Automation of the SHIELD Methodology for System Hazard Analysis and Resilient Design

Anthony Marcus and Ionut Cardei
Department of Computer & Electrical Engineering and Computer Science
Florida Atlantic University
Boca Raton, Florida, United States of America
{ amarcu10, icardei}@fau.edu

Gabriel Alsenas
Southeast National Marine Renewable Energy Center
Florida Atlantic University
Boca Raton, Florida, United States of America
galsenas@fau.edu

Abstract—The System Hazard Indication and Extraction Learning Diagnosis (SHIELD) methodology was developed as a novel method to perform system hazard analysis and resilient design. In an earlier paper we described SHIELD conceptually and outlined the details necessary to conduct the analysis manually. This approach integrates state space examination into the analysis process in order to facilitate efficient and comprehensive identification of undiscovered risks and hazard scenarios. SHIELD requires that three phases be performed serially to achieve a system hazard evaluation: decomposition, evaluation and prescription. The first phase of SHIELD, decomposition, breaks the system down hierarchically and recursively into smaller components so that the state space associated with each component is more manageable for the user. In the evaluation phase experts analyze the associated state space and transitions for each component, recursively, bottom-up. The prescription phase applies a set of heuristics to the results from the preceding phase to reduce system hazard. The main contribution of this paper is the automation of the methodology to reduce the effort used for analysis without sacrificing accuracy or overlooking hazardous state combinations. We describe in detail our automation concept and preliminary tests with the prototype.

Keywords— risk analysis, hazard analysis, system resilience engineering; Bayesian Trees

I. Introduction

As one of the research teams of scientists around the world searching for cleaner, more efficient means of harvesting renewable energy, the Southeast National Marine Renewable Energy Center (SNMREC) at FAU is developing an ocean current turbine testbed (OCTT)[1]. The OCTT will be located in the Gulfstream current off the Florida coast where it will be available to test small-scale underwater demonstration turbines which harvest energy from ocean currents. An ocean-based test platform and a shore-based remote data storage, collection, and analysis site are located approximately 25 kilometers apart and will transfer data via a wireless communication link. A Prognosis and Health Monitoring (PHM) system [1] was created to assess the health and performance of a turbine under test. The challenges of implementing such a system inspired an effort to understand how we ensure that the OCTT will operate autonomously and at optimum efficiency, even when facing disruptive or traumatic events.

The high costs associated with any system repair and failure dictate that, prior to the development and implementation of any complex system proposal, experts should analyze the design for possible weaknesses and fault-prone elements such that, when found, precautionary measures may be implemented to boost overall system resilience and reliability. Many methodologies exist which may be applied to accomplish this task, all of which implement a variety of novel concepts to encourage scenarios to emerge and to be discovered by members of a design team.

In our preliminary study of the OCTT system [2], we applied and individually tested three modern methodologies for system hazard and risk analysis and resilience engineering: STPA (STAMP Based Process Analysis) [3, 4, 5], Hazard and Operability Analysis (HAZOP) [6, 7, 8, 9, 10, 11], and a Heuristic-based approach [12, 13]. We applied them individually to analyze the PHM architecture in an attempt to increase overall system resilience. Once each examination was complete, all were thoroughly evaluated for missed scenarios and risks that were not found by the results [2]. To extract these missing scenarios, we formulated an innovative methodology capable of thoroughly revealing vulnerabilities a system may have. In this article, we describe our work automating the SHIELD methodology and its hazard analysis using Bayesian Networks. We introduce a hazard metric called an Impact Factor that is used to rank risky component state combinations. Our system can also be used to perform a complex "what if" analysis for scenarios where some system components and environmental parameters are "observed" in hazardous states and then compute probability distributions of hazard states for the top-level system. The AutoSHIELD diagnostic algorithm can presume the system (and other components or environment variables) are in hazardous states and then generate probability information and predictions of effects on other system components.

This research is sponsored by the Southeast National Marine Renewable Energy Center (SNMREC).

978-1-4673-3108-1/13/$31.00 ©2013 IEEE

In section II, we give a brief overview of our methodology. Section III describes our automation concept and prototype description. The experimental results from our preliminary tests are discussed in section IV. In section V we discuss some related work and finally, section VI concludes the paper with a discussion of the results and future work.

II. SHIELD Overview

We designed the System Hazard Indication and Extraction Learning Diagnosis (SHIELD) methodology [2] for complex system hazard analysis and resilient design. SHIELD integrates a complete state space examination into the analysis process in order to facilitate the location of undiscovered risks and hazard scenarios. First, we briefly describe SHIELD and outline the general topics necessary to conduct the analysis manually.

SHIELD requires three phases to complete a system hazard evaluation: decomposition, evaluation and prescription. To understand each phase, we first briefly review our definition of state and how SHIELD observes interactions between system components. We have defined a system's state space as the relevant aspects of a component which include resources, services, capabilities, and features which may be combined. As most systems are comprised of interacting components, possible actions between them are described by process cycles that define the sequence of events and component states involved in the execution of any process, including: inputs, outputs, and feedback loops between the components involved in a particular cycle.

The first phase of SHIELD, system decomposition, divides the system hierarchically and recursively into smaller components (and environmental factors) so that the state space associated with each component is more detailed. This phase is conducted until each sub-system is easy to describe and understand like, for instance, individual or stand-alone components. In the evaluation phase, experts analyze the state space associated with each subsystem and the transitions between states recursively and bottom-up. This phase is plagued with two challenges, the time that would be needed for the analysis of highly complex systems and the possibility of human error when observing and analyzing elaborate state spaces. We address each of these challenges in Section III where show how we automated the methodology and integrated measures to minimize the drawbacks associated with each when applying SHIELD.

In the final phase of the analysis, prescription, system designers apply a set of heuristics [12] to the results from the preceding phase. The heuristics are designed to inspire creative ideas from the design team to discover solutions to mitigate hazardous state combinations resulting from the evaluation phase.

III. AutoSHIELD

As part of our goal to automate the SHIELD methodology (AutoSHIELD), we focused initially on a toolset for automating the evaluation phase, in order to reduce the level of effort for conducting the hazard analysis, without sacrificing accuracy or overlooking possible hazardous state combinations.

A. Bayesian Networks for Hazard State Analysis

The evaluation phase relies on analyzing the system hazards and risks using Bayesian Networks [16, 17]. BNs are capable of computing complex predictions with incomplete or uncertain information, not possible with other classical statistical reasoning methods. The evaluation phase applies the BN generated during the system decomposition phase annotated with conditional probability distributions to perform what-if analysis. This provides a measure of risk for components in scenarios where different subsystems are "observed" to be in various hazardous states. BNs also help rank subsystems or environmental factors based on the gravity of the risk they pose to the system operation.

A component c_i resulting from a top-down system decomposition (the first phase of the SHIELD methodology) can be in a state $x_i \in S_i$ (S_i is the state space of component c_i). The current state X_i of component c_i is a random variable (RV) with the state probability distribution (SPD) $P(X_i)=P(X_i=x_j)$, the probability that component c_i is in state x_i. The set of all state RVs is Ξ .

The top-down system decomposition defines for a component c_i the set $\delta(c_i)$ of subsystems on which c_i is dependent on. $c_j \in \delta(c_i)$ if c_j is a subsystem of c_i, interacts with it or c_j models an environmental factor on which c_i depends on. For instance, for a *wireless link* subsystem and a channel noise environment factor, the parent set of the *WirelessLink* random variable can be defined as $\delta(WirelessLink)=\{modem1, modem2, noise\}$. The link subsystem is composed by two modem devices and depends on the state of the wireless channel noise. To keep notation simple we denote state RVs with the component name. Nodes in the set $\delta(c_i)$ are parents of node c_i in the system BN. By construction, X_i is conditionally independent of all other nodes in the network given its parent set $\delta(c_i)$.

The Conditional Probability Table (CPT) for random variable X_i defines the probability $P(X_i \mid \delta(X_i))$ of component c_i being in state $X_i=x_i$ conditioned on parent state random variables. The Bayesian Network is fully

defined by the RV nodes X_i, directed edges (X_j, X_i) where $X_j \in \delta(X_i)$, and a CPT for each state RV.

Risk analysis methods [16, 17] that use BNs may rely on a numeric metric to measure the risk to the system using probabilistic methods. This metric in decision theory corresponds to utility. We define the Impact Factor $IF_{i,j}$ of a state RV X_i in a state $s_j \in S_i$ to describe the "severity of impact" component's c_i state has on components c_k that depend on c_i ($c_i \in \delta(c_k)$). The IF ranges from 0 describing an ideal condition to $IF_{max}>0$. For instance, a wireless modem state RV has IF=0 when there is no noise on the wireless channel, IF=5 when the channel is degraded but still usable, and IF=9 when there is a complete outage.

The full joint state probability distribution is the probability of a conjunction of a specific assignment of state variables, $P(X_1=x_1, ..., X_n=x_n)$ can be computed [14] with the formula:

$$P(X_1, ..., X_n) = \prod_{i=1}^{n} P(X_i \mid \delta(X_i)) \quad (1)$$

The joint SPD helps answer interesting questions about marginal component SPD and IF expectation. In practice modeling tools and APIs facilitate the use of BNs for a large variety of domains [15] without requiring computation of the joint SPD, using variable elimination or approximate inference with Monte Carlo simulations [14]. Our evaluation method generates the BN topology from the system architecture and allows the user to supply the state space S_i, the CPT_i, and the IF_i for each component. It is possible to automatically generate the state space, the IF, and the CPT for subsystems that follow templates, thus reducing the specification effort needed by the user. We will describe this process later in this section. Notice that explicit specification of these parameters require domain expertise and is mandatory for leaf BN nodes.

B. The Evaluation and Presciption Phases

With the BN implemented, the evaluation phase results in the following features:

1. State and impact factor prediction

If the user describes a scenario where an evidence subset $\boldsymbol{e} \in \Xi \setminus X$ of state RVs are known, or "observed" and the system computes the conditional SPD for a subsystem state variable X conditioned on the evidence variables by marginalizing over the *hidden* state RVs, $\boldsymbol{y} \subset \Xi \setminus \boldsymbol{e} \setminus X$ using the joint SPD:

$$P(X \mid \boldsymbol{e}) = \alpha \sum_{\boldsymbol{y} \in \Xi \setminus \boldsymbol{e} \setminus X} P(X, \boldsymbol{e}, \boldsymbol{y}) \quad (2)$$

Here α is a normalization constant. If X is the state RV of the top-level subsystem, then (2) tells us the state distribution of the entire system which depends on the state of some known subsystems and environment factors. The most likely state and the expected impact factor can also be determined.

2. Hazard state diagnostic

If the top-level system state is set to a hazardous state (and other state RVs may be set in various scenarios) and the SPD of key components are derived and compared with their nominal specification (such as MTBF). This feature propagates higher level hazard state down to subsystems allowing the user to diagnose the most likely cause of failure.

3. Subsystem hazard impact ranking

If the method "observes" a top-level subsystem c to be in a hazard state and then computes the SPD of selected RVs to determine their expected impact factors. These RVs ranked by the expected IF can be used to prioritize design efforts to reduce system-level risk.

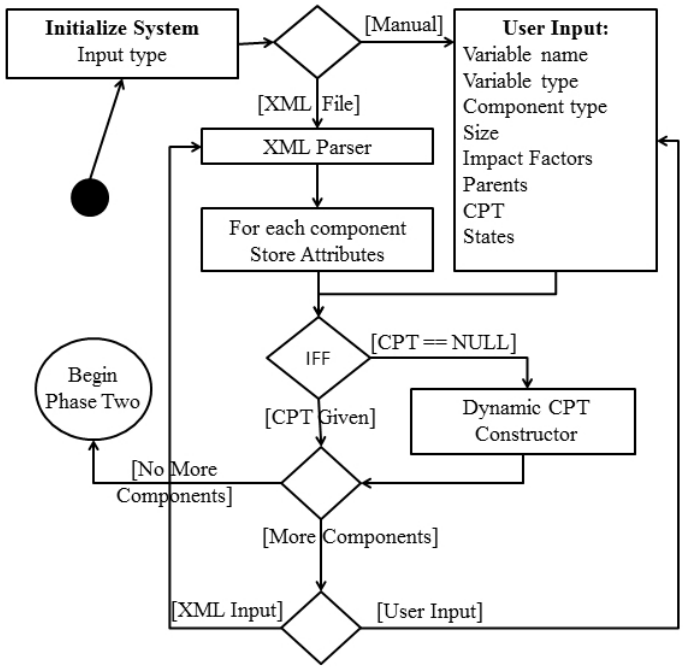


Fig. 1. Activity diagram for phase one, system decomposition.

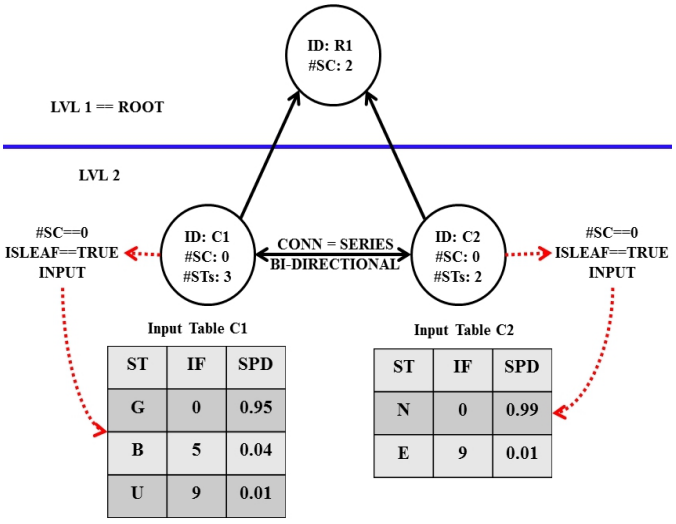
System designers use the results from the evaluation phase to adjust the architecture, perform "what if" analyses, and select components that reduce risk while complying with performance requirements. During the prescription phase, the engineers apply heuristics from [12] to reduce system hazard. As future work, we will develop an algorithm for the prescription phase that recommends what heuristics to apply and how, i.e. modifying the system design automatically under different types of constraints such as physical limitations

(weight, volume), power, cost. For example, the redundancy heuristic suggests adding redundant backups for critical components. Our BN-based evaluation method can assess quantitatively the reduction in risk when these heuristics are used, comparing the top-level SPD and the expected IF before and after the heuristic is applied.

In Figure 1, we show an activity diagram for the decomposition phase. Our tool simplifies decomposition by allowing the domain expert to manually input the system components top down using either a simple interface for generating component objects (i.e. states, impact factor, probability distribution, and conditional probabilities) or XML-formatted files describing the system decomposition. Using the component information, the tool first analyzes each component to ensure all fields, including CPTs, have values which are specified by a domain expert. If these parameters are left unspecified for a subsystem c and the subsystem's structure matches an available template, the tool will generate appropriate values based on the SPD, IF, and state space of the analyzed component's parent nodes in the set $\delta(X_c)$, corresponding to its structural subsystems.

C. *System Decomposition and State Space Reduction for Template Subsystem Structures*

To demonstrate how the decomposition phase works in the interesting case when the parameters for a subsystem are unspecified and the subsystem follows a structural pattern, we illustrate an example in Figure 2.



ST	IF	SPD
G	0	0.95
B	5	0.04
U	9	0.01

ST	IF	SPD
N	0	0.99
E	9	0.01

Fig. 2. System decomposition example. A subsystem R1 is composed of two components C1 and C2 and the state space probability distribution and impact factor are specified for both.

A subsystem R1 consists of two subsystems C1 and C2 connected in series. This means that C1 and C2 are a configured so that if either breaks down then the subsystem R1 (composed of C1 and C2) breaks down, too. C1 and C2 are not dependent on each other, but the subsystem R1 depends on both. The state spaces for the C1 and C2 components are:

S_{C1} = {G = Good; B = Bad; U = Unacceptable}

S_{C2} = {N = Normal; E = Error}

The SPD and the IF are given for C1 and C2 in the figure, but are not specified for the target subsystem R1. To generate a suitable state space, SPD, and IF assignment for R1 we utilize a dynamic hazard level partitioning scheme. Initially, a transitional state space for R1 is defined as the Cartesian product of its subsystems' state spaces:

$$\hat{S}_{R1}=\{x=(x_{C1},x_{C2}) \mid x_{C1}\in S_{C1}, x_{C2}\in S_{C2}\}$$

The size of this transient state space is directly associated with the complexity of the subsystem. A trivial transient CPT for R1 is built with the formula:

$$P(\hat{X}_{R1}=(a,b) \mid X_{C1}=a, X_{C2}=b)=1 \text{, otherwise } 0. \quad (3)$$

The impact factors for the states in X_{R1} are computed based on the template followed by the current subsystem. In the case of a structure organized "in series", the system hazard level depends heavily on its direct subsystems or other environment factors modeled as its subsystems. In our example, if either C1 or C2 breaks down, then R1 will suffer, too.

For a ***series configuration***, the IF for a state RV X will be assigned to the maximum IF of its parents in $\delta(\hat{X})$:

$IF(\hat{X}=(x_{1,..},x_m)\in\hat{S})=max(IF_{X_1},...,IF_{X_m})$,where $X_i\in\delta(\hat{X})$. The SPD, CPT, and IF for the transient state space of R1 are shown in Figure 3.

Subsystems can also be organized in a ***parallel configuration***. In this case, they are redundant backups and the impact factor of the system is reduced. We define the IF of a parallel subsystem in a transitional state to be the arithmetic mean of the IFs of its sub-components' states:

$$IF_{\hat{X}}((x_{1,}x_{2,..},x_m)\in\hat{S}_c)=\frac{1}{m}\sum_{X_i\in\delta(\hat{X})}^{m} IF_{X_i}(x_i). \quad (4)$$

Regardless what template is used for a subsystem for which the user has omitted to specify the state space and other parameters, the following method is used: the states in $\hat{X}_{R1}$ are ordered increasingly by their IF and the transient state space is partitioned into a number of clusters. The number of clusters is indicated by the user

in the specification file. The clusters are defined with uniform bins (e.g. intervals [0,4.5), [4.5,9] for two clusters) or with more sophisticated entropy or error-based methods. For each cluster of transient states we generate a new state x for the state RV X of the target subsystem c. The set of generated states is S_c. For a state $x \in S_c$, with $x = \{\hat{x}_{1,..}, \hat{x}_{k_x}\}, \hat{x}_i \in \hat{S}$ its impact factor is the mean of the impact factors of its elements in $\hat{S}$:

$$IF_s = \sum_{i=1}^{k_x} IF_{\hat{s}_i} / k_x \,. \quad (5)$$

The generated (final, not transient) CPT of the subsystem c is defined as:

$$P(X = x \mid \delta(X)) = P(\hat{X}_c \in x \mid \delta(X)) \,. \quad (6)$$

In other words, the conditional probability that component c is in a state x is equal to the probability that c is in any of the transient states making up state x's cluster. Formulas (5) and (6) define the missing IF and CPT for template subsystems

		R1 Transitional States					
C1	C2	(G,N)	(G,E)	(B,N)	(B,E)	(U,N)	(U,E)
G	N	P=1 IF=0	P=0 IF=9	P=0 IF=5	P=0 IF=9	P=0 IF=9	P=0 IF=9
G	E	P=0 IF=0	P=1 IF=9	P=0 IF=5	P=0 IF=9	P=0 IF=9	P=0 IF=9
B	N	P=0 IF=0	P=0 IF=9	P=1 IF=5	P=0 IF=9	P=0 IF=9	P=0 IF=9
B	E	P=0 IF=0	P=0 IF=9	P=0 IF=5	P=1 IF=9	P=0 IF=9	P=0 IF=9
U	N	P=0 IF=0	P=0 IF=9	P=0 IF=5	P=0 IF=9	P=1 IF=9	P=0 IF=9
U	E	P=0 IF=0	P=0 IF=9	P=0 IF=5	P=0 IF=9	P=0 IF=9	P=1 IF=9

Fig. 3. Transient state space, IF and CPT for R1 during the evaluation phase, prior to state reduction. States are color coded according to the impact factor IF of . Green is "low impact", red is "high impact", and yellow is in between.

For the subsystem in Figure 2, the transient state space $\hat{S}_{R1}$ is defined in Figure 3 and color coded according to clusters {(G,N)}, {(B,N)}, and {(G,E), (B,E),(U,N),(U,E)}, with IF=0, 5, and 9, respectively. Each cluster in $\hat{S}_{R1}$ becomes a state in the final reduced state space S_{R1}, shown in Figure 4, together with the CPT and IF computed with equations (5) and (6).

C1	C2	Green {(G,N)}	Yellow {(B,N)}	Red {(G,E),(B,E),(U,N),(U,E)}
G	N	P=1 IF=0	P=0 IF=5	P=0 IF=9
G	E	P=0 IF=0	P=0 IF=5	P=1 IF=9
B	N	P=0 IF=0	P=1 IF=5	P=0 IF=9
B	E	P=0 IF=0	P=0 IF=5	P=1 IF=9
U	N	P=0 IF=0	P=0 IF=5	P=1 IF=9
U	E	P=0 IF=0	P=0 IF=5	P=1 IF=9

Fig. 4. The state space, CPT, and IF for subsystem R1 after reduction from the transitional state space.

The user can adjust the number of clusters and the IF partitioning parameters. Color-coding clusters by the impact factor offers a more intuitive visualization of hazardous states and of subsystems with different hazard risk.

D. *Prescription*

In the final phase of the analysis, prescription, we apply a set of heuristics to the results from the preceding phase. To assist the teams in this phase, our tool suggests relevant heuristics, for the system being analyzed, to the user. Prior to entering the description of system components in phase one, a domain expert must either choose from a list of system types and application domains or describe these attributes in the XML file description of the system. Using the answers given, we attain two key factors, the type of system and the application domain, in addition to a variety of other component information. Using these data, we remove irrelevant heuristics and only recommend those useful for the particular system under analysis. According to [11], a resilient system is characterized by three attributes: avoidance, survivability, and recovery. If two attributes are present in a design, then the third applies automatically. Our system queries the user on these attributes based on the type of disruption encountered, type of system, and application domain.

AutoSHIELD is implemented in Java and runs as a desktop application. We have selected several classes from the EBayes [15] library from CMU to automatically generate bottom-up the BNs for a complete system decomposition from an XML file, using conditional probability distributions either provided by domain experts, or inferred from the structural inter-component dependency relationships as we described earlier.

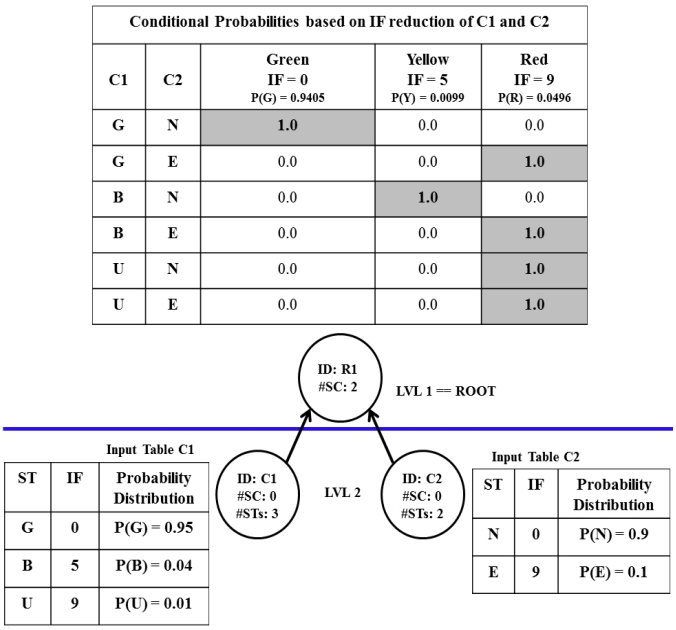
E. *Discussion*

During the decomposition, once the root level is reached and all BN nodes have their parameters fully defined, AutoSHIELD is able to look at particular hazardous states which may stem from processes deeply embedded in subsystem components and extract the contributing sub-component states. Bayesian Network (BN) inference is used to diagnose the low level subsystem component state distribution based on combinations of states for components at a higher level. Bayesian learning [14] is used to bridge the gap between known data and predictions. BNs are beneficial because we may observe the probability that a random variable will be in a particular state, when we have observed the values associated with some other random variables. Additionally, inference with BNs also suggests what the best choice for acquiring new evidence would be.

This capability allows AutoSHIELD to infer sub-component state probability distributions conditioned by the state of other components, possibly at different levels in the design. This feature gives system designers a detailed look at risks present at different levels in the architecture, allows them to identify high risk design areas and failure scenarios, and to efficiently apply

resilient design heuristics (such as adding a redundant component, or increasing operating margins) where it matters most.

The state space reduction method in SHIELD applied to series and parallel template structures reduces the specification and analysis effort for large systems by eliminating the combinatorial state space explosion from the Cartesian state space product. The drawback is lost accuracy for the diagnostic function, as a marginal probability distribution computed for an aggregated state (after reduction) cannot differentiate its individual transient states.



Conditional Probabilities based on IF reduction of C1 and C2

C1	C2	Green IF = 0 P(G) = 0.9405	Yellow IF = 5 P(Y) = 0.0099	Red IF = 9 P(R) = 0.0496
G	N	**1.0**	0.0	0.0
G	E	0.0	0.0	**1.0**
B	N	0.0	**1.0**	0.0
B	E	0.0	0.0	**1.0**
U	N	0.0	0.0	**1.0**
U	E	0.0	0.0	**1.0**

Input Table C1

ST	IF	Probability Distribution
G	0	P(G) = 0.95
B	5	P(B) = 0.04
U	9	P(U) = 0.01

Input Table C2

ST	IF	Probability Distribution
N	0	P(N) = 0.9
E	9	P(E) = 0.1

Fig. 5. The BN annotated with the generated state space, impact factor marginal probabilities for subsystem R1 from the example in Figure 2.

Using the CPT and SPDs for the network, we can infer, for example, what the probability is for C2 to be in state E when we know that R1 is in state Red and C1 is in state Good. This assists AutoSHIELD in finding the most likely causes of particular hazardous states associated with higher level components.

Figure 5 shows how the child, in this case R1, becomes the parent of the next higher level with a CPT generated based on the IFs and CPTs of its parents. The child nodes with the CPT specified or generated become the parents at the next level in the system BN. With inference from the higher level subsystems we can perform queries into the marginal probabilities associated with other RVs. Then we can pinpoint a particular state RV, examine the probabilities for each parent state combination, and identify the most hazardous state with the highest expected impact factor.

We plan to integrate the capability for system reevaluation once risk and hazard avoidance measures, such as component redundancy, are input by the user. We anticipate that this feature will substantially increase engineers' ability to design systems which are highly resilient to any hazards it may encounter by displaying the new probabilities for the state space after the additional measures have been integrated into the design such that optimal mitigation measures may be assimilated into the final design.

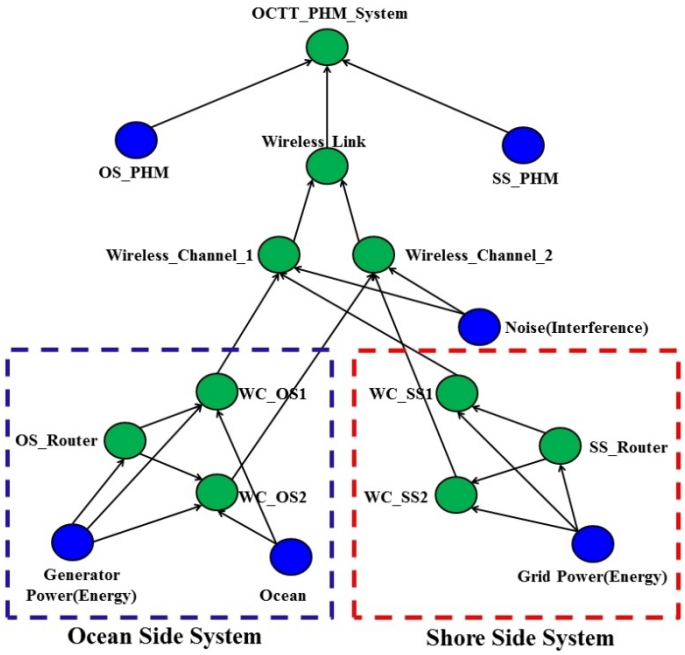


Fig. 6. The Bayesian Network defined for the OCTT PHM architecture [1] used for evaluating the prototype AutoSHIELD system.

IV. Prototype Evaluation

For our initial prototype tests, we used the OCTT PHM general architecture described in [1]. The corresponding Bayesian Tree is shown in Figure 6. We utilized a bucket tree as the data structure of our BN, where each bucket is a node. Each bucket has several attributes associated with it, including name, type, states, IFs, parents, and the SPD (if it is a leaf node) or the CPT. We use this format, as it allows us to conduct multiple queries from a single generation of a BN rather than continuous regeneration for each query using variable elimination. The design used in Figure 6 integrates a redundant wireless link to increase the resiliency of the OCTT PHM System. This backup channel is evaluated in parallel and is utilized to satisfy the Physical Redundancy Heuristic in the prescription phase. This reduces the

probability of overall system failure at a higher level by increasing the capacity, one of the four attributes of systems resiliency, and tolerance to disruptions by adding a physical system which performs the same function. All single hop arcs directed toward a node indicate the parents, and each conditional probability distribution for a node is conditional on those parents.

The leaf node variables are shown in blue. We input all the necessary attributes, states, SPDs, IF values in an XML file for each component except for the WirelessLink node. The OCTT PHM architecture consists of the ocean side PHM (OS PHM) used to monitor turbine health on the ocean platform 25 km from shore, while the SS PHM is used for user interaction and data storage with the OS PHM via the Wireless Link on land. We did not further decompose the OS PHM or SS PHM components for our preliminary testing as our focus was exclusively the wireless communication between the two separated components.

During the first phase, the XML file was properly parsed generating the system decomposition object. When the software encountered the Wireless Link, it generated a CPT based on the SPDs and IFs for the BN parents nodes, Wireless Channel 1 and 2. Having successfully completed the system decomposition, the software then proceeded to the evaluation.

Phase two begins by generating a BN using the system decomposition object and associated attributes. During each iteration of this phase, every set of states in the state space is explored and evaluated using the IF to rank the system impact level for that particular state combination. Once completed, the system evaluation outputs a prioritized list of component states for the top level component, which is the OCTT PHM System in our case. This list shows the most probable subcomponent state combinations (SSC) for each top state where the SSCs are grouped by IF and ordered by the top level state probability.

This output allows us, for example, to see that when the OCTT PHM System is in a fault state and Wireless Channel 1 is in an error state, the probability of the Generator Power being off is 8% and the OS Router in an error state is 56.2%. This information in conjunction with the heuristic suggestions in phase three will assist a system designer in improving the reliability and boosting the resilience of the system in a more efficient manner. In this example, the designer may integrate a backup power supply (a Physical Redundancy Heuristic) or upgrade the router to a more reliable one (Margin Heuristic). When these additional components are integrated, the system may then be re-evaluated until a satisfactory resilience level is achieved.

Another example we model is the effect of channel noise and ocean wave state on wireless channel performance. When the ocean is agitated, the ocean-side communication antenna may be out of line-of-sight and thus lose connectivity, intermittently. In this example, when there is noise at the receiver and the ocean is in the calm state, the wireless channel is bad 75.0% of the time, based on the evidence. We may also backtrack: take the child (wireless channel) and set it in a bad state and see the probability of noise at the receiver. Through BN inference, the answer is 34.14%. Furthermore, if we know the ocean is calm, we may see that the chance of noise reduces to 23.80%. This capability allows experts to test parameters of their systems in a variety of hypothetical scenarios prior to finalization and deployment.

Our initial results from the decomposition and evaluation phases of AutoSHIELD reduced the time needed to conduct the analysis from days to hours when compared to the time required to conduct SHIELD manually. Of course, this time reduction is absolutely dependent on the size and complexity of the system under analysis. As these results represent a work in progress, there are still portions of the application which requires fine-tuning to improve both the accuracy and the runtime needed to conduct the analysis. Automation of phase three, prescription, is in the testing phase and will be presented in detail in a future publication.

V. Related Work

In addition to the related work associated with the development of SHIELD mentioned previously, we looked at several implementations of BNs and their versatility. In [16], the authors extend the concept of simple variable occurrences and results to probabilities over these variables. They compare a variable to an experiment, where, for each run of the experiment, an outcome results which sets this variable to a particular state. Each set of states associated with this variable must be both mutually exclusive and exhaustive. Mutual exclusivity ensures that the variable will only be in a single state at any given moment and exhaustion ensures that it is in a state from its state space. They also introduce Decision Graphs which are used to assist in identifying a likelihood associated with each path to an end node in the acyclic directional graph. This book was a great asset to our conceptual understanding of BNs and Decision Graphs which we used in the automation of our methodology.

In [17], Fenton et al. describe how BNs can be used to model and reason uncertainties. They detailed how BNs may be used to identify, quantify, and understand highly complex interrelations between components of a situation. According to the authors, the use of BNs

greatly assist in understanding exactly how risks emerge, their interconnections, and how we may attempt to mitigate or control them. Furthermore, this added insight may assist future investigation into alternative mitigation measures and consequences that may result from the measures or other alternative actions or effects. Here, they require that all parameters of the BN are known. Our application utilizes several factors as evidence to determine unknown probabilities prior to generating the BNs and is unique to our prototype. We generate BNs hierarchically for each level of a system's decomposition.

The authors in [18] created an application using BNs as a key component to the decision system associated with the assessment of terrorist threats against military installations. The application, named Site Profiler, allows experts to input their current knowledge which is augmented with additional information not accessible or available that may then be utilized to perform complex data analysis employing knowledge-based BN construction. Observing how they integrated additional information into and utilized BNs helped us brainstorm ideas how to integrate BNs into SHIELD and finalize our prototype design.

VI. Conclusions and Future Work

The risk and hazard scenarios within a system's architecture only emerge when analyzing each state combination for each subsystem component. For complex system architectures, the state space may be so large that consideration of all the state tuples may be infeasible. To circumvent this problem, we have demonstrated that a recursive hierarchical breakdown of the system simplifies the analysis by reducing the number of state combinations analyzed at each level of the system.

System states for each sub-component typically differ by system and implementation, and identifying these assists in defining the variations that occur within a system and its components (i.e. a wireless link may have several states of operation, each of which will have a different impact on elements in its process cycle from the other states). Our objective was to find a thorough methodology which incorporates resilience and mitigates hazardous scenarios from a system's architecture, so we developed SHIELD to uncover any vulnerabilities a system may have. This includes causes, consequences, and possible safeguards, by observing the system from multi-dimensional perspectives.

Furthermore, we automated SHIELD to simplify the process of system analysis by reducing the analysis overhead associated with manual risk assessments without losing accuracy and while ensuring no overlooked state combinations. This additional feature, which uses BNs to calculate associated probabilities, allows the user to see the likelihood each subcomponent state contributes to its parent component state. We are refining and integrating additional features into the prototype tool, including automation of the prescription phase, adding the capability to set variables into particular states, and an easy to use GUI with the goal of creating a comprehensive solution for system designers to create new resilient system designs.

References

[1] I. Cardei, A. Agarwal, B. Alhalabi, T.Tavtilov, T. Khoshgoftaar, and P. Beaujean, "Software and Communications Architecture for Prognosis and Health Monitoring of Ocean-based Power Generator." (2011): 1-8.

[2] Marcus, A., I. Cardei, and T. Tavtilov. "Resilient System Design for Prognosis and Health Monitoring of an Ocean Power Generator." IEEE Systems Conference 2012 (2012): 1-8. IEEE, 2012.

[3] Thomas, J., N. Levenson, and T. Ishimatsu. "Performing Hazard Analysis on Complex, Software- and Human-Intensive Systems." NASA 2010 IV&V Annual Workshop: 1-9. Massachusetts Institute of Technology, 17 Sept. 2010.

[4] Levenson, N. "Engineering a Safer World: Systems Thinking Applied to Safety." , 2009.

[5] Nakao, H., M. Katahira, Y. Miyamoto, and N. Leveson. "Safety Guided Design of Crew Return Vehicle in Concept Design Phase Using STAMP/ STPA." (2011): 1-5. IEEE, Oct. 2011.

[6] Kletz, T. "Hazard & Operability Analysis (HAZOP)." (2009): 1-9. Heavy Organic Chemicals Division of ICI, Sept. 2009.

[7] Andrews, J. D., and T. R. Moss. Reliability and Risk Assessment. 1st ed. UK: Longman Group, 1993. Longman Group UK, 1993.

[8] Aven, T. Reliability and Risk Analysis. 1st ed. Elsevier Applied Science, 1992.

[9] Sutton, I. Process Reliability and Risk Management. 1st ed. VanNostrand Reinhold, 1992.

[10] Center for Chemical Process Safety. Guidelines for Chemical ProcessQuantitative Risk Analysis. American Institute of Chemical Engineers,1989.

[11] Suokas, J., and V. Rouhiainen. Quality Management of Safety and Risk Analysis. Elsevier Science Publishers B.V., 1993.

[12] Jackson, Scott. Architecting Resilient Systems: Accident Avoidance and Survival and Recovery from Disruptions. Hoboken, NJ: John Wiley & Sons, 2010. Print.

[13] Viterbi School of Engineering. Studies in the Architecting of Resilient Systems. University of Southern California, May 2009.

[14] Russell, S., and P. Norvig. "Artificial Intelligence: A Modern Approach." (1995): 1-932. Print.

[15] Cozman, F. G. "EBayes Version 0.1." EBayes: Embedded Bayesian Networks. N.p., n.d. Web. 1 Nov. 2012. <http://www.cs.cmu.edu/~javabayes/EBayes/index.html/>.

[16] Jensen, F.V., "Bayesian Networks and Decision Graphs". Springer. 2001.

[17] Fenton, N. E. and M. Neil (2007). Managing Risk in the Modern World: Bayesian Networks and the Applications, London Mathematical Society, Knowledge Transfer Report. 1.

[18] Hudson, L. D., B. S. Ware, K. B. Laskey, and S. M. Mahoney. "An Application of Bayesian Networks to Antiterrorism Risk Management for Military Planners." (2005): 1-8. The Air Force Antiterrorism/Force Protection (AT/FP) Program, 18 Nov. 2005.