Resilient System Design for Prognosis and Health Monitoring of an Ocean Power Generator

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

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

Abstract — **In this paper we introduce a new methodology that integrates system resilience engineering and hazard analysis into complex system design. We then demonstrate its performance by applying it to the design of a Prognosis and Health Monitoring (PHM) system for an ocean current power generator. Three common methodologies for system hazard analysis were tested by applying them to the PHM system's network topology architecture; STAMP-based Process Analysis (STPA), Hazard and Operability Analysis (HAZOP), and a Resilience Engineering, Heuristic-based approach. While all three approaches adequately revealed *most* PHM system hazards, which assisted in identifying the means with which to mitigate them, none of the approaches fully addressed the multi-state dimensionality of the sub-components of the system, missing risky and hazardous scenarios. We developed the System Hazard Indication and Extraction Learning Diagnosis (SHIELD) methodology for system hazard analysis and resilient design. SHIELD integrates state space analysis into the hazard analysis process in order to facilitate the location of undiscovered hazard scenarios. Our approach uses recursive, top-down system decomposition with subsystem, interface, and process cycle identification. Then, a bottom-up recursive evaluation is completed where we analyze the subsystem state space and state transitions with regard to hazards/failures in process cycles. This yields a comprehensive list of failure states and scenarios. Finally, a top-down prioritized application of resilient engineering heuristics which address hazard scenarios is prescribed. This final phase results in a comprehensive, complete analysis of complex system architectures forcing resilience into the final system design.**

Keywords - risk analysis, hazard analysis, system resilience engineering;

I. INTRODUCTION

Novel technology development is inherently exposed to high risk in many forms which include cost overruns, performance and fault mode unknowns, safety, and lack of robustness. Therefore, we must build resilient systems capable of foreseeing, withstanding and recovering from traumatic events. This can only be successfully accomplished using a methodology which thoroughly analyzes the state space of a system architecture for all possible risks and hazards. Once these risks and hazards are identified, it is necessary to integrate measures which mitigate them. The application of a set of rules, or heuristics, to find and integrate these mitigation measures is likely to increase the resiliency of a system, reducing the inherent risk in projects.

The Southeast National Marine Renewable Energy Center (SNMREC) at FAU is establishing an ocean current turbine testbed (OCTT), shown in Figure 1, which will be deployed off the Southeast Florida coast in the Gulfstream current. The OCTT is designed to test small scale pre-commercial and experimental underwater turbines which harness energy found in large-scale ocean currents. The OCTT is aimed at allowing early stage demonstration and validation projects an opportunity to be installed in a relevant environment for short-term tests. In addition, a research-purposed ocean current turbine (OCT) has been constructed at the SNMREC to provide researchers a platform to evaluate and develop missing gap technologies that will enable commercial turbine developers to reduce the cost of producing energy and will allow the sector to accelerate commercial turbine evolution.

One such research project, a Prognosis and Health Monitoring (PHM) system [1], is responsible for both the long term health monitoring of a turbine generator system and forecasting the likelihood of future system health reactions, states, and possible failures, *before* they occur. The system consists of sensor and data acquisition, synchronization, standardization, and storage. Sensors monitor various real-time characteristics of an OCT such as vibration, motion, temperature, depth, etc. and the collected measurements are then ultimately made available in a database on shore. Because the OCTT is located ~25km offshore, and the PHM system components are physically located both on and off shore, the system requires two wireless links for data transfer (Figure 2). Environmental conditions experienced by the OCTT will have a major impact on certain attributes associated with these wireless links, affecting parameters such as, line of sight, link quality, bandwidth, delay, and congestion, which present many challenges for the design and development team.

978-1-4673-0750-5/12/$31.00 ©2012 IEEE

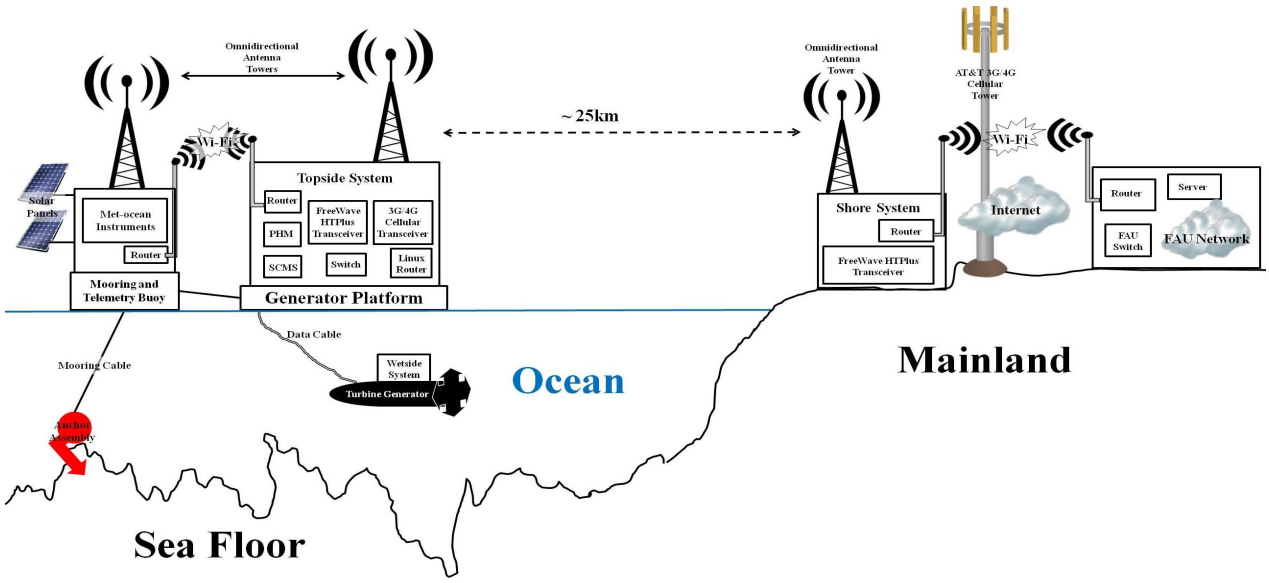

Figure 1: The Ocean Current Turbine Testbed network system architecture

The shore-side PHM system can be accessed or monitored locally or remotely via web server physically located at FAU's Boca campus. The server hosts web services which provide remote observation, adjustment, and control of a turbine. The security measures needed for access control include password protection, MAC address filtering, fixed IP addressing, WPA2 encryption, and disabling SSID broadcast from wireless routers and access points.

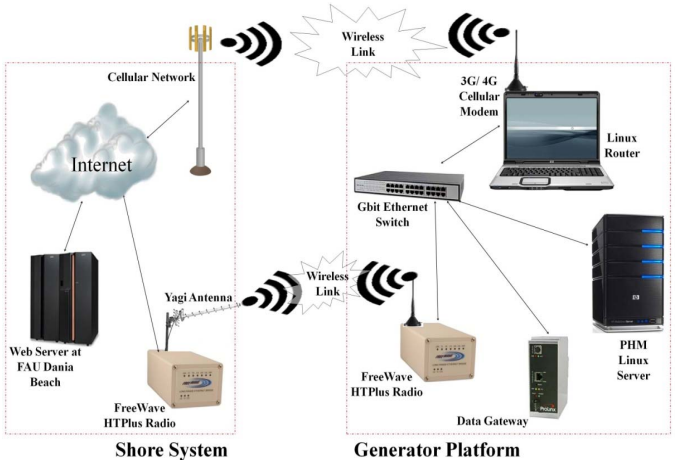

Figure 2: OCTT PHM network topology

Ocean-side PHM components are housed on the generator deployment platform, where they are connected to other data resources, including: a Safety Control and Monitoring System (SCMS), which allows on-site operators to control and monitor a turbine's performance [1], a 1Gbit/s Ethernet switch, and a Linux-based embedded computer which acts as the Data Gateway (DG). The Data Store (DS) web services are handled by a Linux server which also hosts PHM evaluation and analysis software.

The two redundant wireless links used to connect PHM components will be implemented with the goal of improving link quality and reliability, and increasing throughput for data transfer between subsystems. The two wireless links are a cellular 3G/ 4G connection and a FreeWave HTPlus [17] point-to-point RF system. Satellite technologies for communication were not selected because link bandwidth is inadequate and data transfer costs are prohibitive. While other wireless options exist, the selected communication technologies are the most cost effective and optimally suited for the application.

To understand and predict how systems will perform when faced with disruptive or traumatic events and provide recommendations for the mitigation of these disruptions is especially challenging when components are exposed to harsh and extreme conditions found in the oceanic environment. Major disruptive events are intuitively easy to envision in the context of natural events like tornados or hurricanes, but disruptions may also result from technologies and processes. These types of hazardous events and disruptions are much more difficult to foresee and require, prior to finalizing system design, the implementation of some specialized methodology which integrates both system resilience and hazard analysis. Without a thorough systematic analysis, any system is vulnerable to high cost risk and unforeseen, possibly disastrous, consequences.

We applied two methodologies for hazard analysis (STAMP Based Process Analysis (STPA) [2, 3, 4] and Hazard and Operability Analysis (HAZOP) [5, 6, 7, 8, 9, 10]), and a Resilience Engineering [13, 14, 15, 16], Heuristic-based approach [11, 12] to the PHM network system design. Based on our experience with these, we developed the System Hazard Indication and Extraction Learning Diagnosis (SHIELD) methodology for system hazard analysis and resilient design, a new hazard mitigation methodology that takes into account the multidimensionality of a system's state space and the process cycles involved with multiple subsystems. Our new methodology integrates system resilience engineering with a hierarchic and recursive state-based risk and hazard analysis. The two-phase analysis technique provides a thorough

examination which includes software, hardware, and network configurations, as well as human aspects of a system's architecture.

Section II provides an overview of the hazard analysis and resilient engineering methods applied to a preliminary study. Section III introduces our new SHIELD methodology and describes its application to the PHM system. The article concludes in Section IV with a summary of our contributions.

II. Preliminary Study

To reduce development and operational costs associated with highly complex systems, new systematic methods of risk and hazard evaluation have been created. Research teams have developed very thorough methodologies that implement various ingenious concepts to identify and hazardous scenarios. We selected and tested three such methodologies for system hazard and risk analysis/ resilience engineering for the PHM architecture. These were STPA (STAMP Based Process Analysis) [2, 3, 4], Hazard and Operability Analysis (HAZOP) [5, 6, 7, 8, 9, 10], and a Heuristic-based approach [11, 12].

STPA [2, 3, 4] is a system hazard detection methodology based on STAMP (Systems - Theoretic Accident Model and Processes) developed by Professor Nancy Levenson from MIT. This methodology is used to analyze system interactions involving software components and assists in the development of appropriate software safety requirements. It views systems as interacting feedback control loops, where software outputs issue control actions to other hardware and software components, and inputs provide feedback about the state of the controlled system. The software automation is controlled by operators that issue commands to the software component and may receive feedback in the form of aural alerts and displays. Once identified, these control loops allow a detailed examination of the interactions between the hardware, software, and human operators. In addition, the control loop diagrams may identify safety-critical components and interfaces, control actions, threats to the safety of the system, and potentially hazardous system states.

We applied the above methodology to the OCTT PHM subsystem design. Before analyzing the hazards associated with the system, the system control structure was mapped out and a diagram of the control loops was generated. This was accomplished by extracting subsystem components and examining all of the interactions between software, hardware, and human operators within the system. The control loops identified the system control structure and system specification information. This allowed safety-critical components and interfaces, the control actions (software outputs), threats to the safety of the system, and many potentially hazardous states to be discovered. Once all the subsystem components and their interactions were identified, combinations of these parts, which together could yield a risky or hazardous control action, were analyzed.

The analysis using STPA was very thorough and identified the majority of system hazards which were then used to develop means to mitigate them. After analyzing the system with this approach, we observed that STPA views safety as the result of control problems rather than those of failure. STPA was designed to be implemented while the system architecture is being designed so that decisions regarding the components made with safety concerns could be accommodated. We found, however, that the resulting control loops and actions only displayed when the system as a whole was in, or transitioning to, a specific state, and as a result, left undiscovered hazardous scenarios. This methodology was capable of finding most of the hazardous control actions in the PHM design.

Following an evaluation of STPA, we applied HAZOP [5, 6, 7, 8, 9, 10] to the same PHM architecture. HAZOP is a structured and systematic technique for system examination and risk management. In particular, HAZOP is often used as a technique for identifying potential hazards in a system and identifying operability problems likely to lead to nonconforming products. HAZOP is based on theory that assumes risk events are caused by deviations from the original design or operating intentions. A unique feature of this methodology that facilitates the identification of such deviations is the use of "guide words" as a systematic means to list various deviation perspectives and help stimulate the imagination of team members while exploring these potential scenarios.

HAZOP is capable of analyzing a system from multiple perspectives, including design, physical and operational environments, and operational and procedural controls. In a design stage, HAZOP is capable of assessing the ability of the system design to meet user specifications and safety standards and locating and identifying system weaknesses. In an operational stage, this method assesses the environment to ensure the system is appropriately supported, serviced, and contained. HAZOP can also evaluate engineered controls such as system automation, operational sequences between components, and procedural controls such as human interaction with the system. The methodology is divided into four phases; definition, preparation, examination, documentation, and team follow-up.

Once the HAZOP analysis was complete, we reviewed whether more explicit definitions or clarifications of these risks were needed and removed any redundancies from the resulting solution/ mitigation list. Following the analysis, system operators and users would typically give feedback with regard to the proposed solutions and whether they satisfactorily achieve the intended goals. If they do not, the particular scenario would be reassessed in order to find a better solution. This also allows detection of new failure points and hazards that were not previously observed. HAZOP proved to be a useful tool with which system hazards were assessed in great detail. While our intention was only to validate the methodology and assess its' integrity, the results were surprisingly thorough, and uncovered detail for many hazards within the process flow.

There were, however, weaknesses of the method worthy to note. First, HAZOP does not assess hazards arising from interactions between process flows and components. Also, HAZOP fails to identify hazards occurring between components when their subsystem states vary. Risks in the analysis were not prioritized or ranked and there was no means to evaluate or prove the effectiveness of proposed solutions and

safeguards prior to implementation (with the exception of expert knowledge). The analysis consumed approximately the same time as STPA, with equivalent levels of detail and completeness. Overall, we found this methodology useful but lengthy to implement, and, when integrated with other analysis tools, would provide a complete means to examine and evaluate almost any system architecture for risk and hazards.

The final methodology evaluated was a heuristics-based approach [11, 12] developed by Scott Jackson at the University of Southern California. His approach is based on the expectation that a resilient system is capable of anticipating and avoiding catastrophic events as well as surviving and recovering from disruptions. A system's resilience encompasses all the processes, disciplines, and infrastructure that need to be in place to insure that traumatic or disruptive events never happen, how survival from such events can be achieved, and how each system component can continue to operate following such events. Integrating resiliency into a system is accomplished by creating a robust infrastructure that designs, builds, operates, maintains, and tests the system. A resilient system may go through three phases; avoidance (where the system takes steps to avoid any damage from possible disruption), survival (where the system attempts to minimize the damage caused by the disruption), and recovery (where the system takes steps to recover as much of its functionality as possible). In this method, once the risk identification analysis is complete, we apply a set of design heuristics described in [11].

According to [11, 12], a resilient system should include at least two of the three aforementioned phases. *Avoidance* describes the preventative aspects of a system's' resilience to internal or external disruptions. It goes beyond the traditional system safety considerations to include anticipation of an accident based on the ability to detect a system's "drift toward brittleness" and possible accidents. Therefore, during the avoidance phase, the system will take steps to avoid any damage from a possible disruption. *Survival* indicates that the system has not been incapacitated or destroyed and continues to function when experiencing a disturbance. To survive phase two the system will attempt to minimize the damage caused by the disruption. Finally, *recovery* is the capability of a system to survive a disturbance with reduced performance. Recovery is the goal during the final phase where the system will take steps to recover as much of its original capability as possible.

In this method, a resilient enterprise system is tasked with achieving system resilience. If avoidance is possible, then survival and recovery are automatic outcomes, and the enterprise system's job is done. However, some systems may be so complex that avoidance may not be feasible, and therefore, system resilience would be achieved through survival and then recovery of the system.

Heuristics are dependent on three factors: type of system, achievable phases of resilience, and types of disruptions. Some heuristics apply to all system types, including governments and infrastructure systems, while others apply to product-centered infrastructure systems, technical systems, technical systems with human components (such as pilots or maintainers), or socio-ecological systems in which there is human intervention. The second factor, achievable phases of resilience, was detailed earlier and the third factor is the type of disruption present. There are two types of disruptions: the degradation of input, including any unexpected environmental changes, and degradation of function, capability, or capacity.

Upon completion of this preliminary study we found additional hazardous scenarios which were not revealed by each analysis. This incited the creation of a new methodology, inspired by these three methodologies.

III. System Hazard Indication and Extraction Learning Diagnosis (SHIELD)

Following each analysis, which compared the results observing several metrics, including level of detail, time duration, and completeness, each methodology was analyzed and loopholes were discovered that left undiscovered risks or hazards. By determining solutions which would close or remove these risks and hazards, we formulated SHIELD, a novel approach which integrates concepts from other common methodologies. SHIELD is designed to be conducted by members of a design team like engineers, experts, and managers during any stage of the technology development cycle, instead of only being useful, like other methodologies, prior to finalizing component selection.

We integrated a systems thinking approach [18, 19] because, in order to conduct a thorough analysis of any system, teams must observe the cycles of interacting components instead of just the linear interactions between them. The systems thinking approach is a cyclical process of analyzing and understanding how components within a system interact to benefit/ handicap the system as a whole, rather than individual parts. To achieve maximum completeness, any risk and hazard analysis must take into consideration that a system and its components may have multiple states. In the development of SHIELD, we determined that not only is it important to consider each process cycle within a system, but it is equally important to consider the multi-state dimensionality of each component in the system.

To ensure that all component states are considered when using the SHIELD methodology to analyze a system, we observe process cycles occurring within the system. A process cycle is defined as the sequence of events and component states involved in the completion of any process, to include inputs, outputs, actions and reactions between the sub-components involved in the particular cycle. A system typically has process cycles for every operation it conducts, whether they are indirectly dependent on outside circumstance or completely autonomous.

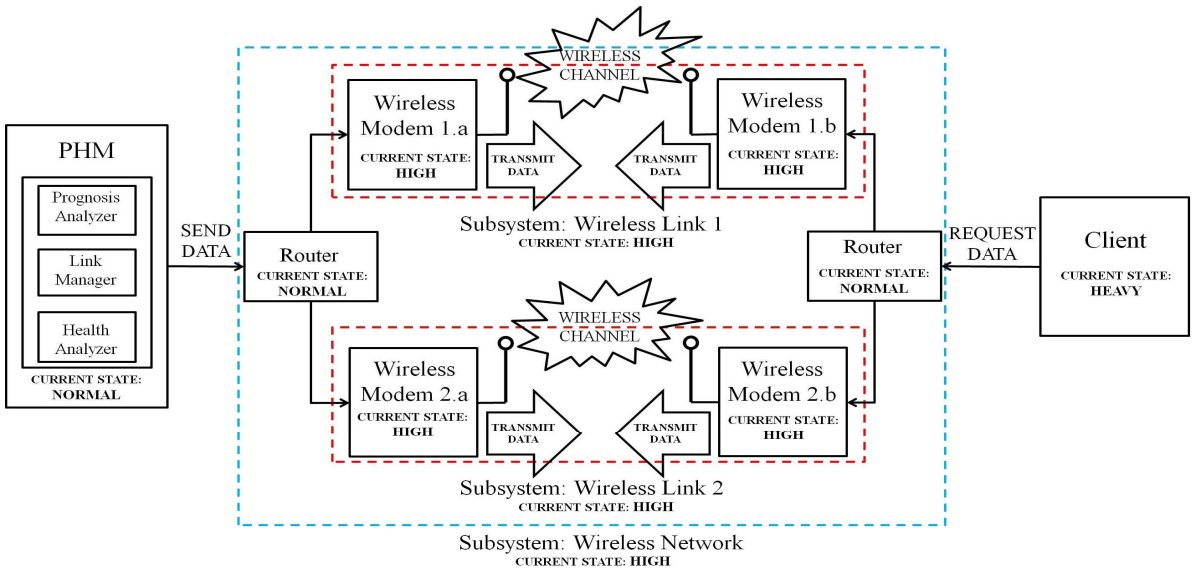

Figure 3: Example of a process cycle with the block system architecture

A simple example which introduces states as used in SHIELD and which demonstrates the necessity to consider the multi-state dimensionality of the various sub-components is shown in Figure 3. Here, we show a process cycle where a client switches its state from 'off' to 'heavy' (as in heavy load) and sends a large data request to the PHM server. Simultaneously, the PHM system sends a long message (heavy load) to the client side through both radios, which transition their states from 'idle' to 'high'. At this point, with their state shifted to 'high,' both radios consume their fully allotted share of data rate capacity while transmitting data to the client-side. With all four radios attempting to transmit heavy loads simultaneously, many hazardous scenarios may occur, including transmission delay and loss. If the data from the PHM server contained alarm information about a turbine that was about to overheat, and the data was lost without being received, major damage may result that a resilient system could have foreseen and avoided. Without consideration of a system's multi-state dimensionality, as in SHIELD, a major scenario would not have been revealed by the other three common methodologies.

A. The SHIELD Methodology

For a system, the aggregate system state space is the Cartesian product of the subsystems' state spaces. The size of a system state space is combinatorial and can be huge. It is important to note that some states are hazardous and need to be flagged, while other state combinations may not be feasible and can be disregarded. To analyze a system with a very large number of state combinations would be time consuming and would render any hazard identification methodology impractical. For this reason, we rely on a top-down recursive hierarchical breakdown of the system prior to beginning the analysis. This decomposition identifies subsystem boundaries, interfaces, and dependencies in process cycles. The decomposition is continued recursively for each subsystem until newly identified subsystems are easy to describe (e.g. COTS parts) and are well understood.

For a subsystem, state space is defined by characterizing subsystem aspects relevant to the hazard analysis. For instance, for a network router, the relevant states may be {*normal, off, error*}. States could describe resources, services, capabilities, and may span several aspects, such as a state *congested*, defined for the router when facing increased demand while the data rate on adjacent links is diminished.

The method starts from the bottom of the system hierarchy (leaves) where state space is defined and analyzed for each component recursively towards the higher levels (branches, etc.). A state space is defined and hazard analysis performed for a subsystem at a level *i* in the architecture decomposition, relying on its subsystems defined at level *i*+1. We consider the top-level system at level 1. The hazard analysis looks at the combination of states for each subsystem, identifying faults and hazardous combinations. When analyzing a subsystem at level *I,* a state transition diagram is defined for each of its subsystems at level *i*+1, describing the possible transitions between states. State transitions for a component at level *i*+1 are dictated by that component's state and the states of sibling components at the same level, *i*+1.

To prepare for analysis at level *i*-1 (higher level) the state space for a subsystem at level *i* is defined. The state of a component can be defined accurately by the tuple made of the states of all its child subsystems. But this notation leads to a combinatorial number of possible states for non-leaf (composed) subsystems, and it is not practical for analysis at a higher level. Many such states may be qualitatively equivalent for hazard analysis, so the state space for a subsystem is therefore defined to be composed of representative states that make sense for analysis at the higher level. Thus, a state for a level *i* component *represents* a subset of the Cartesian product of the level *i*+1 state spaces that is equivalent for hazard analysis.

For example, consider the 'Wireless Network' subsystem from Figure 3. It contains two 'Wireless Link' subsystems. Each of these subsystems contains two components, 'Wireless Modems' labeled *.a, *.b. For each 'Wireless Link', each of the modems in the subsystem can be described by a state indicating the quality of the incoming signal, S_{WM} = {*normal, lossy, down*}. For these modems, the Cartesian state space for the higher level 'Wireless Link' subsystem would be:

$$S_{WL} = S_{WMa} \times S_{WM\,b} = \{(normal, normal),..., (down, down)\}$$

This results in 9 possible state combinations. Considering that wireless data link protocols require a symmetric link to be fully operational (so that link-level ACKs are received correctly), we can reduce the state space for the 'Wireless Link' subsystem to S_{WL} = {*normal, lossy, down*}. Here, its *normal* state is when the modems are in the *normal* state: $normal_{\mathrm{WL}} \equiv$ {(*normal, lossy*)}. Similarly, $lossy_{\mathrm{WL}} \equiv$ {(*normal, lossy*), (*lossy, normal*), (*lossy, lossy*)}$_{\mathrm{WM}}$, etc. In effect, the unnecessary details from the lower level i+1 are hidden to the analysis at the higher level, which reduces the complexity of the analysis and keeps the state space within realistic bounds.

This state-based approach for analysis is quite different in concept and purpose from UML state diagrams [20]. UML state charts are used for behavior analysis of an individual component. SHIELD defines component states based on resource states, constraints, load/ demand, and other relevant factors pertinent to hazard analyses. Its state transitions do not reflect guard/ action-based component behavior; rather, they rely on states of sibling components or internal behaviors. However, the UML Hierarchically Nested States behavior analysis provided the inspiration for the hierarchical recursive hazard analysis in SHIELD and its approach to reducing analysis complexity. In addition, UML nested states have different semantics than those use by SHIELD, where a component state describes a representative state aggregate for the underlying subsystems.

B. Implementing the SHIELD Methodology

The Implementation of SHIELD requires that three phases be conducted: *decomposition, evaluation* and *prescription.* The ***decomposition*** phase breaks down the system recursively and hierarchically into subsystems, identifying interfaces, inputs, outputs. This is a standard engineering method for breaking down complex system architectures into simpler subsystems that together fulfill all system requirements. The project manager assigns top-level subsystems to departments or engineering teams for further refinement using the same decomposition approach. The concept of decomposition is used to reduce the state space which must be analyzed by each team. The threshold value at which this recursive decomposition halts is determined by the project manager, who must decide where components have a practical number of state combinations that can be analyzed by the teams.

The ***evaluation*** phase is conducted recursively bottom-up for each subsystem. After all lower level subsystems have been evaluated, the analysis teams assigned to subsystems at the lowest level, analyze the possible state combinations and state transitions for their subsystems and their interactions looking for possible hazard conditions and scenarios. Following this step, the team defines a state space for the evaluated subsystem that is representative for the hazard analysis. This method continues upward, recombining the levels of interacting components, until the top level is reached and the entire system is again viewed holistically. The project leadership then compiles all the hazardous states, interactions, and inputs/ outputs between the subsystems for each sub-level of the system. This reduces the systems' state space complexity such that each team must only analyze a reasonable number of state combinations.

The evaluation phase of a subsystem at level i may begin only after all its lower level subsystems (at level i+1) have already been evaluated. To fulfill this requirement and to effectively find all possible hazardous scenarios, the project manager ***must*** identify each of the necessary subsystems using the aforementioned top-down, recursive approach and assign design and analysis teams (i.e. experts, engineers, scientists) to each of these parts for the methodology. The steps involved in conducting the evaluation analysis are outlined below:

I. Analyze the subsystem, identifying and diagramming process cycles involving lower level components.
   a. For each process cycle: Note dependencies and adjacent subsystems required to complete each process cycle (this will be used in a later step in the analysis **).
      i. Compile the following information into a table:
         1. All possible states (e.g. off, idle, normal, heavy, error) of the individual components involved in the process cycle diagram.
         2. All possible inputs/ outputs, actions/ interactions between components in each of these states.
   b. For each element in the resulting table, discuss and note possible component failure states, state combinations, and possible hazardous scenarios.
   c. Remove any irrelevant redundancies and unfeasible state combinations.
   d. Generate a representative state space for the subsystem analyzed that captures normal operation and failure cases. This state space will be used for evaluation at the higher level.
   e. Compile failure states, combinations, and scenarios into a list for each system state, by subsystem.

II. Each subsystem team then, using the adjacent subsystem dependency list (from ** above), meets with the associated personnel from the other design teams and together conduct step I. This is a hierarchical, bottom-up, recursive process that should be repeated until the entire system is analyzed as a whole by the entire development team and their assigned sub-components.

The ***prescription*** phase applies a set of heuristics [11], developed from basic design principles, to the results of the analysis from the preceding phase. The objective of this phase is to find possible solutions to each of the scenarios identified. The use of these heuristics integrates the four attributes

required for system resiliency: capacity, flexibility, tolerance, and inter-element collaboration. *Capacity* is the ability of a system to absorb or adapt to a disruption without a total loss of integrity. *Flexibility* refers to a system's ability to restructure itself in response to disruptions. *Tolerance* refers to a system's ability to adapt to disruptions, meaning that near the boundaries of its performance, rather than abruptly halting, its functional capacity gracefully degrades. The adaptability of a system requires that the system understands its current state *and* that of the impending incident in order deviate from the path that leads to this event. The last attribute, *inter-element collaboration* defines inter-element communication and cooperation between elements. Inter-element collaborations, in the context of a human system, may lead to local problem solving and how local adaptation can influence strategic goals and interactions.

Having completed the evaluation phase, the design team will possess a very thorough list of possible risk and hazard scenarios to be used in the prescription phase outlined below.

Prescription Phase:

I. For each element of the risk and hazard scenarios list:
   a. Apply relevant heuristics from the heuristic set
   b. Brainstorm ideas for each scenario and determine possible measures which may mitigate or alleviate the hazardous activity/ situation
   c. Remove redundancies from this list and consolidate measures which may alleviate several scenarios
   d. Prioritize the list according to each item's level of impact on the system functionality.

II. Once complete, all design team members must meet to review and finalize results. A list must be prepared to include precautionary measures, necessary mitigation solutions, and any possible hazardous scenarios for system architects before the design is finalized

After applying SHIELD to analyze the PHM architecture, we found that our methodology was able to pinpoint possible hazardous and risky scenarios not found when conducting the other analyses. A simple example was presented in Section III which was used to define state and describe process cycle. In Table 1, we see the state possibilities for each component in a single process cycle from that example.

There are some groups of state tuples which were easily identified as possible hazardous state combinations, such as [*a, *b, 5c, 5d, *e, *f]. This state tuple group describes the case where both wireless links are in an error state. If this is the scenario, it is irrelevant what state the other components are in because we know there will be no communication between the two components (ocean/ land), resulting in hazardous scenarios. A slightly more complex tuple would be [4a, 3b, 5c, 4d, 3e, 4f], which describes the case where the PHM component is sending a heavy load of data (i.e. sensor and environmental metrics) to the shore and one of the two wireless links is in an error state. The second wireless link is currently under heavy load by a land-based client updating several critical software components on the ocean-based system. We identified this tuple as being prone to hazardous occurrences, such as the case where the PHM system attempts to send an alarm message to the shore. This alarm may require immediate attention by personnel, and if not attended to, may result in costly equipment damage.

Component	State (ID #)				
PHM (ocean)	Off (1a)	Idle (2a)	Normal (3a)	Heavy (4a)	Error (5a)
Router (ocean)	Off (1b)	Idle (2b)	Normal (3b)	Congested (4b)	Error (5b)
Wireless Link #1	Off (1c)	Idle (2c)	Normal (3c)	Heavy (4c)	Error (5c)
Wireless Link #2	Off (1d)	Idle (2d)	Normal (3d)	Heavy (4d)	Error (5d)
Router (shore)	Off (1e)	Idle (2e)	Normal (3e)	Congested (4e)	Error (5e)
Client (shore)	Off (1f)	Idle (2f)	Normal (3f)	Heavy (4f)	Error (5f)

Table 1: Component states by component and associated id's

Having identified the state tuples and scenarios that may result in risky or hazardous outcomes, the prescription phase of SHIELD is then applied. Applying the system resilience heuristics from [11], we were able to find solutions that reduce the possibility of, and may even completely mitigate occurrences of these scenarios. Using the previous scenario as an example, we choose a single heuristic from two of the four attribute groups which define a resilient system so that we may demonstrate their use. From the capacity group we apply 'The Margin Heuristic,' that states *a system should have an adequate margin to absorb disruptions*. Knowing that such an alarm is critical, we have developed QoS mechanisms which allocate data link capacity by message priority, such that the system does not lose essential functionality with only a single operational wireless link. This is one solution to fulfill the requirements of this particular heuristic. From the tolerance group, we use 'The Prevention Heuristic' which states that *a system should be able to suppress future disruptions*. To fulfill the requirement of this heuristic, we will prefetch frequently accessed PHM data to a shore data cache in order to reduce sporadic intervals of heavy data rate usage, reducing the risk of data and equipment loss.

We noticed a trade-off in hazard analysis methods: as we gain greater precision and completeness, we spend more time conducting the analysis. SHIELD alleviates this tradeoff by reducing the time required to analyze all combinations that exist in the systems state space by integrating a recursive, hierarchical breakdown, while maintaining the precision achieved by analyzing each possible state combination of the system. Future work will include a software solution to automate the SHIELD methodology, which we believe will further reduce the analysis time and simplify the entire process.

IV. CONCLUSIONS

The risk and hazard scenarios within a systems' architecture only emerge when analyzing each state combination for each subsystem component. In section III we describe an example which justifies our claim that certain hazardous state combinations within a systems state space may only be found when considering all possible state tuples in this space. 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 hierarchal breakdown of the system will simplify the analysis by reducing the number of state combinations analyzed at each level of the system.

System states and those for each sub-component typically differ by system and implementation, and assist 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). Since our objective was to find a thorough methodology which incorporates resilience and mitigates hazardous scenarios from a systems' architecture, 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, SHIELD simplifies the process of finding solutions with the use of a set of standardized resilient system-building heuristics [11].

We applied three common methodologies to the PHM network architecture to demonstrate and capture their strengths and weaknesses. While the resulting analyses were extensive and thorough, there were circumstances that resulted from state combinations that were overlooked by each methodology that resulted in open, unexplored gaps where hazardous scenarios were found. We consequently developed the SHIELD methodology which integrates a hierarchal, system-based approach with strong concepts from each of the other three methodologies. In addition, we found it useful to integrate the Resilience Engineering heuristics from [11] into the prescription phase of SHIELD. While the selection of heuristics is based on the type of system being analyzed, they are a great asset when conducting a systematic search for solutions and mitigation measures to boost the overall system resiliency.

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. IEEE, 17 Feb. 2011.

[2] 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.

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

[4] 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.

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

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

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

[8] Sutton, I. *Process Reliability and Risk Management*. 1st ed. Van Nostrand Reinhold, 1992.

[9] Center for Chemical Process Safety. *Guidelines for Chemical Process Quantitative Risk Analysis*. American Institute of Chemical Engineers, 1989.

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

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

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

[13] Hollnagel E, Woods DD, Leveson N. Resilience engineering: Concepts and precepts. Ashgate, Aldershot, England, 2006

[14] Hollnagel E, Nemeth CP, Dekker S. Resilience Engineering Perspectives, Volume 1: Remaining sensitive to the possibility of failure. Ashgate, Aldershot, England, 2008

[15] Nemeth CP, Hollnagel E, Dekker S. Resilience Engineering Perspectives, Volume 2: Preparation and restoration. Ashgate, Aldershot, England, 2009

[16] Hollnagel E. Barriers and accident prevention. Ashgate, England, 2004

[17] FreeWave Technologies, http://www.freewave.com/products/product-290.html

[18] Bahill, T., and B. Gissing. "Re-evaluating Systems Engineering Concepts Using Systems Thinking." *IEEE Transactions on Systems, Man, and Cybernetics* 28.4 (1998): 516-27. IEEE Transactions.

[19] Lamb, C.T., and D.H. Rhodes. "Systems Thinking as an Emergent Team Property: Ongoing Research into the Enablers and Barriers to Team-level Systems Thinking." *SysCon 2008 - IEEE International Systems Conference* (2008): 1-7. IEEE, 2008.

[20] "Introduction to UML 2 State Machine Diagrams." *Agile Modeling (AM) Home Page: Effective Practices for Modeling and Documentation*. Web. 10 Jan. 2012. <http://www.agilemodeling.com/artifacts/stateMachineDiagram.htm