Title: A Methodology for Exposing Software Safety Risk in Early Development Phases

Abstract Text: Problem statement: One driver of system development costs in defense and aerospace is that such systems often have challenging non-functional requirements, such as safety, reliability, security and performance. Assurance processes are applied throughout system development to assure that the final system is functional, safe, reliable, on-time, and so on. However, these emergent properties can only be fully tested when the system is complete. Corrective actions based on the results of final tests or operations are difficult and costly to implement. The challenge is to identify and mitigate risks associated with not achieving desired emergent properties early in the lifecycle. This challenge is particularly difficult for systems involving software, as software assurance processes are often not well-understood or well-defined in the context of system development.

Planned content: We will present the Process Risk Assessment (PRA) methodology for analyzing and evaluating emergent properties such as safety early in the development cycle. The PRA method analyzes data provided by assurance process artifacts (e.g., designs pertaining to reliability, safety analysis reports). By examining the data contained in process artifacts (going beyond simply checking that such artifacts exist), we determine: 1) whether the process itself is appropriate for achieving the desired emergent property; and 2) whether the process is being followed appropriately. Thus, process risk can be quantified to indicate whether the system will have the desired emergent properties. The purpose of the PRA method is not simply to focus on process conformance, but to identify risks in the assurance processes and to create responses to identified risks throughout the lifecycle. Furthermore, by leveraging existing artifacts and processes to find risk indicators, the PRA method introduces little overhead to the engineering team.

The six steps of the PRA method are: 1. Identify insight areas (e.g., process artifacts) from the development process that provide insight into risk areas. 2. Identify measurement opportunities that provide insight into each insight area. 3. Develop readiness assessment questions to identify if sufficient information exists to implement process risk measures. 4. Define goals, questions, and measures for each risk area to expose risks associated with process artifacts. 5. Develop and enumerate models of how the measures will be interpreted via threshold values. 6. Propose responses to identified risks (e.g., decisions and actions) in order to mitigate those risks.

In this presentation, we will describe the PRA method and its steps using concrete examples from case studies of large systems where PRA was applied to uncover software safety risks early in the development lifecycle. These systems include a network-centric, Department of Defense system-of-systems, a human-spaceflight program at NASA, and an unmanned satellite at NASA. In the systems, we applied the PRA method to the hazard analysis processes and process artifacts and identified several process risks in the application of the hazard analysis process itself that may lead to software safety risk.

Contribution: With the PRA method, we identified potential risks in the software safety process, provided feedback to the projects for reducing these risks, and suggested improvements to software safety management processes. The PRA identified three process risks that were common across projects with respect to software safety: 1. Inability to track software safety hazards and requirements – software safety risks were often not specifically marked in the hazard reports 2. Inadequate traceability – No bi-directional traceability between safety requirements, hazards, causes and controls 3. Inconsistent scope and unstructured details – safety engineers on each project wrote their hazards, causes and controls in unique ways.

Based on these risks, we have identified three lessons learned for future programs with software safety concerns: 1. Need to provide more specific guidance that provides clear, consistent information on how to apply safety analyses to software. 2. Need to plan for automated analysis and traceability and promote usage of the hazard tracking system capabilities. 3. Need to require software safety management and measurement in the acquisition process in order for appropriate data to be made available for safety analysis during development.