Proceedings of the 35th European Safety and Reliability & the 33rd Society for Risk Analysis Europe Conference
Edited by Eirik Bjorheim Abrahamsen, Terje Aven, Frederic Bouder, Roger Flage, Marja Ylönen
 Published by Research Publishing, Singapore.
doi: 10.3850/978-981-94-3281-3_ESREL-SRA-E2025-P5438-cd

Systems Engineering Approach to DfR

Ryan Aalund

Risk, Reliability, and Resiliency Characterization (R3C) Lab, Department of Systems Engineering, Colorado State University. Fort Collins, CO 80523, USA. E-mail: ryan.aalund@colostate.edu

Vincent P. Paglioni

Risk, Reliability, and Resiliency Characterization (R3C) Lab, Department of Systems Engineering, Colorado State University. Fort Collins, CO 80523, USA. E-mail: vincent.paglioni@colostate.edu

Reliability engineering predominantly approaches product development by separating a system into individual components and working bottoms-up, emphasizing hardware/component reliability. As systems become more complex and interconnected, especially with increasing software integration, these methods fail to capture interdependencies and integration points critical to system reliability. A Design for Reliability (DfR) framework solely focused on hardware neglects the intricate dependencies and risks arising from interactions across components.

A systems engineering approach to reliability emphasizes the entirety of the system, ensuring a comprehensive understanding of how different components—hardware, firmware, and software—function together. By examining the system holistically, this approach uncovers hidden vulnerabilities such as cross-system dependencies, cascading failures, and integration point weaknesses that compartmentalized methods overlook. In conventional reliability models, failures are often treated differently across hardware, software, and firmware without recognizing the critical importance of their interactions. This lack of unified analysis frequently results in missed failure modes caused by combining unique parts that do not arise in component-level assessments. Moreover, by focusing only on individual components, organizations may fail to analyze how these components contribute to the overall system function and whether they meet the customer's operational needs. A systems approach ensures that the customer-facing outputs and functional requirements are prioritized so the end product performs reliably under real-world conditions.

This paper explores case studies in critical and emerging industries, such as aerospace, automotive, and internet-of-things (IoT), to highlight the limitations of current reliability practices. It proposes a systems-oriented DfR methodology that shifts focus from isolated hardware approaches to one that accounts for system interdependencies and integration points. This framework enhances system-wide reliability by incorporating hardware and software alongside modeling, simulation, and cross-disciplinary collaboration, ensuring resilience and addressing customer needs in increasingly complex technological environments.

Keywords: Design for Reliability (DfR), Systems Engineering, Reliability, Product Development.

1. Introduction

Reliability is a cornerstone of product design, ensuring that systems perform their intended functions consistently over their expected lifespans. Failures in reliability can have far-reaching consequences, as highlighted by high-profile incidents across industries. For example, in 2023, General Motors' Cruise division experienced significant failures with its autonomous vehicles. These included inappropriate braking, inaccurate path predictions, and inability to reset after system hangs, leading to accidents and regulatory scrutiny (Krisher, 2023; Bursztynsky, 2024; Korosec, 2023). Similarly, critical systems like Boeing's Maneuvering Characteristics Augmentation System (MCAS) and St. Jude Medical's pacemakers suffered catastrophic failures caused by flaws in specific subsystems, resulting in global groundings and recalls (Osborne, 2017). These examples illustrate that product reliability is not merely a function of individual components but also of their integration within the broader system.

Historically, reliability engineering emerged during World War II to quantify design quality and reduce failure probabilities (Modarres & Groth, 2023). Over time, industries such as nuclear power, aviation, and medical devices developed well-established hardware and

mechanical reliability methods. However, modern products often feature an intricate web of interdependent components—hardware, software, and communication systems—operating within a broader context of third-party integrations, shared software, and increasingly complex supply chains. These dependencies introduce new vulnerabilities, from hidden cross-layer interactions to common-cause failures from shared software platforms.

Complex dependencies make it insufficient to evaluate reliability solely at the component level. For example, a software flaw in a subsystem designed for fault tolerance can inadvertently compromise an otherwise robust hardware design (Marwedel, 2021). Similarly, shared software deployed across different products can create unforeseen common-cause failures as vulnerabilities propagate through interconnected systems (Guo et al., 2017). These dynamics highlight the importance of addressing dependencies and interactions, often overlooked in traditional reliability models.

The growing complexity of modern products necessitates a shift in reliability engineering practices. A systems engineering approach to design for reliability (DfR) expands the focus beyond individual components to consider the full product ecosystem, including interfaces, communications, and dependencies between subsystems. By adopting systems thinking, engineers can better assess how interconnections and integration influence overall reliability. This approach enables not only the identification of potential failure modes but also the mitigation of cascading failures that arise from system-wide interactions.

This paper explores the need for a systems engineering approach to DfR, addressing the critical role of high-level perspectives in designing reliable products. The discussion spans various industries, examining case studies and best practices to highlight how a systems-focused mindset can address reliability challenges. Finally, we propose a framework for integrating systems thinking into DfR methodologies, providing a roadmap for improving product reliability in an era of growing complexity.

2. Systems Thinking

Systems thinking is an approach to problem-solving that views a system as a whole rather than focusing solely on its individual components. It emphasizes understanding the relationships, interdependencies, and feedback loops within the system to reveal how changes in one part can impact the entire system. This holistic perspective is especially valuable in complex systems, where isolated analysis may overlook emergent behaviors or cascading effects. By addressing the components and their interactions, systems thinking enables more effective decision-making, design, and management across various fields.

2.1. *Frameworks*

Systems thinking is more than systems engineering. It is a holistic mindset that bridges the practical application of systems engineering to an abstract understanding of the highly-dimensional problem space (Camelia & Ferris, 2016). Systems thinking, in various conceptualizations, is found across disciplines from biology to engineering, and accordingly, there are different ways to understand the philosophy and application of systems thinking. This section will review the conceptualizations of systems thinking that are most relevant to systems engineering and embedded systems. For a richer understanding of the complexities of various systems thinking frameworks, readers are encouraged to refer to (Camelia & Ferris, 2016).

2.2. *Systems thinking philosophy*

In almost all modern conceptualizations, the philosophical underpinnings of systems thinking are recognized as originating in the "Eastern" philosophies of Asia, the Pacific, and Africa. In contrast with the mechanistic, reductionist view of the world offered in Western philosophy, the tendency among Eastern philosophies was, and is, to view the world as a cohesive, organic, and inseparable whole (Camelia & Ferris, 2016). Modern science and engineering inherited the mechanistic view of its Western philosophy roots, decomposing the complex world into discrete blocks to be studied independently and stitched back together to get an approximation of the whole picture (Capra, 1976). This approach allows us to develop a highly detailed understanding of parts at the expense of understanding the whole.

The mechanistic approach to understanding a system is appropriate when the parts of the system are independent or weakly connected, and the system behaviors can be

described through simple cause-and-effect relationships (Kellam, Walther, & Babcock, 2009). However, embedded systems feature non-linear interactions between many highly interconnected components, feedback loops, emergent behaviors, and environmental interaction. These characteristics indicate that embedded systems will not be understood with a mechanistic approach. Systems Thinking is necessary to capture the complexity of embedded systems fully.

2.3. *Applying systems thinking*

Multiple theories related to Systems Thinking, including Tektology, General Systems Theory (GST), Cybernetics and System-of-Systems theory, have found salience in various fields. Cybernetics and System-of-Systems theories are the most relevant to engineering systems and form the basis for the functionalist methodologies of systems engineering (Camelia & Ferris, 2016). Functionalist methodologies approach systems by using the system's logic to understand the internal relationships and ultimately enable the prediction of system behavior. This is precisely the capability desired by reliability engineers to enable intervention planning.

Approaching the logic of a system requires analysts to apply systems thinking, which can be done through the recursive application of discrete rules, as synthesized by (Camelia & Ferris, 2016):

(i) *Question the system boundary*: Identify the components and functions within the system and those external (to the system) phenomena relevant to system function. This is especially important in embedded systems, where system boundaries can be fuzzy.

(ii) *Question the system logic*: Develop the system's structure that connects its components, subsystems, functions, and objectives. This is critical in embedded systems, which feature complex logic.

(iii) *Question the system interrelationships*: Determine how components, subsystems, and functions interact with each other, including feedback loops and hidden dependencies. These interrelationships may be rich, non-linear, and even emergent in complex systems.

(iv) *Adopt multiple perspectives*: Build the system model with a diverse (in terms of experience, expertise, background, etc.) team to ensure that no aspect of the system is overlooked and minimize the effects of biases. This is critical in embedded systems, where expertise is needed in hardware, software, communications, and peripheral functions.

(v) *Consider dynamic characteristics*: Determine how the system's function and/or structure may evolve. Embedded systems can significantly feature emergent behavior as various pieces are updated and new functionality is added. The emergent behaviors must be included in the system model.

Engineers can capture component-specific information and higher-level system characteristics that impact performance and reliability by iteratively applying the above rules when conceptualizing, modeling, and analyzing embedded systems.

2.4. *Systems Engineering*

Systems engineering (SE) is the main approach used to apply systems thinking concepts to real, complex systems. The iterative SE process is often visualized by models such as the Systems V-model (Figure 1) or Spiral Model. Systems Engineering implements the five components of applied systems thinking identified in Section 2.3 (Camelia & Ferris, 2016). While risk analysis can be incorporated within SE, and risk/reliability analysis can take a systems-level view, a true fusion of the two fields is not well defined (Perreault, et al., 2024).

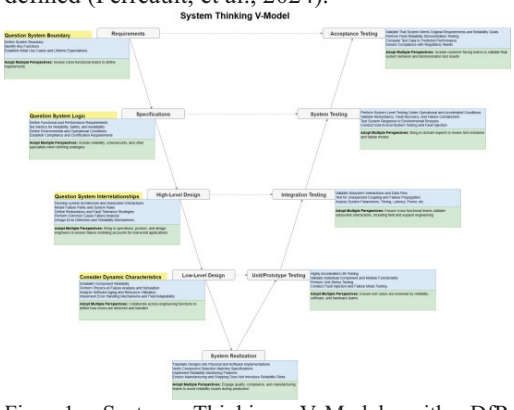

Fig. 1. System Thinking V-Model with DfR Integration. This enhanced V-Model integrates systems thinking tenets to improve reliability across

design phases. The left side represents system definition and development, while the right side ensures verification and validation, reinforcing cross-functional collaboration, failure analysis, and adaptive reliability planning.

3. A Systems-Level Approach to Reliability

As we have seen, component-level approaches to system reliability fail to capture the full complexity of system behaviors. System reliability is not necessarily guaranteed by simply employing high-reliability components. While this remains essential to ensuring reliability, engineers must account for the intricate dependencies and interconnections that define modern products. The challenge lies in developing methodologies that address these complexities, as traditional approaches often fail to capture system-level interactions fully. Here, we offer preliminary insights into the characteristics of a successful systems-level approach to reliability.

System Boundary and Logic. One of the most critical aspects of a robust reliability assessment is the careful definition of the system boundary and a thorough understanding of system logic. Defining the boundary of a product can be particularly challenging. For instance, should peripheral elements—such as optional accessories or third-party components—be included in the analysis? While including such elements ensures a comprehensive assessment, it also increases the scope and complexity of the evaluation.

Determining system logic involves understanding the nonlinear relationships and interdependencies within the product. Processes from software reliability approaches, which often handle complex systems, can be adapted to map these dependencies. Thieme et al. (2020a, 2020b) provide a taxonomy of software failures related to functions, interactions, output values, and timing. Extending this taxonomy to physical components and their interactions is crucial for capturing cascading, hidden, and value-related dependencies. Such an approach facilitates the identification of failure propagation paths, a critical step in building reliable systems. Future research will further refine these systems-level methodologies for reliability assessments.

Adopt Multiple Perspectives. Modern products are inherently multifaceted, requiring software, hardware, communications, manufacturing, and implementation expertise. Developing a comprehensive reliability model requires input from each of these domains and systems engineers to integrate their perspectives into a unified framework. Collaborative efforts ensure that the system model accurately reflects dependencies between components, subsystems, and functions, providing a holistic view that minimizes oversights and blind spots.

Definition and Distinction. A systems engineering approach to reliability is a holistic method that views the product as an interconnected whole rather than as a collection of discrete components. This approach contrasts with traditional hardware-focused methods, which adopt a bottom-up perspective by focusing on individual components and assuming system reliability is simply the aggregate of component reliabilities. Systems engineering, by contrast, emphasizes the interdependencies among components and integration points, identifying how interactions and interfaces can impact overall performance.

By adopting this broader perspective, engineers can address critical aspects of system reliability, such as communication pathways, interdependencies, and cascading effects. While still allowing for detailed analyses of individual components, a systems approach ensures these analyses are contextualized within the larger system (Shneiderman, 2020). This mirrors how end users interact with a product—as an integrated whole rather than as separate parts—reinforcing the importance of a unified, systems-level perspective.

Designing Reliability into Products. A robust methodology for achieving reliable systems begins with clearly defining system boundaries and identifying key interactions. The following sections outline eight key activities in a design-for-reliability (DfR) approach, drawing from systems engineering principles and systems thinking rules (Camelia & Ferris, 2016). These activities provide a framework for building resilient products that address modern reliability challenges.

3.1. *System definition and boundary identification*

(i) Define the System Boundary: A clear and well-defined system boundary is critical to understanding what elements fall within the scope of a reliability assessment. This requires engineers to:

a. Identify Components and Functions: Clearly identify all components, subsystems, and external factors that impact system performance. For instance, in a consumer electronics product, both internal hardware and external power sources may need to be included in the boundary.
b. Question the System Boundary: Regularly evaluate whether the system boundary reflects all relevant elements as the product evolves, ensuring critical factors are not overlooked.

(ii) Determine System Logic: Understanding system logic involves mapping how components, subsystems, and external factors interact to achieve the product's intended function. This includes:
a. Develop System Logic: Define the logical relationships between components, including physical connections, functional dependencies, and data flows.
b. Question the System Structure: Continuously assess and update the system logic as new dependencies or design changes emerge, ensuring an accurate understanding of how the product operates as a whole.

3.2. *Modeling and simulation*

(iii) Create Comprehensive Models: Understanding the system logic, developed in activity (2) above, is critical to constructing usable models of the system, which themselves are crucial to identifying and propagating component failures to understand system reliability. However, the complex interactions inherent in many products mean that traditional reliability modeling, e.g., reliability block diagrams (RBDs), do not adequately describe the system dependencies or behavior, necessitating the inclusion of more flexible techniques such as functional modeling. Comprehensively modeling embedded systems means engineers must:
a. Develop Detailed Models: Build detailed models that reflect the product's physical, functional, and behavioral characteristics. These models should include interactions between components and subsystems, as well as environmental and operational factors.
b. Adopt Multiple Perspectives: Involve cross-disciplinary teams to capture diverse viewpoints and ensure critical aspects of the system are not overlooked.

(iv) Simulate System Behavior: Determining the system logic and creating system models will not necessarily uncover all behaviors of the system, which may be emergent with time and thus not determinable at an early stage in design or operation. Simulating system behavior is critical to validating the system logic and models and identifying dynamic and emergent system behaviors that might have essential reliability effects. This includes:
a. Use Simulations: Test these models and identify reliability issues under different scenarios without costly and time-consuming physical testing. For instance, the RelIoT framework (Ergun, et al., 2020) is a reliability simulator for IoT networks that incorporates power, performance, and temperature models, which are required to model reliability.
b. Consider Dynamic Characteristics: Ensure that simulations account for the system's dynamic characteristics, including emergent behaviors as various pieces are updated and new functionality is added.

3.3. *Error detection and handling*

(v) Implement Error Detection and Handling Mechanisms: Today's systems feature hardware and software components tightly coupled to perform various tasks, and risks/errors may propagate widely. In some cases, system errors may not be readily apparent, leading to dire consequences for users or other reliant systems.

Modern products require robust mechanisms to identify and manage errors, preventing them from propagating and causing system-wide failures. This involves:

a. Design Error Detection Mechanisms: Incorporate diagnostics and monitoring tools to detect anomalies and address failures proactively.
b. Question System Interrelationships: Examine how errors in one subsystem might affect others, ensuring the system is resilient to cascading failures.

3.4 *Integration and testing*

(vi) Ensure Proper Integration: The reliability of a system is not a simple function of the component reliabilities but must also include the multimodal dependencies and interconnections between the components. Simply integrating highly reliable components does not necessarily guarantee a reliable system. Instead of focusing on components, systems engineering requires a focus on integrating components to ensure system reliability. This means that engineers must:

a. Focus on Integration Points: Evaluate the interfaces and connections between subsystems to detect and resolve potential issues. For example, firmware and hardware mismatches might lead to timing errors compromising performance.
b. Conduct Comprehensive Testing: Perform rigorous testing, including stress and scenario-based tests, to validate the integrated system's functionality and reliability.

3.5. *Analysis and continuous improvement*

(vii) Analyze Simulation and Test Results: A critical aspect of designing reliability into embedded systems is using the results of analyses to address reliability issues before system deployment. This means that, throughout the system lifecycle, engineers should develop insights from simulations and tests and use those insights to mitigate reliability concerns. This practice weaves all of the systems thinking questions together as a microcosm of the iterative reliability analysis process.

a. Use Analysis Techniques: Interpret the results of simulations and tests using various analysis techniques. These techniques help pinpoint the causes of reliability issues, predict system behavior, and develop strategies for improving reliability.

(viii) Implement Continuous Improvement: Reliability engineering is an iterative process throughout the product lifecycle. This includes:

a. Monitor and Improve: Continuously monitor the product in the field and use real-world data to refine models and improve designs.
b. Adopt Multiple Perspectives: Ensure that iterative improvements incorporate input from diverse stakeholders, minimizing bias and maximizing robustness.

A systems-level approach to reliability ensures that interdependencies, cascading effects, and emergent behaviors are accounted for, enabling the development of robust and resilient products. By integrating the eight key actions outlined above into the product development lifecycle, engineers can design products that perform reliably and adapt to the complexities of modern systems and their operating environments.

3.6 Considerations for AI-Based Reliability Analysis

Artificial intelligence (AI) has gained significant traction in reliability engineering thanks to its ability to detect hidden patterns, anticipate failures, and optimize maintenance schedules based on large datasets. In certain domains, such as predictive maintenance for industrial equipment (Kong & Pecht, 2018) or anomaly detection in sensor networks, machine learning algorithms have proven successful at reducing downtime and identifying early warning signals of component degradation (Pang et al., 2021). When paired with robust data-collection infrastructures, these AI-driven models can automate many reliability assessments, allowing engineers to focus more on system-level decision-making.

However, these successes often center on subsystems or operational data rather than entire product ecosystems, which require rigorous verification and validation (V&V). Many AI approaches, function as "black boxes," making them difficult to audit in safety-critical or regulated environments that demand high traceability. Without explicit, interpretable models, demonstrating correctness or pinpointing the root cause of AI-driven decisions remains challenging. As a result, purely AI-based approaches may not meet the evidence requirements for certifications or reliability guarantees—particularly in applications where human safety or mission-critical operations are at stake. Nonetheless, research in explainable AI (XAI) is advancing (Gunning & Aha, 2019; Barakat et al., 2021), and such techniques can help bridge the transparency gap by providing insights into model reasoning that are more amendable to industrial and regulatory scrutiny.

In the long term, AI's continued evolution promises to become a major asset in analyzing complex interactions, discovering novel failure modes, and streamlining reliability models. As methods for explainability, model transparency, and certification improve, AI-driven tools will likely merge more fully with systems engineering frameworks. For a broader overview of AI's role in embedded systems reliability, refer to (Aalund & Paglioni, 2025). As AI rapidly evolves, future developments may overcome today's limitations and fully integrate with the holistic, systems thinking mindset essential for designing reliable systems.

4. Conclusion

In conclusion, modern systems' increasing complexity and interconnected nature necessitate a shift from traditional hardware-focused reliability engineering to a more holistic, systems-oriented approach. The traditional methods, which treat system components in isolation, are insufficient for addressing the intricate dependencies and interactions that characterize advancing technologies. By adopting a systems engineering approach, we can better understand and manage the complex behaviors and issues arising from component interactions, improving reliability.

This paper has highlighted the critical need for a systems-level perspective in reliability assessments, emphasizing the importance of considering the entire system as an integrated whole. Through comprehensive modeling, simulation, and analysis techniques, we can predict system behavior under various conditions, identify potential points of failure, and develop strategies for mitigating reliability issues.

The incidents involving autonomous vehicles, aviation, and medical industries underscore the urgency of ensuring the reliability of today's systems, particularly in critical infrastructure. By embracing a systems thinking framework, engineers can capture component-specific information and higher-level system characteristics that impact performance and reliability.

Ultimately, a systems-oriented approach to reliability will enhance our ability to anticipate and address the challenges posed by these systems' growing complexity and criticality, ensuring their safe and reliable operation in an increasingly connected world.

References

Aalund, R, and Paglioni, V. "Enhancing Reliability in Embedded Systems Hardware: A Literature Survey." IEEE Access 13 (2025): 17285–17302. https://doi.org/10.1109/ACCESS.2025.3534138.

Aalund, R., and V. P. Paglioni. "Enhancing Reliability in Embedded Systems Hardware: A Literature Survey." *IEEE Access* 13 (2025): 17285–17302. https://doi.org/10.1109/ACCESS.2025.3534138.

Barakat, B., W. Abualhaija, and R. Taghipour. "Explainable AI for Prognostics and Health Management: A Review on Methods, Challenges, and Opportunities." *Reliability Engineering & System Safety* 209 (2021): 107485.

Bursztynsky, J. "Cruise Recalls Autonomous Vehicle Fleet." *Fast Company*, August 22, 2024. https://www.fastcompany.com/91177678/cruise-recalls-autonomous-vehicle-fleet.

Camelia, F., and T. L. J. Ferris. "Systems Thinking in Systems Engineering." *INCOSE International Symposium* 26 (2016): 1657–1674. https://doi.org/10.1002/j.2334-5837.2016.00252.x.

Capra, F. "Modern Physics and Eastern Mysticism." *Journal of Transpersonal Psychology* 8, no. 1 (1976): [Page range if known].

Ergun, K., W. Song, K. Lee, Z. Yan, L. J. Zhang, and H. Chen. "RelIoT: Reliability Simulator for IoT Networks." In *Internet of Things – ICIOT 2020. Lecture Notes in Computer Science*, vol. 12405, edited by W. Song, K. Lee, Z. Yan, L. J. Zhang, and H. Chen, 78–91. Cham: Springer, 2020. https://doi.org/10.1007/978-3-030-59615-6_5.

Gunning, D., and D. Aha. "DARPA's Explainable Artificial Intelligence (XAI) Program." *AI Magazine* 40, no. 2 (2019): 44–58.

Guo, H., C. Zheng, H. H. C. Iu, and T. Fernando. "A Critical Review of Cascading Failure Analysis and Modeling of Power Systems." *Renewable and Sustainable Energy Reviews* 80 (2017): 9–22.

Kellam, N., J. Walther, and A. Babcock. "Complex Systems: What Are They and Why Should We Care?" In *2009 Annual Conference & Exposition*, 2009.

Kang, M., and M. Pecht. *Prognostics and Health Management of Electronics: Fundamentals, Machine Learning, and the Internet of Things.* Wiley-IEEE Press, 2018.

Korosec, K. "Cruise Recalls 300 Robotaxis, Issues Software Update After Crashing Into City Bus." *TechCrunch*, April 7, 2023. https://techcrunch.com/2023/04/07/cruise-recalls-300-robotaxis-issues-software-update-after-crashing-into-city-bus/.

Krisher, T. "General Motors' Cruise Division Recalls Vehicles for Software Update After Pedestrian Incident." *Associated Press*, November 8, 2023. https://apnews.com/article/cruise-general-motors-pedestrian-recall-software-crash-bf08c0c6e7914649750b4dde598af5fc.

Marwedel, P. *Embedded System Design: Embedded Systems Foundations of Cyber-Physical Systems, and the Internet of Things.* 4th ed. Springer Cham, 2021.

Modarres, M., and K. Groth. *Reliability and Risk Analysis.* 2nd ed. Boca Raton, FL: CRC Press, 2023.

National Highway Traffic Safety Administration. *Safety Recall Report: NHTSA Recall No. 24E-067.* Washington, DC: U.S. Department of Transportation, August 19, 2024. https://static.nhtsa.gov/odi/rcl/2024/RCLRPT-24E067-9924.PDF.

O'Kane, S. "Cruise Recalls AV Fleet as NHTSA Probe Closes." *TechCrunch*, August 22, 2024. https://techcrunch.com/2024/08/22/cruise-recall-av-fleet-nhtsa-probe-closed/.

Osborne, C. "FDA Issues Recall of 465,000 St. Jude Pacemakers to Patch Security Holes." *ZDNet*, August 29, 2017. https://www.zdnet.com/article/fda-forces-st-jude-pacemaker-recall-to-patch-security-vulnerabilities/.

Pang, G., C. Shen, L. Chen, and A. van den Hengel. "Deep Learning for Anomaly Detection: A Review." *ACM Computing Surveys* 54, no. 2 (2021): Article 38, 1–38.

Perreault, D., E. Gallegos, V. Paglioni, and T. Bradley. "Usability Challenges of Failure Mode and Effects Analysis (FMEA) within the V-Model." In *INCOSE Human-Systems Integration Conference 2024*, Jeju, Korea, 2024.

Shneiderman, B. "Bridging the Gap Between Ethics and Practice: Guidelines for Reliable, Safe, and Trustworthy Human-Centered AI Systems." *ACM Transactions on Interactive Intelligent Systems* 10, no. 4 (2020): 1–31.

Thieme, C. A., A. Mosleh, I. B. Utne, and J. Hegde. "Incorporating Software Failure in Risk Analysis—Part 1: Software Functional Failure Mode Classification." *Reliability Engineering & System Safety* 197 (2020): 106803. https://doi.org/10.1016/j.ress.2020.106803.

———. "Incorporating Software Failure in Risk Analysis—Part 2: Risk Modeling Process and Case Study." *Reliability Engineering & System Safety* 197 (2020): 106804. https://doi.org/10.1016/j.ress.2020.106804.