# NASA's Approach to Software Assurance

**Abstract:** NASA defines software assurance as: the planned and systematic set of activities that ensure conformance of software life cycle processes and products to requirements, standards, and procedures via quality, safety, reliability, and independent verification and validation. NASA's implementation of this approach to the quality, safety, reliability, security and verification and validation of software is brought together in one discipline, software assurance. Organizationally, NASA has software assurance at each NASA center, a Software Assurance Manager at NASA Headquarters, a Software Assurance Technical Fellow (currently the same person as the SA Manager), and an Independent Verification and Validation Organization with its own facility. An umbrella risk mitigation strategy for safety and mission success assurance of NASA's software, software assurance covers a wide area and is better structured to address the dynamic changes in how software is developed, used, and managed, as well as it's increasingly complex functionality. Being flexible, risk based, and prepared for challenges in software at NASA is essential, especially as much of our software is unique for each mission.

**Background**. NASA's system safety and system reliability, has traditionally looked at the software of the system as either "it works" or "it does not work". Not that NASA did not do good software development and develop extensive software fault tolerance approaches, but NASA relied on the hardware for the safety aspects even as software took on more complex and critical functions and most of the roles for fault detection, isolation and recovery. The amount of software in our earliest missions was very small and comparatively straight forward to the missions of today. Originally, Space shuttle and then the International Space Station worked with balancing hardware's safety role with strict development and design criteria for any software that was considered "safety critical". The software safety criteria coming from early NASA projects was so strict, that many projects tried to avoid having their software labeled as safety critical. The system safety teams on many projects often did not have personnel with the software expertise for examining software at the appropriate level. At that time, those system safety teams with limited software resources were not able to go much lower than considering software as a system component that either did or did not work. These teams with limited expertise were unable to take into account the many ways software can fail, let alone why and what the impacts were on the system. NASA, for most of those earlier projects, relied on the hardware. Shuttle and ISS recognized the need and created a special computer software safety committee to review software issues and support the program safety panel(s) (which review the hazard analyses processes for projects from start to acceptance) and help projects when software becomes critical. As software evolved, taking on more and more functionality with growing system complexities, NASA saw the need for software assurance to grow as well. While many felt that providing software process checks and software product evaluations was sufficient, the reliability and safety aspects of software was, and in some cases still is, undervalued. In the 1990's, software safety at NASA was further promoted via an agency standard and guidebook that were produced to lay out the principles of both analyzing the software for contributions to system faults and failures as well as assessing the risk software takes on in reporting and mitigating hardware and system hazards. The lessons learned from the Shuttle software safety processes were incorporated and analyses and evaluation methods were stressed as well as providing support for tailoring the safety effort to the project. Software Assurance and its other sub-disciplines have been growing and evolving as well.

**Software Assurance:** The software assurance process is the planned and systematic set of activities that ensure conformance of software life cycle processes and products to requirements, standards, and procedures. Software assurance assures that the software and its related products meet their specified requirements, conform to standards and regulations, are consistent, complete, correct, safe, secure and as reliable as warranted for the system and operating environment, and satisfying customer needs. Note, scientific principle investigators, many of our customers, sometimes need a continuing discussion to discover what is needed verses what is "wanted" and what is possible as we push forward the principles of science and physics. Thus, some requirements are actually "desirements" where something less is actually sufficient; and sometimes NASA can provide them with more than they knew was possible or alternative solutions.

Software assurance reviews and analyzes all processes used to acquire, develop, assure, operate and maintain the software independently; evaluating if those processes are appropriate, sufficient, planned, reviewed, and implemented according to an adequate plan, meeting any required standards, regulations, and quality requirements. Software assurance utilizes relevant project-based measurement data to monitor each product and process for possible improvements. NASA software assurance has begun to work with the NASA Chief Information Office and Protective Services Office to assess the role of software assurance for mission software security. It is a joint effort between Software Assurance, the Chief Engineers Office, Project and program management as well as the CIO and Protective services to address the many facets of mission development and operational environment security.

At NASA, Software Assurance has evolved in to an umbrella risk identification and mitigation strategy for safety and mission assurance of all NASA's software [Figure 1]. It provides a consistent, uniform basis for defining the requirements for software assurance programs to be applied and maintained throughout the life of that software, that is, from project conception, through acquisition, development, operations and maintenance, and then evaluates if the software is properly retired.

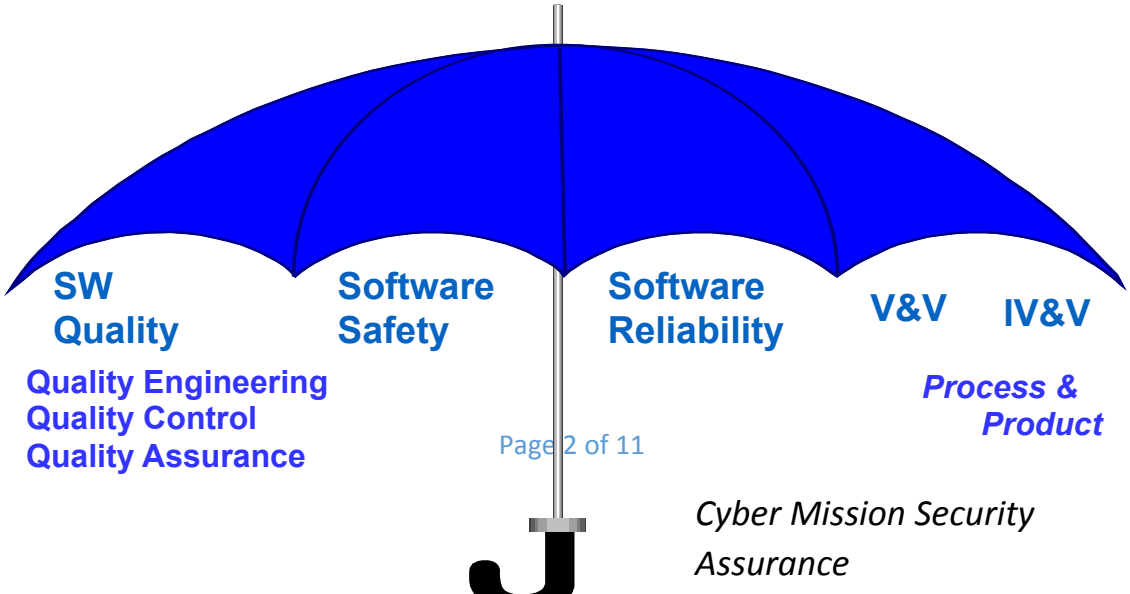

**Figure 1. NASA's Software Assurance Umbrella of Risk Mitigation**

The purpose of software assurance is to assure that software products are of sufficiently high quality and operate safely, securely and reliably. This includes products delivered to and used within NASA, and products developed and acquired by NASA. Software assurance assists in risk mitigation by helping expose potential defects in products and processes, thus preventing problems from evolving. However, it also, through its metrics, tracking and analyses activities, enables improvement of future products and services. Software assurance often serves as the corporate memory from project to project, sharing potential problem areas and lessons learned.

Software engineering and the software assurance disciplines are integrally related and yet each has its own responsibilities. Jointly they are responsible for providing project management with the optimal solution for software to meet the engineering, safety, quality, and reliability needs of the project. This necessitates a close working relationship to establish the appropriate levels of effort for both. The NASA Procedural Requirements, NPR 7150.2, NASA Software Requirements invokes the NASA Software Assurance Standard (NASA-STD-8739.8) and the NASA Software Safety Standard (NASA-STD-8719.13), requiring a close working relationship, understanding of roles and responsibilities, and establishing expected communication paths. NPR 7150.2, besides laying out the NASA minimum requirements for software development, provides the NASA software classification upon which software engineering, software assurance and software safety all base their tailoring. Table 1 shows the NASA Software Classes and a very brief, summary definition of them. The Office of the Chief Engineer "owns" Software Classes A-E as those are used for Mission software and support while the Chief Information Office "owns" the infrastructure software like desktop operating systems and applications, web based applications, etc. which are Software Classes F, G & H. Software Assurance has focused on the mission software.

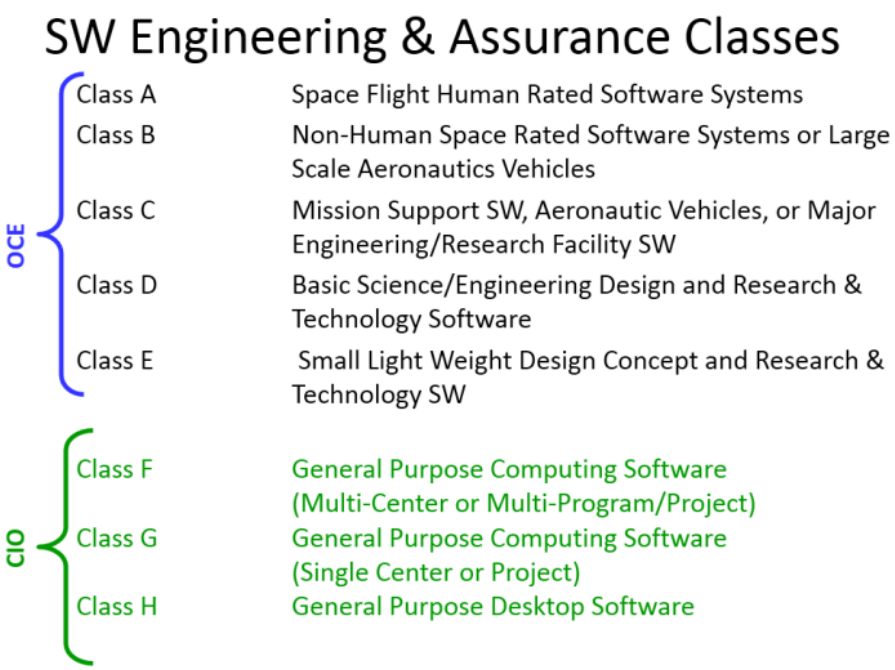


**Table 1. NASA Software Classifications**

The NASA Software Assurance and Safety standards are also invoked from the Agency System Safety, Reliability and Quality policies and procedures, thus stating not just the recognition of software assurance as an explicit special discipline, but also the expectation of software assurance as part of the joint assurance, safety and reliability support to NASA's systems. The struggle is balancing the need for software to be part of the overall system assurance, safety and reliability analyses and having the expertise needed to take those systems analyses down to the proper depth to see the potential impacts of software errors on that system.

The NASA Software Assurance Standard (NASA-STD-8739.8) provides a common framework for software assurance definition, activities, and implementation across NASA and its contractors. It provides tailoring recommendations in order for software assurance planning and execution to meet the needs of different flight, ground, facility and experimental software projects. The NASA Software Safety Standard lays out a systematic approach to software safety as an integral part of the overall systems safety, establishing the activities, data, and documentation necessary for the acquisition and development of software in a critical system. It also defines the levels of criticality for software, starting with a "litmus test" to determine if the software is safety critical (See Table 2 below) or not. Then provides additional risk based scoping based on severity and likelihood of occurrence, level of autonomy, complexity, and time to criticality. Time to criticality is important and changes with missions and functions, it is the amount of time to detect, recognize and react to a fault or potential failure before it becomes a failure, or if a failure occurs, then the time to put the system in to safe mode while correcting the problem. This determines the level and extent of autonomy of the fault detection, isolation and recovery activities. Both standards provide a clear acquirer –provider perspective as well as tailoring that meets the level of effort for the software class and criticality.

**NASA software is classified as safety critical if it meets at least one of the following:**

a. Causes or contributes to a system hazard/condition/event.
b. Provides control or mitigation for a system hazards/condition/event
   (1) Controls safety critical functions.
   (2) Mitigates damage if a hazard/condition/event occurs.
   (3) Detects, reports, and/or takes corrective action, if the system reaches a potentially hazardous state.
c. Processes safety critical commands (including autonomous commanding)
d. Resides on the same processor as safety critical software and is not logically separated from the safety critical software.
e. Processes data or analyzes trends that lead directly to safety decisions (e.g., determining when to turn power off to a wind tunnel to prevent system destruction).
f. Provides full or partial verification or validation of safety critical systems, including hardware or software subsystems. (e.g., this can include models and simulations )

**Table 2. NASA Software Safety Litmus Test**

With the basic software engineering and assurance requirements firmly established across NASA, it becomes a matter of training, implementation and improvement. NASA has a robust training program for software assurance. The NASA Safety Center maintains not only NASA created instructor and web-based training on all the sub-disciplines of software assurance, it also contracts with outside experts to bring in specialized training where needed. NASA's Software Engineering also has agency wide training that software assurance participates in as well as project level training for project specifics.

Organizationally, within NASA, the number of actual practitioners of software assurance assigned to the independent offices of Safety and Mission Assurance across the Agency may be relatively small. Thus, the requirements are intentionally written so that many different groups may perform different aspects of software assurance (e.g., systems engineering might perform the software safety analyses, software engineering might collect and trend defects). An entity/organization independent from the organization creating the software still is required to either perform or guarantee that software assurance activities are performed correctly and to the necessary level, and that records of those activities are created, analyzed, and maintained. Software Assurance metrics are also important. Software engineering and software assurance organizations share many software product quality metrics and process metrics, but NASA also requires software assurance performance metrics, to track and measure the performance of the software assurance activities and to improve activities for missions. Many software assurance activities may be tailored and performed within the project structure, but a group independent from the project evaluates those activities and the results. For NASA this is the Safety and Mission Assurance (SMA) organization; for a contractor, this should be a managerially separate safety and assurance organization which should be called out in the contract. Often, one or more software assurance engineers from an SMA organization may be assigned to work with a project throughout its life cycle. While these software assurance engineers are a part of the project and participate in day-to-day activities, perform most or all of the assurance functions, and attend project meetings and reviews, they maintain a separate reporting chain through their SMA organization. This activity is much like an oversight role, that is, the software assurance engineers are closely tied in with the project and provide input on a daily basis. At other times, the independent organization, SMA, may provide only insight for the project, evaluating if the software assurance activities are performed and performed sufficiently by

the project personnel and participating more by audits and at formal review intervals. In either case, there must be a close working association and joint reporting to both the project and the SMA organization.

NASA's Independent Verification and Validation (IV&V) is the third look at our most critical software. Engineering is responsible to build the software correctly and according to known good principles and thus is the first look. Software assurance works with the projects on a day to day bases, assessing the quality, safety, security and reliability of the processes and products and is the second look, with independent reporting chain up through the Center Safety and Mission Assurance Office and more closely associated with the total software processes and products. For NASA's most critical software, NASA's IV&V provides the third look, an objective examination of safety and mission critical software processes and products, delving into the analyses of the most critical aspects of the software on a project looking at safety, security and reliability. IV&V is considered to be technically, managerially and financially independent from the projects it works on. IV&V focuses on three perspectives:

- Will the system's software do what it is supposed to do?
- Will the system's software not do what it is not supposed to do?
- Will the system's software respond as expected under adverse conditions?

As a part of Software Assurance, IV&V plays a role in the overall NASA software risk mitigation strategy applied throughout the lifecycle, to improve the safety and quality of software systems.

Improvement of the software assurance program is achieved via four main paths and sundry smaller ways. First, there is a robust audit program that checks not only that the requirements are being followed in the field, but also brings the NASA Software Assurance Manager data to consider for systemic problems with the requirements implementation, training, and with the requirements themselves. Each of NASA's Centers and facilities, as well as some of the major programs are audited at least every 2-3 years from the Headquarters level. After each audit, the findings are discussed and compared to previous audits and center/facility results. Any repeat or evolving problem areas are then discussed with the Center/facility personnel for resolution and the data is used to see what additional training, guidance or even changes are needed in the requirements. At the Center or facility level, internal audits for each project are run more frequently according to the schedule and criticality of project development. In addition, Center Safety and Mission Assurance level technical authorities monitor all safety, reliability, quality, and software assurance on projects. Projects work with SMA to conduct internal audits and CMMI Level 3 assessments (or equivalent) are required for all NASA's Class A, and most of Class B, Software projects.

The second improvement path is the NASA Software Assurance Research Program (SARP). Software engineering development and analyses continues to evolve, in order to stay current with software changes and the environment in which software is developed and operated, NASA has a long standing research program, SARP, which yearly polls the software assurance and software engineering communities for areas of need in software assurance. Then a research call is sent out, mainly within NASA, to solicit proposals to address these issues. SARP seeks practical solutions, tools, guidance, and processes of value to the greater software assurance community. The projects can be from 1 to 3 years in length with a transition to practice as part of the work. The proposals are peer reviewed by the

community and selected according to need and meeting the SARP criteria for a good project. Some examples of SARP's output include a software (and system) hazard tracking system; guidance on better ways to collect, visualize, and present software assurance data; processes for performing Model Based testing of large systems; cost estimation methods for software assurance activities, and command reliability, to name a few. SARP usually has one or two projects that look to future software assurance needs, researching and posing potential solutions, and questions, to the ever evolving software development and operational landscape. Not every year are there sufficient funds to cast the research requests out to all industry and academia, but it is not insular, either. While limited, NASA's SARP program does seek out input from academia and beyond to keep current with new trends in software and computing systems.

Third, and most importantly, the NASA Software Assurance community is a close knit group that shares successes and failures, supporting one another and working together to create the assurance and safety standards, guides, and select needed training and research. Meeting, on average, twice a month via telecon and once a year in person, the NASA Software Assurance Manager, NASA Safety Center Software Assurance Lead and all the Center/facility software assurance leads and personnel stay current with Agency trends and needs. They review SARP work, present on their Center/Facility work, needs, and issues. Then they jointly formulate the NASA Software Assurance Objectives, Goals, Strategies and Metrics to create a road map to improve software assurance, laying out 1 to 5 year goals and strategies. This strong community is the heart of NASA software assurance.

The current NASA Software Assurance Objective: *Demonstrate Software Assurance's contribution to assuring safety and mission success across all of NASA programs/projects/facilities with software.*

- *Goal 1:* Strengthen and maintain software assurance core competencies at all NASA centers [Make sure we have the right skills to do the job]
- *Goal 2:* Establish a core set of SA performance measures for all Centers across the Agency [Measure if we are effective and if not, know where to fix it]
- *Goal 3:* Increase value and establish SA as a core *Engineering* discipline [Provide clear relationship between what SA does and the risks it mitigates. Focus on risks the project and stakeholders need most to resolve.]
- *Goal 4:* For all NASA Projects, obtain the appropriate level of SA funding necessary to meet the projects' software assurance requirements tailored to the Program/Project's risk posture [Assure sufficient software assurance on a project for the scope, criticality and classification of the software, which means knowing what it costs to perform SA.]

The NASA Software Assurance community, while close knit and supportive does look beyond the NASA problem field. Benchmarking of academia, industry and portions of DOD allows NASA to not only compare where we are with others, but infuse new ideas. SMA has benchmarked with the Navy in the past and learned and shared both problems and approaches to success. NASA software engineering and assurance have recently benchmarked with 18 organizations, five of them from industry. The main assurance activities reportedly performed by the benchmarked organizations can be seen in Figure 2, below. The numbers show how many of the 18 benchmarked organizations reported having software assurance involved with each activity listed. Note that two universities had no formal assurance role at all. Also note that the activities are not counted if performed by other roles. For example, engineers rather than assurance personnel are often assigned to software safety and reliability.

Although some activities may not be reflected in the numbers (topics missed in the discussion), they provide a starting point for examining software assurance as documented versus the actual practice of it. While NASA has much more focus on software safety and reliability within the assurance organization, as well as contributing to the acquisition process and verification and validation processes, those benchmarked against NASA usually saw those activities as falling within another group.

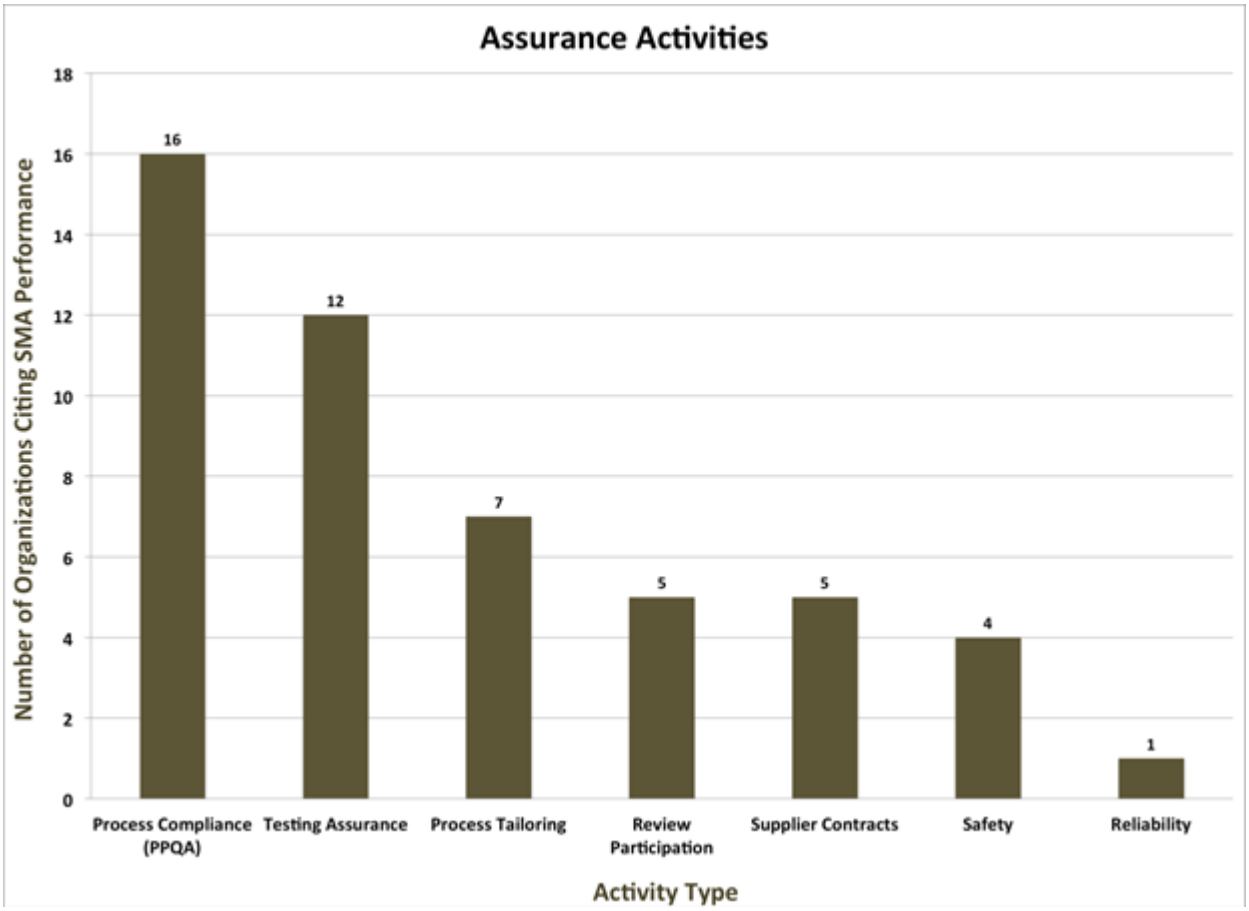


**Figure 2: Activities Performed by Software Assurance Organizations Benchmarked Outside of NASA**

The five Aerospace Industry organizations interviews can be summarized as follows:

- All industry organizations reviewed saw the main function of software assurance as performing process and product quality assurance (PPQA) and maintaining software compliance with institutional standards. Safety and reliability were seen as engineering roles.
- These organizations also tended to have a low ratio of software assurance engineers to the number of developers. For example, one organization had five to six software assurance engineers for about 200 developers. At the far end, an organization had only one "SQA person" for a 100-person software engineering project

- The high CMMI© (Capability Maturity Model Integrated) process maturity of most of these organizations might be a factor in their perceived need for assurance. All but one had been appraised at CMMI© Maturity Level 3 or higher and, as one noted, greater process maturity means more repeatable, institutionalized processes and fewer audit findings. The one organization that hadn't used the CMMI© also used one assurance person for a team of 15 developers and 4 testers – the highest ratio in the group.
- The industry organizations tended to use tools and metrics on the engineering side. Two of the organizations mentioned wide use of Six-Sigma, which also correlates with high CMMI© maturity.

Defense services organizations were interviewed, and their inputs on this topic can be summarized as follows:

- All the defense organizations had been appraised to some CMMI© level; two had achieved CMMI© ML 5 at some point, with one maintaining certification.
- Following a similar pattern to the industry organizations, all four organizations used software assurance primarily in a PPQA role and did not discuss their role in reliability or safety.
- Three out of the four organizations also used software assurance to witness or otherwise assure software testing.
- The one CMMI© ML5 organization maintained a process assurance group, dedicated to process compliance, and QA and IV&V groups for checking products.
- Of the three organizations that discussed Field Programmable Gate Arrays (FPGAs) or other Programmable Logic Devices (PLDs), none mentioned software assurance.

The following trends were identified, based on these interviews and compared to 5 of the 10 NASA centers:

- NASA Centers tended to involve software assurance in a greater range of development activities. Four of the five assurance organizations were involved with process tailoring, in addition to the PPQA audits.
- Four out of the five (not the same four) were witnessing or otherwise assuring that tests were performed properly.
- Four out of five NASA Centers also performed some assurance activity related to software safety.
- Three out of five of the NASA Centers used assurance personnel to monitor suppliers or software contracts in some way.

From this particular benchmarking, NASA software assurance has a broader scope, even if some of the Centers are not as involved as others in all the software activities that NASA describes as software assurance. This can be explained in part by the allowance of software safety and reliability to be performed by other organizations but also, not all NASA Centers work on Software Class A or B software. Many software projects at the NASA research Centers are assuring research and technology development projects.

In today's environment of cyber attacks, NASA has, in the past, considered this to be the realm of the Chief Information Office and the Protective Services Office. This may have worked for us in the past, but in today's world, NASA's Software Assurance has a role to play as well. In the DoD world, the term "software assurance" has almost become synonymous with cyber security and their increased focus in this area is understandable as the effort is large and only getting bigger. We are all vulnerable, and for NASA, our software resources, especially software assurance, are limited. The NASA software community (engineering, assurance, project management) is now joining our CIO colleagues in reaching out to the forums, training and working groups of DHS, DoD, NIST, and others to accelerate our efforts and share what we have learned with those who are also in this struggle. Still, NASA, like our colleagues, must continue to provide the quality, reliability and safety aspects of software that has kept NASA flying for many years and which supports elimination of vulnerabilities. While NASA SA is working more closely with the CIO office to better cover security oversight of Mission Software, it has not given up its strong dedication to safety, reliability, quality and Independent Verification and Validation, rather it has incorporated mission software security assurance into its repertoire.