Developing a Value-Based Methodology for Satisfying NASA Software Assurance Requirements

Daniel Port  
University of Hawaii  
dport@jpl.nasa.gov  

Joel M. Wilf  
Jet Propulsion Laboratory  
jwilf@jpl.nasa.gov  

Madeline Diep  
Fraunhofer Center Maryland  
MDiep@fc-md.umd.edu  

Carolyn Seaman  
University of Maryland  
cseaman@umbc.edu  

Martin Feather  
Jet Propulsion Laboratory  
martin.s.feather@jpl.nasa.gov  

Abstract

NASA imposes a multitude of quality process requirements on the development of its software systems. One source of such is the Software Quality Assurance standard. All NASA sponsored projects are expected to implement these requirements. However, given the diversity of projects and practices at different NASA centers, it is impossible to a-priori dictate how these requirements are to be economically satisfied on a given project. Under the auspices of NASA’s Software Assurance Research Program, the authors have been developing a value-based methodology to guide practitioners in defensibly and economically planning and executing assurance effort to satisfy this standard. The methodology exploits the intimate relationship between assurance value and risk-informed decision making. This paper describes this relationship, the value-based methodology for scaling assurance efforts, support for using the methodology, and our practice-based validation of the approach.

1. Introduction

The NASA Software Assurance Standard [4] provides a set of software assurance requirements for all mission software developed or acquired by NASA, in order to achieve the confidence in software quality, reliability, and safety needed for mission success. NASA Projects satisfy these requirements by performing software assurance activities. The requirements embody assurance goals for all of NASA, although not all requirements are applicable in all situations. For example, a project may be developing all software in-house, in which case subcontractor requirements would not apply. Still, the NASA Software Assurance (SA) Standard identifies a global set of requirements for all of NASA.

Software assurance activities, on the other hand, are tailored to the needs of individual projects, and vary according to the criticality, size, and complexity of the software being developed, and the budget and risk posture of the project itself. The challenge is to modulate the assurance activities to satisfy the requirements in “the best way possible” – for the individual project. The goal is to have an approach that enables application of the NASA SA Standard in a way that maximizes the value of assurance for that project given project goals and constraints.

Such standards are not unique to NASA or assurance in general. Indeed, there are a wide variety of software assurance standards such as IEEE Std 730-2014 [8]. More generally, there are important software standards addressing safety, security, reliability, and quality of development process such as ISO 9001 [6]. As such, the value-based approach described here should be applicable to other standards compliance contexts.

This paper describes the experience of the authors in developing and validating this methodology, the plan for its use and possible extensions and other applications. A notable contribution of this work is the finding that value-driven satisfaction of standards is deeply bound to risk-driven decision-making. It is this relationship that forms the basis of the methodology described herein.

We begin with a description of the NASA Software Assurance Standard and a discussion on the need for a value-based approach to satisfying this standard. Then some background on what software assurance is, its role in decision-making, and the software assurance value proposition that facilitates a SA value-model to serve as the primary basis for the methodology being developed. Subsequently the methodology is described followed by example applications and tool support. Lastly, validation and evolution efforts and possible future extensions and applications are discussed.
2. The NASA SA Standard

The NASA Software Assurance Standard is developed and maintained by the NASA Office of Safety and Mission Assurance (OSMA) “… to provide the requirements for ensuring consistent software assurance (SA) practices across all NASA centers, programs, projects, and facilities.” This standard is expected “… to be applied to all software developed by, or for, NASA and to the incorporation of open source, reused, commercial off-the-shelf (COTS), Government off-the-shelf (GOTS), or modified off-the-shelf (MOTS) software in the NASA system.” The standard “… applies to new contracts and subcontracts for developing software for use in NASA systems and should be referenced therein.”

The standard is quite broad and covers all phases and aspects of the software development lifecycle (Requirements, Design, Implementation, Test, etc.), development process (planning, measurement, personnel roles, etc.), software quality issues (safety, reliability, security), and assurance disciplines and activities (V&V, IV&V, Quality Control, etc.). Below are examples of requirements in the standard and discussion of issues in implementing them relevant to tailoring.

Some requirements are straightforward mandates of SA oversight on a project:

6.7.4.5 SA shall assure that after the systems integration and test phase, the project’s resolution of any software related defect is acceptable, per SAS-049. (SAS-063)

But who is appropriate to assess if defect resolution is acceptable? Are all defects assured or a only a sample? How far after is acceptable?

Other requirements imply activities to be performed by SA staff:

6.7.1.1 SA shall analyze the project’s documented software requirements to verify that the requirements are, at a minimum, well formed, complete, consistent, attainable, individually verifiable and traceable to a higher level requirement. (SAS-053)

What criteria determine a “well formed, complete, consistent, attainable, individually verifiable” requirement? What determines a valid trace or mistaken trace to a higher level requirement? If there are a large number of requirements, this effort can be huge. Is the assurance budget accounting for this?

There are independent check requirements:

6.7.2.2 SA shall assure that the project’s software systems’ architecture satisfies design readiness criteria for that project, and also check for proper inclusion of safety and reliability mitigations and controls, as well as completeness. (SAS-055)

Determining “proper inclusion” and “completeness” is highly subjective and prone to errors of omission and bias. This is why this requirement exists and implies an independent check from assurance staff. However to remain independent assurance staff do not directly participate in development and as such may not have the background or expertise in the given project area or architecture. “Coming up to speed” to provide creditable assessments can be expensive and viewed as unnecessary, ineffective, or duplicate effort.

There are requirements that indicate specific activities to be performed by SA staff:

6.7.5.1 SA shall verify software deliveries by conducting or participating in the execution of Functional Configuration Audit (FCA) and Physical Configuration Audit (PCA) activities. (SAS-065)

Projects may object to paying for audits where there is little doubt about the quality or if the project has a relatively small amount of software or perhaps has a high amount of re-use, COTS, or sub-contractor developed code.

Yet some requirements do not specify an assurance activity, rather an expectation as to how assurance is performed or a management function:

6.2.9 SA shall establish and maintain a schedule of SA plan activities that aligns with the project plans/schedules. (SAS-028)

For these kinds of requirements it is difficult to see how to directly plan to satisfy them. At the end of the project either this happened or it did not. It is unclear what, if any, tailoring could be done here.

For the past year the standard has been undergoing a major revision to enable it to be more easily and clearly adopted. We have taken an action research [10] approach to development of the methodology by working closely with the standards revision committee to help them converge on requirements that are “tolerable” and have clear purpose. In return for this we better understand what is needed for a practical to use value-based method for implementing this standard. As such the examples and methodology described herein are only applicable to the revised
standard that at the time of this writing has not been officially released.

3. Need for Value-Based Approach

The requirements given previously are just a small sample. There are approximately 75 software assurance requirements in all, and given the large amount and diversity of software developed for NASA, the scope of the standard is astronomical (pun intended). But it is impossible to specify requirements that universally and eternally apply to all projects.

Indeed one of the frustrating roadblocks projects have had in applying the standard is in how to address mismatches or misalignments with the project, non-applicable requirements, and the appropriate scaling of activities to satisfy the requirements. The dominant complication in this is that NASA requires all software to be classified in terms of criticality (Class A: Human Rated Space Systems through Class H: General Purpose Desktop Software) [9].

Each software class has significantly different assurance expectations. This varies not only in what is required to assure, but also in the level to which it is assured. Furthermore it’s not simply cumulative according to class. That is, the assurance requirements for a given class are not necessarily a subset of the class above it.

So the approach is to provide a comprehensive set that can address most situations. To this end the standard now specifies a provision for adaptation:

The requirements provided in this Standard may be tailored, as necessary, to develop project SA plans commensurate with specific software classification, criticality, objectives and risk postures. For additional information on requirements tailoring, see Section 6.6 Waivers, Deviations and Tailoring.

But how does a project go about tailoring? What does it mean to “tailor the standard” and what exactly is tailorable and to what extent? To answer such questions and provide a practical means for performing a tailoring that is fitting to NASA (i.e. as indicated in the standard in Section 6.6) is the primary purpose of the methodology under development.

Where does value come in? By far the most common issue that arises in the use of standards is in choosing a scaling (sometimes misleadingly referred to as “tailoring”) of activities, in both approach and amount, to defensibly and economically satisfy them.

Fundamentally NASA assurance practitioners are continually challenged with trade-off decisions between the assurance needs of the project, addressing NASA assurance requirements, and tightly limited budgets. They need practical assistance (via tool support, tailoring guidelines, and training sessions) to accurately and defensibly scale and plan for assurance activities needed to both satisfy the NASA SA standard and establish confidence in the quality of the software for risk-managed project decision making. Current approaches are primarily ad-hoc or experience driven. This more often than not leaves stakeholders with uneasiness about how well the standards have been addressed and confusion on what standards effort is actually needed and useful (i.e. of “value” versus perfunctory satisfaction of requirements).

Therefore a key objective of this work is to develop a methodology that can engage SA practitioners and project stakeholders in a constructive dialog about cost and value trade-offs for implementing the NASA SA standard. To this end we must have a clear understanding of what assurance is, its expected benefits and costs, and its intimate relationship with decision making.

4. SA Value and Decision Making

The official NASA definition considers SA an umbrella for numerous disciplines such as Safety, Reliability, and IV&V as indicated in Figure 1. Their definition of SA is [5]:

The planned and systematic set of activities that ensure that software life cycle processes and products conform to requirements, standards, and procedures.

![Figure 1: The NASA SA Umbrella](image)

This is elaborated in the SA Standard:

The purpose of SA is to provide an independent look at the software engineering processes and products to assure that software quality and safety are properly implemented and are commensurate with the project software class and level of risk. The SA process systematically ensures the software life-cycle products and processes conform to project requirements, standards and procedures; are consistent, complete, correct, safe, secure and reliable within the operating environment; and satisfy customer needs. SA also checks that all of the processes used to acquire, develop, verify, validate, operate, maintain and retire the software are appropriate, and planned, reviewed,
and properly implemented, meeting any required standards, regulations, and quality requirements. SA utilizes project-based measurement data to monitor products and processes for possible improvements. SA assists in risk mitigation by minimizing defects early in the software life-cycle and, through its activities, enables improvement of future products and services.

However currently there are a wide range of viewpoints on the nature and practice of SA that differ from NASA’s official view, even between and within NASA centers and aerospace organizations that contact with NASA. These viewpoints range from something as simple as “SA is extended testing” to activities performed by a fully independent verification and validation organization. With such differing views it can be confusing as to what activities are within the scope of SA practice, how and to which degree they are to be performed, and what results are expected. This complicates identifying what value is expected from these activities and more generally, the value of software assurance itself. To illustrate this challenge we present a few of these different perspectives that can found in various official NASA documents.

Consider for example, the difference between the NASA definition of Software Assurance, the definition in the JPL SQA Charter (a form of institutional standards), and another definition taken from the ICCA (Space Station) Mission Assurance Plan:

**JPL charter:**

The Software Quality Assurance (SQA) Group promotes awareness of and reduces exposure to software-related risk for JPL missions.

**ICCA Mission Assurance Plan:**

The purpose of SQA is to achieve the highest quality-to-cost ratio within the Project’s constraints and policies and to increase the probability of overall mission success.

It is notable that the latter definition, unlike the NASA definition, indicates two value measures. Generally speaking, though, such definitions do not connect the implied or stated activities to value objectives even when these are stated explicitly. However the different official definitions have shaped a few common goals for SA:

- To lower software risk
- Ensure compliance to institutional standards and requirements
- Ensure compliance to standards and processes adopted by project

Less officially, our interviews with stakeholders reveal the following objectives for SA:

1. **During operations:** fewer missions failing due to software quality problems
2. **During development:** confidence that code and other work products have an appropriate level of quality throughout the life cycle, especially at “gateway” decision points such as acceptance reviews
3. **Support for decision-making:** reduce risk of making a “bad” decision.

An observation from our investigation of assurance that differentiates it from other quality management efforts is that SA is never expected to directly improve the quality of the product or process. For example there are no SA activities that involve modifying code. Indeed all activities focus on artifacts (e.g. code, documentation, test results, problem reports, etc.) produced or will be produced by the project or processes the project used or intends to use. This is a key point for our SA value-model that serves as the basis for the methodology.

The issue is, whatever the objectives and expectations are, what specific role does SA play in achieving these? There are many options for managing quality other than SA such as automated testing and CMMI certified development processes. One can evaluate the costs and benefits of different investments in managing quality, then based on economic arguments, a decision is made on where how to manage quality. The required level of quality for a particular project is not an inherent attribute of the system, but rather it is a business decision.

To make this decision, management needs to determine how much to invest in quality and what benefits they can expect. Management also needs to compare alternative investment strategies to determine which focus on. This indicates the need for operable definition of assurance and the value of what assurance can do to manage quality (a value proposition). Our investigations yielded the following potentially useful operational definitions:

**Operational definition of SA:**

Software Assurance is the planned and systematic set of activities that provides justified confidence that the system is of expected quality.

**SA Value Proposition:**

Software Assurance enables more confident decision-making by providing independent credentialed information to reduce uncertainty in systems decisions that depend on quality – and thus reduce decision risk.
Figure 2 illustrates an operational model of an assurance activity derived based on the above definition and the value proposition:

![SA Activity Model](image)

**Figure 2: SA Activity Model**

To explain the model, consider the following example of reviewing a test case to requirements tractability matrix. The model indicates that:

Software assurance performs a review on a test case to requirements trace matrix against relation/no-relation criteria in order to decide if testing was sufficient to decide if the system is ready for release.

The ready for release decision depends greatly on having sufficient confidence in the assessment of the system quality (e.g. expected functionally, non-functional attributes, defects, etc.). Note that we are being careful to not assume that the quality is good. The decision to release depends on having sufficient confidence that the quality is good enough. This is not simply a binary decision. To explain this, consider if we are confident that the quality is not good enough. Then clearly the decision is not to release. Conversely, if we are confident the quality is good enough, release. But what if there is insufficient confidence that the quality is good enough? That is, the uncertainty about the quality ranges from good enough to not good enough. There is a risk of making a bad decision simply because of this uncertainty. That is, release and you find out later that the actual quality was in the not good enough range; or do not release and you find out later that it was actually in the good enough range. Either way there is an opportunity loss for making a decision under the uncertainty of the quality.

This is the heart of assurance. It enables decisions to be made more confidently by accounting for this kind of uncertainty. If the uncertainty is too high, more assurance can be performed to reduce it to the point that a confident enough decision can be made. Note that this does not necessarily entail improving the quality. It only has to address the degree of doubt there is about the qualities that the decision depends on. If there is no uncertainty about the qualities, then there is no value of assurance activity because the optimal decision would be straightforward. But quality is a factor that is always uncertain. Even in the most controlled circumstances, it is either impossible or impractical to eliminate all uncertainty about quality (e.g. “complete” testing, “proving” via formal modeling, etc.).

The Operational Definition, Value Proposition, and Activity Model constitute our SA Value Model that forms the foundation for the methodology which is discussed in the next section.

5. Value-Based Tailoring of Assurance

The value model provides a description of the relationship between assurance activities, standards, and project decision-making. This provides a pathway for tailoring of assurance activities as indicated in Figure 3.

![Value Based Tailoring of Assurance](image)

**Figure 3: Value Based Tailoring of Assurance**

Driving this is the value proposition, which connects the assurance activity to the reduction of decision risk. The approach is to (1) begin with an assurance requirement, (2) consider how much risk there is in making the decision it relates to, (3) assess the amount of uncertainty there is in the factors needed for making the decision, then (4) modulate the SA activity to reduce the uncertainty to the point where a confident enough decision can be made. This is all implicitly relative to the cost for performing the activity. If the cost for performing the activity exceeds the potential loss from making a bad decision (because of the uncertainty in the decision factors only), then the activity needs to be scaled back or the assurance requirement should be waived. If the loss potential from a bad decision is very high, the assurance activity needs to be scaled up.

The tailoring approach is expected to be performed primarily at the initial planning and costing of assurance effort. The plan can be re-evaluated and adjusted as needed to accommodate project re-scopes.
and unforeseen changes and events. The main benefit here is that there is a defensible basis and visible purpose for performing SA activities and adhering to an appropriate scaling of the assurance standard.

To make the approach actionable, the standard will provide a “Requirement-Activity” mapping matrix for planning SA activities that is both cost- and value-conscious. This mapping is structured and developed directly from the SA value-model. The Requirement-Activity mapping contains all the SA requirements that have been imposed on projects as defined standard. Below is an excerpt of the Requirements-Activity mapping matrix:

<table>
<thead>
<tr>
<th>SectNum</th>
<th>Requirement Note</th>
<th>Activity Name</th>
<th>Decision Supported</th>
<th>Inputs</th>
<th>Outputs</th>
<th>Uncertain decision factors</th>
<th>Impact of wrong decision (decision risk)</th>
<th>Tailoring</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.2.15</td>
<td>SA shall verify the project's software requirements traceability as defined in NPR 7150.2. (SAS-034)</td>
<td>Verify requirements traceability</td>
<td>Review the requirements traceability matrix</td>
<td>Whether to proceed to the next life cycle phase (e.g., this is part of the readiness criteria for design, coding, test, and delivery); decision points in subsequent life cycle phases that depend on correct and complete requirements tracing (e.g., a design decision).</td>
<td>Requirements, the architecture, design, code, test plan, or delivery package; and project-created trace matrix</td>
<td>Assured trace matrix</td>
<td>Mistakenly proceeding when requirements are not fully realized (in design, code, tests, or delivery); inappropriate architecture, design or coding choices; insufficient testing; inability to efficiently or correctly respond to requirements changes; unnecessary implementation of orphaned requirements. All of these can potentially lead to significant cost, schedule and/or mission risks.</td>
<td>1) H: SA performs review of compliance to NPR 7150.2, and in addition review of traceability is performed by SA independently</td>
</tr>
</tbody>
</table>

Table 1: Example Requirement-Activity Mapping

The Requirement-Activity matrix contains the above kinds of information for each requirement that appears in the SA Standard. For each of these requirements, we have the following fields:

- SectNum – The section numbers in the SA Standard document from which this requirement comes.
- Requirement – The requirement exactly as it appears in the SA Standard.
- Requirement Note – Any additional information about the requirement.
- Activity name – The name of the SA activity that is normally carried out to satisfy this requirement.
- Activity – brief description of the activity.
- Decision supported – a decision that is made concerning the project that is made easier (less uncertain) when informed by information resulting from the activity. Note: in many cases there are multiple decisions listed here, usually occurring in different parts of the process.
- Inputs – the inputs to the SA activity; may include artifacts or documents, or may be more informal outputs from other activities.
- Outputs – the outputs of the SA activity. Note: for activities that lead to assuring compliance or concurrence with some development artifact, this column should include outputs that result from a determination of compliance (or concurrence) as well as those outputs that result from a determination of non-compliance (or non-concurrence).
- Uncertain decision factors – factors that are important to know about when making the decision described in the “decision supported”
column, but that about which there is often uncertainty, which in turn makes the decision difficult to make. The activity in this row is intended to produce information about these factors and thus reduce the uncertainty related to the decision.

- **Impact of wrong decision (decision risk)** – the negative consequences of incorrectly making the decision in the “decision supported” column because there is inadequate information about the “uncertain decision factors”. These negative consequences, then, are the impacts of not doing, or inadequately doing, the SA activity.

- **Tailoring options** – a description of some variations on the activity, i.e. different ways that it could acceptably be done.

- **Effort (basis of estimate)** – the unit of analysis upon which an effort estimation could be done for this activity. This should be something countable that is thought to have a relatively linear relationship with effort. There might be several potential bases included in this column.

- **Downstream impact (optional)** – elaboration of the “impact of wrong decision” column to include longer-term or higher-level impacts of not adequately meeting the requirement in this row.

- **Notes** – any additional information not covered in the above

To explain the example in Table 1 we consider a software assurance practitioner planning the assurance effort for a project. The standard section 6.2.15 states that there must be an independent check against the NPR 7150.2 criteria of any requirements traceability artifacts. The activity is ostensibly a review of a tractability matrix. The decision this activity helps build confidence for is a readiness for next phase (possibly release). A readiness decision depends greatly on having a complete and correct tracing. High uncertainty about the completeness in a tracing from requirements to testing would leave doubt about the sufficiency of testing performed leaving the project at risk of making a bad decision on readiness for release (i.e. releasing with insufficient testing or holding up a sufficiently tested system). The trace matrix review will help reduce this uncertainty so a more confident decision can be made (reduces decision risk). Note how this follows the SA Activity Model as given in Figure 2: SA Activity Model With the explicit connection of the activity (review requirements traceability) to the decision the requirement is meant to mitigate risk for (in this case “readiness for next phase”), assurance staff is able to assess how much uncertainty there is in the decision factors for this activity (e.g. coverage, completeness, etc.) and how much uncertainty needs to be reduced (or can be reduced) to arrive at a confident decision (i.e. tolerable decision risk).

The tailoring process for using the Requirement-Activity matrix is summarized as follows:

**Entry Criteria**
Prior to performing this planning activity:
1. Software classification has been performed and concurred
2. An acceptable budget range that the project is willing to allocate for SA activities has been established

**Tailoring Steps**
1) For each of the requirements in the SA Standard determine if:
   a) The requirement is applicable to the project and will be implemented (Go to Step 2). Note:
      i) Refer to the Requirements Applicability matrix (which defines what requirements are imposed on a project based on the mission software classification)
      ii) Additionally, a requirement may not be applicable to the project if it is related to a practice that will not be followed/used in the project (e.g., the use of other generated software, simulation tool, the existence of a subcontract, etc.)
   b) The requirement will be waived (assess and include unmitigated decision risk accepted into overall project risk profile), or
   c) The requirement is not applicable (provide justification, do not include decision risk assessment)
2) Choose how the SA activity associated with the requirement will be performed from the options available (possibly extended by the addition of new options). This includes the following steps:
   a) Identify the risk associated with the decision activity supported by the output of the SA activity performed.
      i) Review and assess the presence and degree of the uncertainties that may impact the decision made (with respect to uncertainties listed in decision factors listed). This is the unmitigated “Uncertainty”.
      ii) Review and assess the impact if decision is made incorrectly because of uncertainty in the decision factors. This will give you the unmitigated “Impact”
iii) A combination of the “Uncertainty” and “Impact” is the decision risk profile related to this requirement.

b) Select or add a tailoring option that reduces the uncertainty to a level of acceptable decision risk. Note:
   i) If the uncertainty is “High”, the activity tailoring that provides the highest uncertainty mitigation will provide the largest reduction in decision risk.
   ii) If the uncertainty is “Low”, any activity tailoring that provides “Small”, “Medium”, or “High” risk uncertainty mitigation will provide sufficient risk reduction so any of the activity tailoring may be selected.
   iii) Custom tailored activities not listed need to be assessed as to how much uncertainty will be reduced by performing this activity.

c) Identify the cost associated for the selected tailoring activity (historical, parametric, or expert judgment). Indicate uncertainty in estimate.

3) Once an activity has been selected and tailored for each requirement, review the total budget for the selected activities. If the total budget is higher than what is available:
   a) Revisit the requirements, prioritizing on the higher cost activities. If other tailoring options that have lower cost are available, consider if their impact on risk mitigation is acceptable to the project.
   b) Iterate on Step 3 as needed to reduce to available budget. If this cannot be done, renegotiate assurance budget or perform activities in order of highest risk-reduction to cost ratio.

Exit Criteria:
1. Overall resulting risk profile expected after performing the planned SA activities is acceptable to project and addresses all applicable assurance standard requirements.

In the above process the term “applicable” may seem unclear. To put this process into context, the SA Standard defines which of its requirements are imposed (mandated) for a given mission software classification. Software is, and it is from these that decisions are made whether they are applicable, whether they will be waived, or whether they will be implemented - in the last case, leading to choice of activity.

6. Tool Support

Given the number of requirements and complexity of the SA standard, the tailoring process can be challenging. We have developed a prototype tool to help projects and software assurance planners decide how to implement the requirements in the standard and record their decisions together with (when necessary) rationale for those decisions.

Essentially the tool is a friendly front end to the underlying Requirements-Activity mapping matrix, and guides the user through the tailoring process. In particular the tool:

- Shows the overall status of the decision process for the entire set of Software Assurance requirements implementation choices
- Allows the users to step through the requirements to study the detailed information one requirement at a time
- Allows the users to input their estimates of the factors that determine the project risk were that requirement not implemented, or insufficiently implemented
- Allows the users to scrutinize each of the implementation choices to see their costs and benefits (the latter in terms of risk reduction)
- Allows the users to add new implementation options should they wish to do so
- Allows the users to keep notes on the decisions they have taken
- Keeps track of their decisions, additions and notes in the underlying spreadsheet, thus serving as a record that can be retained as a straightforward Excel file.

The tool assists project and software assurance personnel to make use of the information in the Requirements-Activity mapping matrix (which is expected to be provided in the guidelines for use of the standard) as they negotiate and decide the intensity (breadth and depth) of software assurance that the project will fund.

This software - currently in PROTOTYPE form - focuses on assisting in the choice of how to implement the NASA Software Assurance standard, it has a specific area of application that would apply across NASA. More generally, the same overall approach - and the same or very similar software - could presumably apply to choices of how to implement other kinds of assurance requirements, not just software (e.g. hardware, mission, safety, security, etc.). Also the same approach would presumably apply to similar situations outside of NASA.

The advantage of using this tool for SA requirements tailoring is that there is no existing
software customized to this decision process. Without this software, the same information would have to be accessed by examination of the (somewhat unwieldy) Requirements-Activity mapping matrix, and manually recording decisions and rationale, but this would be tedious and lack the cogent presentation and interactive visualization and summary information that this software provides. Figure 4 presents a screenshot of the prototype tool and explanation of its elements.

The prototype is implemented as macros within a Microsoft Excel spreadsheet, and thus will run on any PC that has a reasonably current version of Excel. The functionality should also work on Excel on Macs, but this has not been extensively tested.

7. Applications and Integrations

The tailoring methodology will be included in the revised SA standard guidelines (probably in Section 6.6, as indicated previously). When the revised standard is released, all NASA centers that develop software will be required to perform tailoring according to these guidelines. Major NASA software producing centers include JPL, Goddard, Johnson Space Center, Ames, and Glenn Research Center. Although the methodology will be provided in the revised standard and made publicly available, the tool support may not. Furthermore the tailoring process is ideally performed during the planning and budgeting of assurance (i.e. developing the assurance plan) when cost, needs, and benefit trade-offs are made.

Several NASA centers have already adopted an cost model for their assurance planning needs. Essentially it is a work breakdown (WBS) of assurance areas and the activities that can be performed in these areas. The tailoring methodology has a fairly straightforward integration with this model by extending it to begin with the requirements in the standard, choose the activities according to the SA Value model, then cost these activities as per the WBS, and iterate until the appropriate cost and risk profile is achieved. The integrated cost-value model not only helps deliver a cast and value driven assurance plan, it also ensures that the SA assurance standard is satisfied as mandated by NASA.

Closely related to our project, there is another assurance action research project “SAPE” to develop a methodology and tool for the planning and management of assurance effort. The goal for the project is to provide an across center tool to aid assurance staff in the costing and reporting of assurance effort. This is another natural opportunity for application and integration with our methodology and we have begun working with their team to explore this.

8. Evolution and Validation

How do we know that the methodology is practical, useful, and beneficial for performing tailoring to satisfy the standard? As an action research project we work closely with assurance practitioners and NASA assurance management to ensure we are addressing their needs and their problems. In addition to studying assurance practice, we participate in the revision of the standard and perform extensive surveys and interviews on how the standards are currently addressed by projects. We include a diversity of stakeholders outside assurance practitioners such as mission assurance managers, project software system engineers, project managers, IV&V staff, and NASA Office of Safety and Mission Assurance managers. We are currently performing detailed run-throughs of the methodology to identify gaps, errors, and difficulties in application.

The data and experience from the above is used to evolve the methodology and ensure that it is practical to use and is useful for the value-driven satisfaction of the SA standard. Comprehensive validation was performed through interviews with four software assurance engineers from JPL, AMES, and the NASA IV&V center, four Mission Assurance Managers, and full run-through of the methodology with a software assurance group supervisor at JPL who would be a primary user of this work. A final validation with the NASA Quality Assurance Requirements auditor is scheduled before conclusion of the project.

We have found that our models are consistent with our study of SA practice, the revised standard, and interviews with SA practitioners and stakeholders. Furthermore we have found that our validation efforts have led to an increased clarity on the role of software assurance and of the NASA SA standard and have been an excellent initial dissemination of the methodology.

9. Summary

The Office of Safety and Mission Assurance is revising their SA standard that all software developed at NASA must satisfy. Due to differing expectations for assurance from different software classifications, variations in assurance across NASA centers, and tight competition for project resources, projects must adapt their assurance efforts to maximize value which still satisfying the standard. Through extensive action research, we have developed an assurance value model that serves to structure and guide this can be done. Using the value model we have created a detailed Requirements-Activity matrix that guides assurance staff to have a constructive cost and value dialog with project stakeholders when planning assurance effort and satisfy the standard. This and the methodology in
general will be provided to NASA centers with the forthcoming revision of the SA standard. A prototype tool has been developed to support implementing the methodology. There are several possible integrations of the methodology with existing assurance cost models and a forthcoming assurance planning and management tool. We continue to evolve and validate the methodology through surveys, interviews, run-throughs, and ultimately a pilot study.

10. Acknowledgement
This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology under a contract with the National Aeronautics and Space Administration, and at the Fraunhofer Center in Maryland, and the University of Hawaii Shidler College of Business.

The work was sponsored by the NASA Office of Safety and Mission Assurance under the Software Assurance Research Program led by the NASA Software IV&V Facility. This activity is managed locally at JPL through the Assurance and Technology Program Office.

The authors thank Tuan Do, supervisor of the JPL Software Quality Assurance group, for his discussions with us on the topics of this paper.

11. References

A PROTOTYPE of an Excel-based tool to help software assurance together with a project decide how to implement the NASA Software Assurance requirements

Bird’s eye view” of overall status
Unmitigated risk (if the requirement is not implemented)
Mitigated risk (as mitigated by the chosen implementation option)
The users can add new implementation options
The users can record notes on each requirement
Uncertainty and Impact are assessed for the project in question
Uncertainty: the approach here is that assurance decreases uncertainty
Sets of information relevant to choosing the implementation
Currently selected requirement
Estimated hours for implementation options
Implementation options for the selected requirement
Currently selected implementation option
The text of the selected requirement
Not chosen: the choice of how to implement the requirement is yet to be made
Chosen: the choice of how to implement the requirement has been made
N/A: the requirement has been deemed Not Applicable
Waived: the requirement is being waived.

Figure 4: Screenshot of Tailoring Tool