expected and infrequent, so we can consider the assumption valid for the purpose of this discussion.

TABLE 1 BASIC TYPES OF CAS

CA	De nition	Example
Rework	Activity that substitutes a faulty element with a working one to correct defects. Faulty means that the element does not provide the performance that was expected (e.g., from design baselines in new developments or historical data in recurring systems).	The lens of a telescope does not o er the expected transmission performance. The cause for this degradation is a crack on its surface. It is removed from the telescope and a new lens, which is not damaged, is installed instead.
Repair	Activity that directly corrects system defects by using processes, materials, or tools not initially considered in the design while the initial system elements are retained and/or modi ed. A defect is de ned as a misalignment with respect to an expected performance (e.g., from design baselines in new developments or historical data in recurring systems).	The lens of a telescope does not o er the expected transmission performance. The cause for this degradation is a scratch on its surface. The surface is polished and covered with a new coating.
Redesign	Activity that changes the design baseline of the system.	The lens of a telescope does not o er the expected transmission performance. The cause for this unexpected performance is a limit on the ability of the material that has been used. A new lens with a di erent shape and material is designed and installed in the telescope.

sensible because rework only applies in cases in which a unit is faulty, that is, there are no inherent design, process, or materia problems with the unit. Third, while it is assumed that the new element does not present the problem identi ed in the faulty unit, it may still have other problems (both related to design and production) that have not been identi ed yet. Therefore, the impact of rework can be interpreted as the inclusion of additional sources of information to the original BN.

To illustrate this, the previous computer BN is used as a rework example. Let us assume that $_1$ fails first and it is determined that the processor is faulty. A new processor replaces the faulty one. The speed of the new processor shares the same parame $_1$ with the faulty one. This is because the new processor is an exact replica of the faulty one (except for the fault). Thus, the distribution of $_1$ remains the same as the faulty one. The impact of rework can be interpreted from two aspects. First, as the faulty element is removed after rework, the evidence about the faulty element becomes uninformative. (Here, while the faulty processor can be considered as new information that should update the belief distributions for future processors, we assume th such an e ect is negligible, being a single event.) Second, the new element is known to be working after rework. This piece of new evidence is interpreted as a belief update on the independent factor $_1$. That is, rework does not in uence other parameters and their dependency relationships.

We assume the rework information is a $LR_{Rework} = L.X_0. _1/ = Pass\partial Rework/ : L.X_0. _1/ = Fail\partial Rework/ = 99 : 1$ and virtual evidence can be used to capture this repair information in three steps. First, the LR is represented as a virtual node $_0$ attached to the independent factor $X_0. _1/$, as shown in Fig. 2 (a). In particular, virtual nodes are usually set as child nodes of the target node to ensure that the prior distribution of all existing nodes are not a ected. Second, since rework information is presented as a LR_{Rework} on the distribution of $X_0. _1/$, the CPT of $_0$ should be restricted by $_{Rework}$. In the example, there is a constraint on the CPT of $_0$:

$$_{Rework} = \frac{L.X_0. _1/ = Pass\partial Rework/}{L.X_0. _1/ = Fail\partial Rework/} = \frac{P. _0 = Pass\partial X_0. _1/ = Pass/}{P. _0 = Pass\partial X_0. _1/ = Fail/} = \frac{99}{1}: \quad (4)$$

In particular, as the virtual evidence only specifies a LR, there is no requirement on the specific values of the CPT of $X_{0.\_1}$. Third, because the new processor is known to be working, the virtual node needs to be instantiated with $X_0 = \text{Pass}$ to capture this evidence. When the virtual evidence is added, the posterior distribution of $X_{0.\_1}$ can be updated with the Bayes rule, as shown in Eq. 5.

$$P(X_{0.\_1}|\text{Rework}) = \frac{P(X_{0.\_1}|P_{.\_0} = \text{Pass})P(X_{0.\_1})}{P(P_{.\_0} = \text{Pass})} = \frac{P(X_{0.\_1})L(X_{0.\_1}|X_0 = \text{Pass})}{\sum_{X_{0.\_1}} P(X_{0.\_1})L(X_{0.\_1}|X_0 = \text{Pass})} \quad (5)$$

(a) Rework (b) Repair 1 (c) Repair 2

(d) Redesign 1 (e) Redesign 2

FIGURE 2 BNs for corrected systems

4.2.3 | Impact of repair

According to the definition of repair in Table 1, when a repair activity is executed, an existing system element is modified with parts, processes, or materials that were initially unplanned for that element. We make four assumptions about the impact of repair. First, all initial system elements are retained after repair. This implies that all prior beliefs of the initial elements remain useful for system verification and the network structure remains the same. Second, similar to rework actions, a repair action also provides information that cannot be deduced from the prior beliefs. Third, when repairing a system element, all cause factors of this system element may be changed simultaneously to correct the fault. Fourth, a repair activity may inadvertently degrade the system. With these characteristics, the impact of repair is treated as a belief update of the repaired elements with extra repair information.

Even though all cause factors of an element may be influenced by repair, it is impractical to build a direct relationship between a repair activity and all cause factors. Instead, for simplicity, the belief update is assumed to be applied on the whole conditional distribution of the repaired element. The repair information can be captured in two ways, depending on whether the likelihood of repair success derives from historical records of a repair activity or direct elicitation about the repaired elements. If information about the success of the repair activity is available, the repair information can be captured as virtual evidence. Continuing with the computer BN as an example, consider that a_2 is executed and its results indicate that the computer has not achieved the desired speed. Further inspection indicates that the computer does not achieve the desired speed because part of the shielding is damaged. A repair activity is conducted directly on the computer by fixing the shield with metallic tape, leading to the computer exhibiting the desired speed. This repair activity has been conducted in the past, providing a track record of 80% success.

We capture this by de ning a $LR_{Repair} = L.\ _2 = Pass\partial Repair/ : L.\ _2 = Fail\partial Repair/ = 80 : 20$ and conduct a belief update in the same three steps as that of rework. First, the virtual evidence is modeled as a virtual node that is attached to th BN. So if repair is applied to $_2$, a virtual node $_1$ can be inserted into the BN, as shown in Fig. 2 (b). Second, the CPT of the virtual node is restricted by $_{Repair}$. That is, there is nothing but a LR constraint on the CPT of $_1$

$$_{Repair} = \frac{P.\ _1 = Pass\partial_2 = Pass/}{P.\ _1 = Pass\partial_2 = Fail/} = \frac{80}{20}. \quad (6)$$

Third, the virtual node needs to be instantiated with $_1 = Pass$ to ensure that only the LR, rather than the speci c conditional probabilities of $_1$, are leveraged to update beliefs. When the virtual evidence is added, the con dence of other parameters can be updated with Pearl's method[32]. For example, the updated marginal con dence of $_2$ can be calculated as:

$$P.\ _2\partial Repair/ = \sum_{1} \frac{P.\ _1/P.\ _2\partial_1/L.\ _2\partial_1 = Pass/}{\sum_{2} P.\ _1/P.\ _2\partial_1/L.\ _2\partial_1 = Pass/}. \quad (7)$$

This procedure can be extended to the case in which multiple system elements are repaired in the same time interval. We do so by extending the current use of virtual evidence to cover multiple parameters in a similar way. First, a virtual node is added as a mutual child node of all repaired nodes. For example, if the repair is conducted for $_1$ and $_2$, a virtual node $_2$ can be added in the BN as shown in Fig. 2 (c). Second, the repair information is estimated as a LR for the virtual node: $L.\ _1 = Pass; \ _2 = Pass\partial Repair/ : L.\ _1 = Pass; \ _2 = Fail\partial Repair/ : L.\ _1 = Fail; \ _2 = Pass\partial Repair/ : L.\ _1 = Fail; \ _2 = Fail\partial Repair/ = 45 : 15 : 21 : 17$. Third, the virtual node is instantiated with $_2 = Pass$ After adding this virtual node, Pearl's method[32] can be applied again to update the BN beliefs.

If direct elicitation of the repaired element is employed, the repair information can be captured as not- xed probabilistic evidence. In this case, a belief is characterized as a probability of the system parameter to be correct after repair, which capture the success in repairing the defect and the possibility of inadvertently deteriorating the system, and then this not- xed probabilistic evidence is converted to virtual evidence, after which the same procedure as described for the case in which historical data is leveraged can be applied. Continuing with the computer example, assume that the experts' belief on whether $_2$ is correct after the repair activity is characterized as 0.8, that is, the marginal probability of the overall speed $P.\ _2 = Pass\partial Repair/$ is 0.8 after repair. This not- xed probabilistic evidence can then be converted to virtual evidence with the formula[76]:

$$_{Repair} = \frac{P.\ _2 = Pass\partial Repair/}{P.\ _2 = Pass/} : \frac{P.\ _2 = Fail\partial Repair/}{P.\ _2 = Fail/}; \quad (8)$$

where $P.\ _2 = Pass/$ and $P.\ _2 = Fail/$ are the prior marginal probabilities before repair. Then the belief update can be realized in the same way as was presented for virtual evidence.

4.2.4 | Impact of redesign

According to the de nition of redesign in Table 1, when a redesign activity is executed, a new system is produced according to a new design. We make two assumptions about the impact of redesign. First, since the system itself changes, all redesigned elements may present errors or defects. Second, because redesign may only a ect parts of the system, certain original system elements may be retained. However, the number of possible redesigns is likely extremely large and, as a result, the possible impacts are also extremely large. For example, some redesign activities could simply a ect a minor element in the system, which results in removing one system parameter node in the BN, while other redesign activities may change the complete system e ectively substituting the BN by a completely di erent one. We, therefore, present a general procedure that may need to be tailored to the speci c redesign case. The general procedure consists of four steps:

1. All system parameters that remain after the redesign activity are connected as a new network according to their dependenc relationships.
2. A set of cause factors $\{X_i\}_{new}$ that contains all factors that are changed after redesign is built. As belief elicitation is necessary to characterize the new cause factors, a set of soft evidence is estimated for each cause factor in $\{X_i\}_{new}$.
3. Another set of cause factors $\{X_i\}_{old}$ is derived and reused from the old network. This set includes all cause factors from the old network that are not a ected by the redesign activity. The priors for these cause factors are directly reused from the old network.[3]

[3]We note that this task is not trivial and may be signi cant. It includes assessing that such parameters have indeed not been in uenced by the redesign activity.

4. The CPTs of all network nodes are generated using Eq. 1 and 2 with these two sets.

Two aspects about this general procedure are worth mentioning. First, selecting which cause factors are reused from the old network (i.e., design) and which ones need to be newly modeled depends on both the redesign activity itself and the experience of the engineer. Specifically, if an engineer assesses that their prior belief on a cause factor has changed because of the redesign activity, this cause factor will be chosen as a new cause factor. Second, even in the case where the old distribution is replaced with soft evidence, past information is not ignored. Rather, eliciting the updated CPTs inherently includes past information in the expert's experience.

We describe two examples of this procedure. Consider the computer case presented earlier and assume that an investigation shows that the processor is inherently unable to achieve the required speed. A redesign activity is conducted resulting in a new processor. The speed of the new computer (since it now uses a different processor) is denoted by $\ddot{y}_2$ and the speeds of the new processors by $\ddot{y}_1$. New VAs are defined for each of the attributes of the new system, as shown in Fig. 2 (d). A key aspect is that, because past information is used, we can model a dependency between $\ddot{y}_1$ and y_1, since the knowledge obtained on the original processor with y_1 shapes the confidence of the performance of the new processor.

Now consider the computer case presented earlier. In the new case, however, instead of only designing a new processor, the overall processing concept is redesigned. Specifically, instead of using a single processor, the computer uses two processors that are coordinated by a central unit. The speed of this new computer is denoted by $\ddot{y}_2$, the speeds of each of the new processors by $\ddot{y}_{11}$ and $\ddot{y}_{12}$, respectively, and the ability of the coordination unit to coordinate the two processors by $\ddot{y}_3$. New VAs are defined for each of the attributes of the new system, as shown in Fig. 2 (e). A key aspect is that, because past information is used, we can model a dependency between y_1 and $\ddot{y}_{11}$ and $\ddot{y}_{12}$, since the knowledge obtained on the original processor with y_1 shapes the confidence of the performance of the new processors.

4.3 | Integrating VAs and CAs under a common verification strategy modeling framework

4.3.1 | Two decisions

Our framework captures VAs and CAs as dedicated decisions, informed by each other but independent from each other as well. In one type of decision (which we will refer to as verification decisions, VDs), whether a VA is executed or not is chosen; in the other type of decision (which we will refer to as correction decisions, CDs), whether a CA is executed or not is chosen.[4] The confidence achieved as per the evidence provided by a VA or after the results of a CA inform both types of decisions.

As shown in the basic system verification model, the decision set of a VD contains all VAs that could be executed at a given time (that is, all observable nodes in the BN). Thus, a VD specifies a set of VAs first. If there are no observable nodes selected, the result of the VD is "No VA". The decision set of a CD is based on two components. First, a set of system parameters is selected as the target of the CD. Second, for each system parameter, there are three possible types of CAs, as explained earlier (i.e., rework, repair, and redesign). If no CA is chosen, the result of the CD is "No CA".

In this paper, VDs and CDs are based on two criteria: confidence and past activities. First, for each system element, if the posterior belief of its parameter is sufficiently high, there is no need to conduct VAs or CAs for it. That is, it is considered that, if $P(\theta_i \mid Evidence)$ is larger than some threshold H, the system element i has low uncertainty of errors and no additional VAs or CAs are necessary. The values of current posterior beliefs can be deduced from the BN. Second, past activities may also influence the decision. For simplicity, this study only considers one case that past CAs may influence VDs. That is, once a CA is executed, some previously collected results about the corrected elements may become meaningless because their corresponding system parameters change. So, if necessary, some VAs that had been previously conducted may have to be repeated to ensure all system parameters meet their desired targets.

4.3.2 | Basic interaction loop

We construct the hybrid verification and correction framework by composing several basic interaction loops (BIL). The BIL captures the VD and CD to be made within a given time interval in the system development process. For simplicity and without loss of generality, we make two primary assumptions in this respect. First, only activities within a given time interval are considered to inform VDs and CDs. This implies, for example, that VAs and CAs within a given time interval can be treated

[4]The selection of which VAs and CAs to evaluate or assess derives or is informed by the evidence that VAs can provide, the corrective power of CAs, and the resources necessary to implement them. A decision-making framework for such decision tuple is outside of the scope of this paper.

independently of activities in other time intervals. Second, because CAs aim to eliminate system errors that are identi ed by VAs, there are sequential constraints between each pair of them. In other words, all repair, rework, and redesign activities are only considered when some errors are identi ed by the results of VAs, or, in general, when a desired result or performance is not achieved. Under these conditions, the BIL has been designed as a work ow:

1. At the beginning of each time interval, a designed system is provided with its BN.
2. List all VAs, including No VA , and choose a set of VAs.
3. Conduct the selected VAs, collect their results, and update the BN.
4. List all alternative CAs, including No CA , and choose a CA from them.
5. Execute the selected CA, evaluate its impact, and update the BN.
6. If the uncertainty of error $P(\theta_i)$ is larger than H , go to Step 7. Otherwise, go to Step 2.
7. If there is a next time interval, go to Step 1 and repeat these decisions and activities above. Otherwise, the whole development process is completed.

The basic relation between the di erent elements of the BIL is shown in Fig. 3. In contrast to existing methods, as described in Section 1, VAs and CAs are chosen based on active decisions, which are implemented interactively with real-time results and informed by posterior beliefs of system elements updated by the e ects of previous decisions.

FIGURE 3 Basic interaction loop

5 | CASE DEMONSTRATION

To show how CAs can be modeled and integrated as part of a veri cation strategy modeling and selection framework, this case demonstration focuses on a partial system and corresponding veri cation strategy.

5.1 | Case description

The hybrid veri cation and correction framework is illustrated in a notional satellite communication payload. Consider that the payload is formed by a Signal Generator, an Ampli er, and an Antenna, as shown in Fig. 4 (a). We restrict our attention to the following performance parameters. We use as the primary target the E ective Isotropic Radiated Power (EIRP) of the payload (denoted by θ_8), which we characterize as a function of the Signal Generator output power (denoted by θ_4), the Ampli er gain (denoted by θ_5), and the Antenna gain (denoted by θ_6). Furthermore, we consider the output power of the integrated assembly formed by the Signal Generator and the Ampli er as an intermediate system parameter of potential interest for the veri cation campaign and denote it by θ_7. In addition, we consider prototypes for the Signal Generator, for the Ampli er, and for the Antenna as potentially interesting for the veri cation campaign, and denote their output power and gain by θ_1, θ_2, and θ_3, respectively. Eight VAs are considered for evaluation of VDs, denoted by $\theta_1, \ldots, \theta_8$, where θ_j provides information about θ_i for $i = 1; \ldots; 8$. The resulting veri cation model represented as a BN is shown in Fig. 4 (b).

TABLE 2 Distributions of all cause factors of the payload

	$X_{0.1}$	$X_{0.2}$	$X_{0.3}$	$X_{0.4}$	$X_{1.4}$	$X_{0.5}$	$X_{1.5}$	$X_{0.6}$	$X_{1.6}$	$X_{0.7}$	$X_{1.7}$	$X_{2.7}$	$X_{0.8}$	$X_{1.8}$	$X_{2.8}$	
Pass	0.95	0.95	0.95	0.92	0.15	0.92	0.15	0.92	0.15	0.92	0.25	0.25	0.92	0.25	0.25	
Fail	0.05	0.05	0.05	0.08	0.85	0.08	0.85	0.08	0.85	0.08	0.75	0.75	0.08	0.75	0.75	

	$X_{*.1}$	$X_{+.1}$	$X_{*.2}$	$X_{+.2}$	$X_{*.3}$	$X_{+.3}$	$X_{*.4}$	$X_{+.4}$	$X_{*.5}$	$X_{+.5}$	$X_{*.6}$	$X_{+.6}$	$X_{*.7}$	$X_{+.7}$	$X_{*.8}$	$X_{+.8}$	
Pass	0.80	0.05	0.80	0.05	0.80	0.05	0.75	0.10	0.75	0.10	0.75	0.10	0.70	0.10	0.70	0.10	
Fail	0.20	0.95	0.20	0.95	0.20	0.95	0.25	0.90	0.25	0.90	0.25	0.90	0.30	0.90	0.30	0.90	

For clarity, it should be noted that the state of the performance of the integrated assembly (X_7) depends on the performance of the Signal Generator (X_4), the performance of the Ampli er (X_5), and the cabling connecting them (embedded in X_7). Similarly, the performance of the overall communication payload (X_8) depends on the performance of the integrated assembly (X_7), the performance of the Antenna (X_6), and the cabling between them (embedded in X_8).

(a) The element structure

(b) The Bayes network

FIGURE 4 A satellite communication payload example

We perform a simulation in which we evaluate VDs and CDs for the veri cation of the communication payload. Considering a starting point in which veri cation activities have not been conducted, a set of cause factors is used to construct the CPTs of all nodes. Their distributions are listed in Table 2. The threshold is notionally set as 0.90. All values in this paper have been synthetically generated, albeit not arbitrarily. The values re ect the physical meanings of the relationships between the di erent elements, as well as the relative prediction power of di erent veri cation activities. Use of synthetic data is considered adequate because actual predictions are not relevant to showcase the application of the modeling framework. This is an accepted practice in this type of research[13,77]. This BN captures the prior beliefs of all elements. The prior marginal probabilities of the eight parameters are shown in the third row of Table 3. While the probabilities of all independent factors ($X_{0.i}$; $i = 1, \ldots, 8$) reach the threshold ($H = 0.90$), the marginal probabilities of X_4, X_5, X_6, X_7, X_8 are lower than $H = 0.90$ because of the dependency relationships among parameters. In particular, $X_8 = Pass$ represents the con dence of the payload-level performance required to deploy the communication payload. The BIL model is applied to capture the e ect of veri cation and correction in updating such prior beliefs. Three BILs are used to demonstrate the hybrid veri cation and correction framework proposed in Section 4.

5.2 | BIL 1: VA and redesign activity

The rst BIL starts with a VD. As there is no evidence about the state of the payload yet, the veri cation action starts with the VAs of all prototypes in the BN (i.e., X_1, X_2, X_3). Assume that X_1 fails while the other two pass. Given these three veri cation results, the posterior marginal beliefs of system parameters ($P(X_i = Pass)$) can be updated using Bayesian inference, as shown in the fth row of Table 3. It can be found that X_8 has a con dence value lower than $H = 0.90$. Let us assume that the failure in X_1 implies that it is hard to achieve the desired con dence on X_8 even if other VAs are executed in the future. So no more VAs are executed for the moment.

An investigation is performed to find the causes of the possible errors, and it is determined that the Signal Generator is unable to provide the desired output voltage due to its operating voltage, which is insufficient. Let us assume that a correction activity is worth doing (given rework costs, expected confidence increase, etc.) and it is decided to redesign the prototype of the Signal Generator to operate with a higher voltage. This correction leads to a change in the structure of the original BN, following the modeling paradigm described in Section 4.2.4. We explain this change in three steps, which are depicted in Fig. 5. First, we get a new Signal Generator with a new prototype. So, we characterize them as two new nodes $\ddot{X}_4$ and $\ddot{X}_1$, respectively. As a result, all higher level assemblies (in this case, the Integrated Assembly and the whole Communication Payload) also become new versions of the system; note that the original Communication Payload used a Signal Generator that will no longer be used, thus effectively, a different Communication Payload is being developed after the correction. The resulting Integrated Assembly and Communication Payload are characterized as new nodes $\ddot{X}_7$ and $\ddot{X}_8$, respectively. These changes are shown in Fig. 5 (a). Second, system elements that are not affected by this change remain unchanged with respect to the original BN. They are used as cause factors of the new nodes defined earlier. In this example, particularly, neither the Antenna nor the Amplifier are affected by the change in the Signal Generator. Yet, the performance of the new Integrated Assembly still depends on the performance of the Amplifier and the performance of the Communication Payload still depends on the performance of the Antenna. Therefore, X_5 and X_6 in the original BN are connected to $\ddot{X}_7$ and $\ddot{X}_8$, respectively. Furthermore, since the learning that occurs in past activities is used in future activities, the confidence on the state of the new Signal Generator is a function of the results obtained from the prototype of the original Signal Generator. This use of past information is captured by the connection from X_1 to $\ddot{X}_1$. These changes are shown in Fig. 5 (b). Third, because the original system is no longer used as a solution (note that the development target is $\ddot{X}_8$ instead of X_8), nodes that represent such solutions can be removed from the BN to simplify the model with no impact on its fidelity. Specifically, X_4, X_7, X_8 and their corresponding VAs (X_4, X_7, X_8) are removed from the network leading to the one shown in Fig. 5 (c).

In this paper, we only re-estimate the two cause factors of $\ddot{X}_1$ after the redesign (i.e., the independent factor X_0 for $\ddot{X}_1$ and the dependent factor X_1. $\ddot{X}_1$ between X_1 and $\ddot{X}_1$). Other affected elements reuse the original cause factors. Two corresponding pieces of soft evidence are estimated for the two cause factors. That is, $P(X_0.\ddot{X}_1 = \text{Fail}) = 0.05$ and $P(X_1.\ddot{X}_1 = \text{Fail}) = 0.40$. With this soft evidence, the posterior marginal probabilities can be calculated using the Bayesian inference, as shown in the seventh row of Table 3. In particular, $P(\ddot{X}_1 = \text{Pass}|\text{Redesign})$ is 0.991, which suggests that the confidence of the prototype of the Signal Generator is larger than the original one (i.e., $X_1 = \text{Pass}) = 0.800$).

(a) Updated nodes (b) Connection between old nodes and updated nodes (c) New network structure

FIGURE 5 Construction of the new BN for the redesign activity

To observe how the impact of redesign stands up under uncertainties in the evidence, we explored the relationship between the soft evidence $(X_0.\ddot{X}_1, X_1.\ddot{X}_1)$ and the posterior confidences of two target nodes $(\ddot{X}_1, \ddot{X}_8)$. The legend of the surface shows the trend of the posterior confidences given different values of the two causal factors. First, the surface between $P(X_0.\ddot{X}_1 = \text{Fail})$, $P(X_1.\ddot{X}_1 = \text{Fail})$ and $P(\ddot{X}_1 = \text{Pass}|\text{Redesign})$ is shown in Fig. 6 (a). It is found that the maximum of $P(\ddot{X}_1 = \text{Pass}|\text{Redesign})$ is 0.990 when all values of $X_0.\ddot{X}_1 = \text{Fail}$ and $X_1.\ddot{X}_1 = \text{Fail}$ are set at 0.01 and the minimum is 0.010 when all these values are set at 0.99. Second, the surface between $P(X_0.\ddot{X}_1 = \text{Fail})$, $P(X_0.\ddot{X}_1 = \text{Fail})$ and $P(\ddot{X}_8 = \text{Pass}|\text{Redesign})$ is shown in Fig. 6 (b). It is found that the two surfaces reach the maximum/minimum with the same values of $P(X_0.\ddot{X}_1)$ and $P(X_1.\ddot{X}_1)$.

TABLE 3 Evidence and confidence results of 3 BILs

		Evidence	Confidence							
Prior			θ_1	-	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8
			0.950	-	0.950	0.950	0.881	0.881	0.881	0.763	0.689
BIL 1			θ_1	-	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8
	VA	θ_1 = Fail; θ_2 = Pass; θ_3 = Pass:	0.800	-	0.997	0.997	0.764	0.917	0.917	0.710	0.675
			θ_1	$\ddot{\theta}_1$	θ_2	θ_3	$\ddot{\theta}_4$	θ_5	θ_6	$\ddot{\theta}_7$	$\ddot{\theta}_8$
	CA	θ_1 = Fail; θ_2 = Pass; θ_3 = Pass; $P(X_0 \mid \ddot{\theta}_1$ = Fail$) = 0.05$; $P(X_1 \mid \ddot{\theta}_1$ = Fail$) = 0.40$:	0.865	0.991	0.997	0.950	0.913	0.917	0.881	0.807	0.716
BIL 2			θ_1	$\ddot{\theta}_1$	θ_2	θ_3	$\ddot{\theta}_4$	θ_5	θ_6	$\ddot{\theta}_7$	$\ddot{\theta}_8$
	VA	θ_1 = Fail; θ_2 = Pass; θ_3 = Pass; $P(X_0 \mid \ddot{\theta}_1$ = Fail$) = 0.05$; $P(X_1 \mid \ddot{\theta}_1$ = Fail$) = 0.40$; $\ddot{\theta}_4$ = Pass; θ_5 = Fail; θ_6 = Pass:	0.868	0.998	0.991	0.986	0.988	0.755	0.982	0.744	0.734
	CA	θ_1 = Fail; θ_2 = Pass; θ_3 = Pass; $P(X_0 \mid \ddot{\theta}_1$ = Fail$) = 0.05$; $P(X_1 \mid \ddot{\theta}_1$ = Fail$) = 0.40$; $\ddot{\theta}_4$ = Pass; θ_6 = Pass; $L(X_0 \mid \theta_5$ = Pass$, \text{Rework}) / L(X_0 \mid \theta_5$ = Fail$, \text{Rework}) = 99 : 1$:	0.849	0.962	0.997	0.999	0.972	0.996	0.988	0.898	0.842
BIL 3			θ_1	$\ddot{\theta}_1$	θ_2	θ_3	$\ddot{\theta}_4$	θ_5	θ_6	$\ddot{\theta}_7$	$\ddot{\theta}_8$
	VA	θ_1 = Fail; θ_2 = Pass; θ_3 = Pass; $P(X_0 \mid \ddot{\theta}_1$ = Fail$) = 0.05$; $P(X_1 \mid \ddot{\theta}_1$ = Fail$) = 0.40$; $\ddot{\theta}_4$ = Pass; θ_6 = Pass; $L(X_0 \mid \theta_5$ = Pass$, \text{Rework}) / L(X_0 \mid \theta_5$ = Fail$, \text{Rework}) = 99 : 1$: $\ddot{\theta}_7$ = Fail:	0.839	0.944	0.994	0.999	0.941	0.992	0.988	0.746	0.738
	CA	θ_1 = Fail; θ_2 = Pass; θ_3 = Pass; $P(X_0 \mid \ddot{\theta}_1$ = Fail$) = 0.05$; $P(X_1 \mid \ddot{\theta}_1$ = Fail$) = 0.40$; $\ddot{\theta}_4$ = Pass; θ_6 = Pass; $L(X_0 \mid \theta_5$ = Pass$, \text{Rework}) / L(X_0 \mid \theta_5$ = Fail$, \text{Rework}) = 99 : 1$: $L(\ddot{\theta}_7$ = Pass$, \text{Repair}) / L(\ddot{\theta}_7$ = Fail$, \text{Repair}) = 19 : 1$:	0.855	0.974	0.999	0.999	0.992	0.999	0.988	0.994	0.908

However, the range of $P(\ddot{\theta}_8$ = Pass$, \text{Redesign})$ is much smaller than that of $P(\ddot{\theta}_1$ = Pass$, \text{Redesign})$. This is attributed to the distance between the redesigned node and the target nodes.

(a) Impact on $P(\ddot{\theta}_1$ = Pass$, \text{Redesign})$ (b) Impact on $P(\ddot{\theta}_8$ = Pass$, \text{Redesign})$

FIGURE 6 Sensitivity analysis of redesign

5.3 | BIL 2: VA and rework activity

Since $P(\ddot{\theta}_8$ = Pass$, \text{Redesign}) = 0.716$ is still smaller than $H = 0.90$ after the first BIL, three additional VAs (i.e., $\ddot{\theta}_4$, θ_5, θ_6) are executed at higher integration levels. Let us assume that θ_5 fails while the other two pass. Given the verification results, the posterior marginal beliefs of the system parameters can be updated with the Bayesian inference and their belief values ar

shown in the ninth row of Table 3. Let us assume that the failure evidence e_5 implies that it is difficult to achieve the desired confidence on θ_8 even if all remaining VAs are executed. Thus, no more VAs are executed for the moment.

An investigation is performed, and it is determined that a_5 is unable to provide the desired gain due to a workmanship problem. Let us assume that a correction activity is worth doing, under similar conditions to the previous case, and it is decided to rework the faulty part of the Amplifier. Another Amplifier is built to conform to the design (i.e., until the test results of the reworked Amplifier are pass). Since no design difference exists between the new Amplifier and the original one, the structure of the BN does not change. Instead, a notional $LR_{Rework} = L(X_{0.5} = Pass|Rework) : L(X_{0.5} = Fail|Rework) = 99 : 1$ is estimated for the rework activity and it is used to update the independent factor $X_{0.5}$ for the reworked Amplifier. The evidence $e_5 = Fail$ becomes uninformative because the rework activity has removed the faulty part of the Amplifier; only the evidence $e_4 = Pass$ and $e_6 = Pass$ are applicable in this case. With this LR, the updated independent factor $X_{0.5}$ becomes $P(X_{0.5} = Fail) = 0.08 \times 1.092 \times 99 + 0.08 \times 1) = 0.0009$. With this updated independent factor, the posterior marginal probability $P(\theta_8|Rework)$ increases to 0.842 using the Bayesian inference, as shown in the tenth row of Table 3.

For the reworked payload, the relationship between LR_{Rework} and $P(\theta_8 = Pass|Rework)$ is plotted in Fig. 8 (a). The list of LRs is $\{2^i\}; i = \{10, 9, \ldots, -10\}$. Three aspects are worth mentioning. First, $P(\theta_8 = Pass|Rework)$ is a monotone function of LR_{Rework} and it is monotone-increasing in this case. Especially, when LR_{Rework} is larger than 1, the marginal probability is larger than 0.806, which is the marginal probability when the rework has no impact (i.e., only the evidence $e_5 = Fail$ becomes uninformative after rework). Second, the upper bound 0.843 approximates the case $e_5 = Pass$ and the lower bound is 0.380 for the case $e_5 = Fail$. Third, $P(\theta_8 = Pass|Rework)$ increases sharply within the interval $[2^{-8}, 2^1]$ and becomes stable when LR_{Rework} is outside of it.

5.4 | BIL 3: VA and repair activity

Since $P(\theta_8 = Pass|Rework) = 0.842$ is still lower than the desired confidence of $H = 0.90$, let us assume that it is decided to conduct a_7 and let us assume that it fails. The posterior marginal beliefs of system parameters can be obtained by the Bayesian inference, which are shown at the twelfth row of Table 3. Let us assume that it will be very difficult to achieve the desired confidence on θ_8 after all remaining VAs are executed in the future. So no additional VAs are conducted for the moment.

After an investigation, it is determined that a_7 is unable to provide the desired output power because of a cable defect between the Signal Generator, the Amplifier, and the cabling between them. So, under similar assumptions as for the previous cases, it is decided to conduct a CA and, instead of producing a new cable, cut and connect the defective wires in the cable. Let us assume that the repair activity is successful and that the repaired Integrated Assembly is reported to operate correctly. Let us also assume that historical records indicate that 95% of this type of repair yields error-free Integrated Assemblies. In this case, the BN evolves as shown in Fig. 7. Using virtual evidence, a $LR_{Repair} = L(\theta_7 = Pass|Repair) : L(\theta_7 = Fail|Repair) = 19 : 1$ is added to the parameter θ_7, shown as a virtual node e_7 in the BN. It is also noticeable that the evidence $e_7 = Fail$ becomes uninformative in this case because the repair activity has changed the Integrated Assembly and the virtual evidence embeds such past information. Therefore, the posterior marginal probability $P(\theta_8 = Pass|Repair)$ becomes 0.908, which shows the positive impact of the repair activity on the BN. Because $P(\theta_8|Repair)$ is larger than $H = 0.90$ now, it is unnecessary to conduct any more VAs or CAs. The development is considered completed.

For completeness, in the case where the direct estimation about the repaired element would have been used, instead of historical records to evaluate the performance of the repair activity, the impact of the repair would have been estimated as the posterior marginal distribution $P(\theta_7|Repair)$. For example, assuming an estimation of 99.4% confidence on the repaired Integrated Assembly correctly operating, the posterior marginal distribution $P(\theta_7 = Pass|Repair)$ would be 0.994. This evidence could be converted to a virtual evidence $LR_{Repair} = \frac{P(\theta_7 = Pass|Repair)}{P(\theta_7 = Pass)} : \frac{P(\theta_7 = Fail|Repair)}{P(\theta_7 = Fail)} = \frac{0.994}{0.898} : \frac{0.006}{0.102} \approx 19 : 1$, where $P(\theta_7 = Pass) = 0.898$ and $P(\theta_7 = Fail) = 0.102$ are the marginal probabilities of $P(\theta_7)$ before the repair activity. Then the Bayesian inference could be used in the same way as the virtual evidence method, as described in the previous paragraph.

To investigate the sensitivity of the impact of repair on $P(\theta_8 = Pass|Repair)$, LR_{Repair} is assigned with 21 values of LRs $\{2^i\}; i = \{10, 9, \ldots, -10\}$ to update the BN. The 21 results are plotted as a blue line in Fig. 8 (b). The curve has the same three characteristics as that of rework in BIL 2. First, $P(\theta_8 = Pass|Repair)$ is a monotone function of LR_{Repair}. When LR_{Repair} is larger than 1, the marginal probability is larger than 0.842, which is the marginal probability when the repair has no impact. Second, the upper bound is 0.912, which approximates the case $e_7 = Pass$ and the lower bound is 0.230 for the case $e_7 = Fail$. Third, $P(\theta_8 = Pass|Repair)$ increases sharply within the interval $[2^{-8}, 2^1]$ and becomes stable when LR_{Repair} is outside of it.

FIGURE 7 Adding a virtual node $_7$ to the BN

(a) Impact of $_{\text{Rework}}$ (b) Impact of $_{\text{Repair}}$

FIGURE 8 Sensitivity analysis of repair and rework

6 | CONCLUSION

We have presented modeling constructs to capture the e ects of correction activities when modeling veri cation strategies using BN. These constructs allow for capturing correction activities as the result of dedicated decisions instead of properties that are inherent to veri cation activities. In this way, Bayesian inference can be used as a general process to quantitatively assess the joint e ect of veri cation and correction activities.

A key tenet of the presented modeling constructs is the recognition that the system being veri ed changes when a correction activity occurs. Consequently, the belief structure that stems from a given veri cation strategy on a system design must necessarily change to re ect the new system. Speci c modeling constructs have been presented for redesign, rework, and repair. For redesign, new nodes are incorporated into the BN: those representing the redesigned elements and those at higher levels of integration that embed such redesigned elements. The elements in the original BN that had to be redesigned then act as cause factors to the redesigned elements, since past information is used to inform the con dence of the redesigned elements. Higher level assemblies in the original network are omitted, since they represent planned development paths that have been discarded. Elements in the original network that are not causally related to the redesign system remain. For rework and repair, it su ces to add statistical features to the BN in the form of uncertain evidence or LR to capture the e ects of the corrections, e ectively leading to a new BN.

In all cases, beliefs must usually be re-elicited for all system elements. However, the presented constructs can be e ective in limiting this e ort. First, the application of uncertain evidence narrows the elicitation scope from joint distribution to likelihood information. This allows the reuse of several of the original beliefs. Second, using the PCI model allows us to decompose the CPTs into cause factors, allowing the reuse of all old cause factors that are a ected by CAs in the new network. While the size of a CPT exponentially increases with the number of parent nodes, the number of cause factors is linearly proportional to that

of the parent nodes. Third, the independence of the cause factors in the PCI models allows us to change the BN structure mor easily than usual, given the joint distributions in the BN.

Overall, we suggest that these constructs should improve the delity of the models used to design veri cation strategies, resulting inmore valuableveri cation strategies. By adopting the presented modeling constructs, we anticipate that a systems engineer will be able to (1) select a more valuable set of veri cation activities and veri cation paths, (2) anticipate valuable points of potential correction, and (3) reduce the amount of reactive work during veri cation. These potential improvements are enabled by the mathematical nature of modeling constructs, to which systems engineers do not have access to with existin methods.

However, it is important to note that the work presented in this paper is fundamental research and should be understood as building step towards a fully implementable paradigm. Therefore, future research is suggested in this direction. First, the CPTs of a BN are decomposed in this paper with basic PCI models (i.e., generalized noisy OR and noisy AND). Other models may need to be explored to cope with cause factors of higher complexity in practice. Second, the proposed constructs need to b tested in decision and/or optimization methods that consider past information, current beliefs, and future alternative actions, as well as other development related attributes, such as costs. Third, scalability aspects related to the construction, managemen and characterization of large BN networks representing real systems need to be explored. Fourth, we recognize that the propose constructs and modeling approach will likely require higher nancial and temporal e orts than existing methods, particularly as the BN becomes larger and the frequency of correction increases. These aspects need to be studied to assess the conditi for adopting the presented constructs in practice.

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant N. CMMI-1762883.

References

1. Engel Avner.Veri cation, Validation, and Testing of Engineered Systems.John Wiley & Sons; 2010.
2. Salado Alejandro, Kannan Hanumanthrao. A mathematical model of veri cation strategies.Systems Engineering. 2018;21(6):593 608.
3. Standardization International Organization.Quality Management Systems-Fundamentals and Vocabulary (ISO 9000: 2015). ISO Copyright o ce; 2015.
4. Romli Fairuz I, Daud Nur Sha qah, Habibillah Nadia Hanani Ahmad. Robust baseline design selection methodology: aircraft redesign case study. In: :012015IOP Publishing; 2018.
5. Fricke Ernst, Gebhard Bernd, Negele Herbert, Igenbergs Eduard. Coping with changes: causes, ndings, and strategies. Systems Engineering2000;3(4):169 179.
6. Thomke Stefan H.Experimentation Matters: Unlocking the Potential of New Technologies for Innovation.Harvard Business Press; 2003.
7. Forsberg Kevin, Mooz Hal, Cotterman Howard. Visualizing project management: models and frameworks for mastering complex systems:245 247. John Wiley & Sons third ed.2005.
8. Dullen Shawn, Verma Dinesh, Blackburn Mark. Review of Research into the Nature of Engineering and Development Rework: Need for a Systems Engineering Framework for Enabling Rapid Prototyping and Rapid Fielding.Procedia Computer Science2019;153:118 125.
9. Osborne Sean M. Product development cycle time characterization through modeling of process iteration. PhD thesisMassachusetts Institute of Technology1993.
10. Kennedy Brian M, Sobek Durward K, Kennedy Michael N. Reducing rework by applying set-based practices early in the systems engineering process.Systems Engineering2014;17(3):278 296.

11. Reichelt Kimberly, Lyneis James. The dynamics of project performance: benchmarking the drivers of cost and schedule overrun. European management journal. 1999;17(2):135 150.
12. Barad M, Engel A. Optimizing VVT strategies: a decomposition approach. Journal of the Operational Research Society. 2006;57(8):965 974.
13. Kulkarni Aditya U, Salado Alejandro, Xu Peng, Wernz Christian. An evaluation of the optimality of frequent veri cation for vertically integrated systems. Systems Engineering. 2021;24(1):17 33.
14. Kulkarni Aditya U, Wernz Christian, Salado Alejandro. Coordination of veri cation activities with incentives: a two- rm model. Research in Engineering Design. 2021;32(1):31 47.
15. Xu Peng, Salado Alejandro. A Concept for Set-based Design of Veri cation Strategies. In: :356 370. Wiley Online Library; 2019.
16. Xu Peng, Salado Alejandro, Deng Xinwei. A Parallel Tempering Approach for E cient Exploration of the Veri cation Tradespace in Engineered Systems. arXiv preprint arXiv:2109.11704. 2021;.
17. Zhang Xilin, Tan Yuejin, Yang Zhiwei. Rework Quanti cation and In uence of Rework on Duration and Cost of Equipment Development Task. Sustainability. 2018;10(10):3590.
18. Salado Alejandro, Kannan Hanumanthrao. Elemental patterns of veri cation strategies. Systems Engineering. 2019;22(5):370 388.
19. Thunnissen Daniel P. Uncertainty classi cation for the design and development of complex systems. In: :1 16. Newport Beach CA; 2003.
20. Stockstrom Christoph, Herstatt Cornelius. Planning and uncertainty in new product development. R&D Management. 2008;38(5):480 490.
21. Gralla Erica, Szajnfarber Zoe. Characterizing representational uncertainty in system design and operations. Systems Engineering. 2016;19(6):535 548.
22. Sentz Kari, Ferson Scott, others. Combination of Evidence in Dempster-Shafer theory. Sandia National Laboratories Albuquerque; 2002.
23. Shafer Glenn. A Mathematical Theory of Evidence. Princeton university press; 1976.
24. Zadeh Lot A. Fuzzy Sets. Information and Control. 1965;8:338 353.
25. Dubois Didier, Prade Henri. Possibility theory: qualitative and quantitative aspects. In: Springer 1998 (pp. 169 226).
26. Dubois Didier. Possibility theory and statistical reasoning. Computational statistics & data analysis. 2006;51(1):47 69.
27. Destercke Sébastien, Dubois Didier, Chojnacki Eric. Unifying practical uncertainty representations I: Generalized p-boxes. International Journal of Approximate Reasoning. 2008;49(3):649 663.
28. Hester Patrick. Epistemic uncertainty analysis: an approach using expert judgment and evidential credibility. Journal of Quality and Reliability Engineering. 2012;2012.
29. Martinez Felipe Aguirre, Sallak Mohamed, Schön Walter. An e cient method for reliability analysis of systems under epistemic uncertainty using belief function theory. IEEE Transactions on Reliability. 2015;64(3):893 909.
30. Cox Lisa, Delugach Harry S, Skipper David. Dependency analysis using conceptual graphs. In: :117 130; 2001.
31. Sharma Arun, Grover PS, Kumar Rajesh. Dependency analysis for component-based software systems. ACM SIGSOFT Software Engineering Notes. 2009;34(4):1 6.
32. Pearl Judea. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann; 1988.

33. Smets Philippe, Kennes Robert. The transferable belief model. Artificial intelligence. 1994;66(2):191 234.

34. Sallak Mohamed, Schön Walter, Aguirre Felipe. Transferable belief model for reliability analysis of systems with data uncertainties and failure dependencies. Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability. 2010;224(4):266 278.

35. Stylios Chrysostomos D, Groumpos Petros P. Modeling complex systems using fuzzy cognitive maps. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans. 2004;34(1):155 162.

36. Yang Jian-Bo, Liu Jun, Wang Jin, Sii How-Sing, Wang Hong-Wei. Belief rule-base inference methodology using the evidential reasoning approach-RIMER. IEEE Transactions on systems, Man, and Cybernetics-part A: Systems and Humans. 2006;36(2):266 285.

37. Grover Je . A literature review of Bayes' theorem and Bayesian belief networks (BBN). In: Springer 2013 (pp. 11 27).

38. Cai Baoping, Huang Lei, Xie Min. Bayesian networks in fault diagnosis. IEEE Transactions on Industrial Informatics. 2017;13(5):2227 2240.

39. Neapolitan Richard E, others. Learning Bayesian Networks. Pearson Prentice Hall Upper Saddle River, NJ; 2004.

40. Dagum Paul, Galper Adam, Horvitz Eric. Dynamic network models for forecasting. In: :41 48. Elsevier; 1992.

41. Dagum Paul, Galper Adam, Horvitz Eric, Seiver Adam. Uncertain reasoning and forecasting. International Journal of Forecasting. 1995;11(1):73 87.

42. Koller Daphne, Pfe er Avi. Object-oriented Bayesian networks. arXiv preprint arXiv:1302.1554. 2013;.

43. Ivanovska Magdalena, Jøsang Audun, Kaplan Lance, Sambo Francesco. Subjective networks: Perspectives and challenges. In: :107 124. Springer; 2015.

44. Xu Peng, Cho Jin-Hee, Salado Alejandro. Expert Opinion Fusion Framework Using Subjective Logic for Fault Diagnosis. IEEE Transactions on Cybernetics. 2020;.

45. Box George EP, Tiao George C. Bayesian Inference in Statistical Analysis. WISCONSIN UNIV MADISON DEPT OF STATISTICS; 1973.

46. Mrad Ali Ben, Delcroix Véronique, Piechowiak Sylvain, Leicester Philip, Abid Mohamed. An explication of uncertain evidence in Bayesian networks: likelihood evidence and probabilistic evidence. Applied Intelligence. 2015;43(4):802 824.

47. Valtorta Marco, Kim Young-Gyun, Vomlel Ji°í. Soft evidential update for probabilistic multiagent systems. International Journal of Approximate Reasoning. 2002;29(1):71 106.

48. Mrad Ali Ben, Delcroix Veronique, Piechowiak Sylvain, Maalej Mohamed Amine, Abid Mohamed. Understanding soft evidence as probabilistic evidence: Illustration with several use cases. In: :1 6. IEEE; 2013.

49. Gowadia Vaibhav, Farkasl C, Valtorta M. Agent based intrusion detection with soft evidence. In: :345 350; 2003.

50. Subramanya Amarnag, Bilmes Je . Virtual evidence for training speech recognizers using partially labeled data. In: :165 168; 2007.

51. Corradi Fabio, Pinchi Vilma, Barsanti Iljà, Garatti Stefano. Probabilistic classi cation of age by third molar development: the use of soft evidence. Journal of Forensic Sciences. 2013;58(1):51 59.

52. Costa Ramon, Sobek Durward K. Iteration in engineering design: inherent and unavoidable or product of choices made? In: :669 674; 2003.

53. Mitchell Victoria L, Nault Barrie R. Cooperative planning, uncertainty, and managerial control in concurrent design. Management Science. 2007;53(3):375 389.

54. Taylor Tim, Ford David N. Tipping point failure and robustness in single development projects. *System Dynamics Review: The Journal of the System Dynamics Society.* 2006;22(1):51–71.

55. Love Peter ED, Ackermann Fran, Teo Pauline, Morrison John. From individual to collective learning: A conceptual learning framework for enacting rework prevention. *Journal of construction engineering and management.* 2015;141(11):05015009.

56. Li Heng, Love Peter ED. Visualization of building interior design to reduce rework. In: :187–191IEEE; 1998.

57. Krishnan Viswanathan, Eppinger Steven D, Whitney Daniel E. A model-based framework to overlap product development activities. *Management science.* 1997;43(4):437–451.

58. Roemer Thomas A, Ahmadi Reza. Concurrent crashing and overlapping in product development. *Operations research.* 2004;52(4):606–622.

59. Thomke Stefan, Bell David E. Sequential testing in product development. *Management Science.* 2001;47(2):308–323.

60. Tahera Khadija, Earl Chris, Eckert Claudia. A method for improving overlapping of testing and design. *IEEE Transactions on Engineering Management.* 2017;64(2):179–192.

61. Yassine Ali, Joglekar Nitin, Braha Dan, Eppinger Steven, Whitney Daniel. Information hiding in product development: the design churn effect. *Research in Engineering Design.* 2003;14(3):145–161.

62. Love Peter ED, Mandal Purnendu, Li Hui. Determining the causal structure of rework influences in construction. *Construction Management & Economics.* 1999;17(4):505–517.

63. Denardo Eric V, Lee Thomas YS. Managing uncertainty in a serial production line. *Operations Research.* 1996;44(2):382–392.

64. Browning Tyson R, Eppinger Steven D. Modeling impacts of process architecture on cost and schedule risk in product development. *IEEE transactions on engineering management.* 2002;49(4):428–442.

65. Liao Gwo-Liang. Production and maintenance policies for an EPQ model with perfect repair, rework, free-repair warranty, and preventive maintenance. *IEEE Transactions on Systems, Man, and Cybernetics: Systems.* 2015;46(8):1129–1139.

66. Taleizadeh AA, Jalali-Naini S Gh, Wee H-M, Kuo T-C. An imperfect multi-product production system with rework. *Scientia Iranica.* 2013;20(3):811–823.

67. Salado Alejandro, Kannan Hanumanthrao. Properties of the Utility of Verification. In: :1–8IEEE; 2018.

68. Berger James O. *Statistical Decision Theory and Bayesian Analysis*. Springer; 1985.

69. Bolt Janneke H, Gaag Linda C. An empirical study of the use of the noisy-OR model in a real-life Bayesian network. In: :11–20Springer; 2010.

70. Oniśko Agnieszka, Druzdzel Marek J, Wasyluk Hanna. Learning Bayesian network parameters from small data sets: Application of Noisy-OR gates. *International Journal of Approximate Reasoning.* 2001;27(2):165–182.

71. Woudenberg Steven P.D., Van Der Gaag Linda C.. Using the Noisy-OR model can be harmful… but it often is not. In: :122–133Springer; 2011.

72. Henrion Max. Practical issues in constructing a Bayes' belief network. *arXiv preprint arXiv:1304.2725.* 2013;.

73. Diez Francisco Javier. Parameter adjustment in Bayes networks. The generalized noisy OR–gate. In: :99–105Elsevier; 1993.

74. Koller Daphne, Friedman Nir. *Probabilistic Graphical Models: Principles and Techniques*. MIT press; 2009.

75. Abbas Ali E, Howard Ronald A. *Foundations of Decision Analysis*. Pearson Higher Ed; 2015.

76. Chan Hei, Darwiche Adnan. On the revision of probabilistic beliefs using uncertain evidence. *Artificial Intelligence.* 2005;163(1):67–90.

77. Engel Avner, Barad Miryam. A methodology for modeling VVT risks and costs. *Systems engineering.* 2003;6(3):135–151.

78. Kulkarni Aditya Umesh. Incentive Mechanism Design for Systems with Many Agents: A Multiscale Decision Theory Approach. PhD thesisVirginia Tech2018.