Proposed Appendix to NPG 2820

Independent Verification and Validation (IV&V) Criteria

1. The purpose of this appendix is to establish quantifiable criteria for determining whether IV&V should be applied to a given software development. Since IV&V should begin in the Formulation Subprocess (as defined in NPG 7120.5, Section 1.4.3) of a project, the process here described is based on metrics which are available before project approval.

   These criteria shall be applied to NPG 7120.5 "projects" as defined in the NPG. Software developments outside the scope of NPG 7120.5 are determined to be within scope of this appendix on a case by case basis. That decision will be made by the NASA Chief Information Officer (CIO), the NASA Office of the Chief Engineer (OCE), and the NASA Office of Safety and Mission Assurance (Code Q) or Center Safety and Mission Assurance.

   Projects meeting the following criteria are not subject to this appendix, and need not be addressed further:

   a. The software product is only used for post mission scientific data analysis

   b. Consequences of software failure (Not to exceed any of the following)

      - Potential for loss of life - No
      - Potential for serious injury – No
      - Potential for catastrophic mission failure – No
      - Potential for partial mission failure – No
      - Potential for loss of equipment – Less than $2,000,000
      - Potential for waste of resource investment – Less than 20 work-years on software
      - Potential for adverse visibility – No more than local visibility
      - Potential effect on routine operations – No more than a Center inconvenience

2. IV&V is intended to assist mitigating risk; hence, the decision to do IV&V should be risk based. NPG 7120.5 defines risk as the "combination of 1) the probability (qualitative or quantitative) that a program or project will experience an undesired event such as cost overrun, schedule slippage, safety mishap, or failure to achieve a needed breakthrough; and 2) the consequences, impact, or severity of the undesired event were it to occur." The exact probability of occurrence and consequences of a given software failure cannot be calculated early in the software lifecycle.

However, there are realistically available metrics which give good general approximations of the consequences as well as the likelihood of failures.

2.1 In general, the consequences of a software failure can be derived from the purpose of the software: i.e., what does the software control; what do we depend on it to do. Section 2.1.1 contains a list of factors, which can be used to categorize software based on its intended function as well as the level of effort expended to produce the software. Section 2.1.2 defines the boundaries of four levels of failure consequences based on the rating factors from 2.1.1.

2.1.1 Factors contributing to the consequences of software failure:

2.1.1.1 Potential for loss of life. Is the software the primary means of controlling or monitoring systems that have the potential to cause the death of an operator, crewmember, support personnel, or bystander? The presence of manual overrides and failsafe devices are not to be considered. This is considered a binary rating: responses must be either yes or no. Examples of software with the potential for loss of life include:

- Flight and launch control software for manned missions
- Software controlling life support functions
- Software controlling hazardous materials with the potential for exposure to humans in a lethal dose
- Software controlling mechanical equipment (including vehicles) which could cause death through impact, crushing, or cutting
- Any software which provides information to operators where an inaccuracy or misinterpretation of the data could result in death through an incorrect decision (e.g., mission control room displays)

2.1.1.2 Potential for serious injury. Serious injury is here defined as loss of digit, limb, or sight in one or both eyes, sudden loss of hearing, or exposure to substance or radiation that could result in long term illness. This rating is also binary. This rating considers only those cases where the software is the primary mechanism for controlling or monitoring the system. The presence of manual overrides and failsafe devices are not to be considered. Examples of software with potential for serious injury include software controlling milling or cutting equipment, class IV lasers, or X-ray equipment.

2.1.1.3 Potential for catastrophic mission failure. Can a problem in the software result in a catastrophic failure of the mission? This is a binary rating. Software controlling navigation, communications, or other critical systems whose failure would result in loss of vehicle or total inability to meet mission objectives would fall into this category.

2.1.1.4 Potential for partial mission failure. Can a problem in the software result in a failure to meet some of the overall mission objectives? This is a binary rating. Examples of this category include software controlling one of several data collection systems or software supporting a given experiment, which is not the primary purpose of the mission.

2.1.1.5 Potential for loss of equipment. This is a measure of the cost (in dollars) of physical resources that are placed at risk due to a software failure. Potential collateral damage is to be included. This is exclusive of mission failure. Examples include:

- Loss of a $5 million unmanned drone due to flight control software failure. (Assuming the drone is replaceable, this wouldn't be a mission failure)
- Damage to a wind tunnel drive shaft due to a sudden change in rotation speed.

2.1.1.6 Potential for waste of software resource investment. This is a measure or projection of the effort (in work-years, civil service, contractor, etc.) invested in the software. This shows the level of effort that could potentially be wasted if the software doesn't meet requirements.

2.1.1.7 Potential for adverse visibility. This is a measure of the potential for negative political and public image impacts stemming from a failure of the system as a result of software failure. The unit of measure is the geographical or political level at which the failure will be common knowledge—specifically: local (Center), Agency, national, international. The potential for adverse visibility is evaluated based on the history of similar efforts.

2.1.1.8 Potential effect on routine operations. This is a measure of the potential to interrupt business. There are two major components of this rating factor: scope and impact. Scope refers to who is affected. The choices are Center and Agency. The choices for impact are inconvenience and work stoppage. Examples:

- A faulty firewall which failed to protect against a virus resulting in a 4-hour loss of e-mail capabilities at Goddard would be a "Center inconvenience".

- Assuming that the old financial management software was no longer maintainable, the failure of the replacement system to pass acceptance testing and the resulting 2-year delay would be a potential "Agency work stoppage." This doesn't imply that workarounds couldn't be implemented, but only that it has the potential to stop work Agencywide.

2.1.2 Software Consequences of Failure Rating

2.1.2.1 Consequences of failure are considered "Grave" when *any* of the following conditions are met:

- Potential for loss of life - Yes
- Potential for loss of equipment – Greater than $100,000,000
- Potential for waste of resource investment – Greater than 200 work-years on software
- Potential for adverse visibility - International

2.1.2.2 Consequences of failure are considered "Substantial" when *any* of the following conditions are met:

-
- Potential for serious injury – Yes
- Potential for catastrophic mission failure – Yes
- Potential for loss of equipment – Greater than $20,000,000
- Potential for waste of resource investment – Greater than 100 work-years on software
- Potential for adverse visibility - National
- Potential effect on routine operations – Agency work stoppage

2.1.2.3 Consequences of failure are considered "Marginal" when *any* of the following conditions are met:

- Potential for partial mission failure - Yes
- Potential for loss of equipment – Greater than $2,000,000
- Potential for waste of resource investment – Greater than 20 work-years on software
- Potential for adverse visibility - Agency
- Potential effect on routine operations – Center work stoppage or Agency inconvenience

2.1.2.4 Consequences of failure are considered "Insignificant" when *all* of the following conditions are met:

- Potential for loss of life - No
- Potential for serious injury – No
- Potential for catastrophic mission failure – No
- Potential for partial mission failure – No
- Potential for loss of equipment – Less than $2,000,000

- Potential for waste of resource investment – Less than 20 work-years on software
- Potential for adverse visibility – No more than local visibility
- Potential effect on routine operations – No more than a Center inconvenience

2.2 The probability of failure for software is difficult to determine even late in the development cycle. However, Table 1 contains simple metrics on the software, the developer, and the development environment, which have proven to be indicators of future software problems. While these indicators are not precise, they provide order of magnitude estimates, which are adequate for assessing the need for IV&V. (The IV&V Facility and the NASA Software Working Group will further refine these indicators and their associated weighting factors as more data becomes available.)

3. Combining the software consequences of failure and the likelihood of failure rating from Section 2 yields a risk assessment, which can be used to identify the need for IV&V. The indication of whether IV&V is required is obtained by plotting in Figure 1 the intersection of the Consequences of Software Failure determination and the Total Likelihood of Failure determination. Application of these criteria simply determines that a project is a candidate for IV&V – not the level of IV&V nor the resources associated with the IV&V effort. These will be determined as a result of discussions between the project and the IV&V Facility.

3.1 Figure 1 shows a dark region of high risk where software consequences, likelihood of failure, or both are high. Projects having software that falls into this high-risk area shall undergo IV&V. The exception is those projects which have already done hardware/software integration. An IV&V would not be productive that late in the development cycle. These projects shall undergo a Software Independent Assessment (IA). (See Section 3.2.)

3.2 Figure 1 shows three gray regions of intermediate risk. Projects having software that falls into these areas shall undergo a Software IA. The IV&V Facility shall conduct the Software IA according to established IV&V Facility procedures. One purpose of the Software IA is to ensure that the software development does not have project-specific risk characteristics that would warrant the performance of IV&V. Should such characteristics be identified, a recommendation for IV&V performance will be made.

4. All projects containing software shall evaluate themselves against the criteria of this document to determine if a Software IA or an IV&V is

required and shall notify their Governing Program Management Council (GPMC) and/or the Center Director of the results. Projects identified as candidates for IV&V or Software IA shall be contacted by the IV&V Facility to discuss the appropriate level of effort to be applied.

Factors contributing to probability of software failure	Un-weighted probability of failure score					Weighting Factor	Likely-hood of failure rating
	1	2	4	8	16		
Software team complexity	Up to 5 people at one location	Up to 10 people at one location	Up to 20 people at one location or 10 people with external support	Up to 50 people at one location or 20 people with external support	More than 50 people at one location or 20 people with external support	X2	
Contractor Support	None	Contractor with minor tasks		Contractor with major tasks	Contractor with major tasks critical to project success	X2	
Organization Complexity*	One location	Two locations but same reporting chain	Multiple locations but same reporting chain	Multiple providers with prime sub relationship	Multiple providers with associate relationship	X1	
Schedule Pressure**	No deadline		Deadline is negotiable		Non-negotiable deadline	X2	
Process Maturity of Software Provider	Independent assessment of Capability Maturity Model (CMM) Level 4, 5	Independent assessment of CMM Level 3	Independent assessment of CMM Level 2	CMM Level 1 with record of repeated mission success	CMM Level 1 or equivalent	X2	
Degree of Innovation	Proven and accepted		Proven but new to the development organization		Cutting edge	X1	
Level of Integration	Simple - Stand alone				Extensive Integration Required	X2	
Requirement Maturity	Well defined objectives - No unknowns	Well defined objectives - Few unknowns		Preliminary objectives	Changing, ambiguous, or untestable objectives	X2	
Software Lines of Code***	Less than 50K		Over 500K		Over 1000K	X2	
Total							

Table 1 Likelihood of Failures Based on Software Environment

The following notes and definitions apply to Table 1:

* Organization complexity is an indirect measure of communications challenges inherit in the software developer. A single organization working from multiple locations faces a slightly greater challenge than an organization in one location. When the software development is accomplished by multiple organizations working for a single integrator, the development is significantly complicated. If the developing organizations are coequal such as in an associate contractor relationship (or a similar relationship between government entities) then there is no integrator. Experience has shown this arrangement to be extremely challenging as, no one is in charge.

** Under "schedule pressure" a deadline is negotiable if changing the deadline is possible although it may result in slightly increased cost, schedule delays, or negative publicity. A deadline is non-negotiable if it is driven by immovable event such as an upcoming launch window.

*** As the problems identified in IV&V are often mismatches between the intended use and the actual software built, "software lines of code" shall include reused software and autogenerated software.

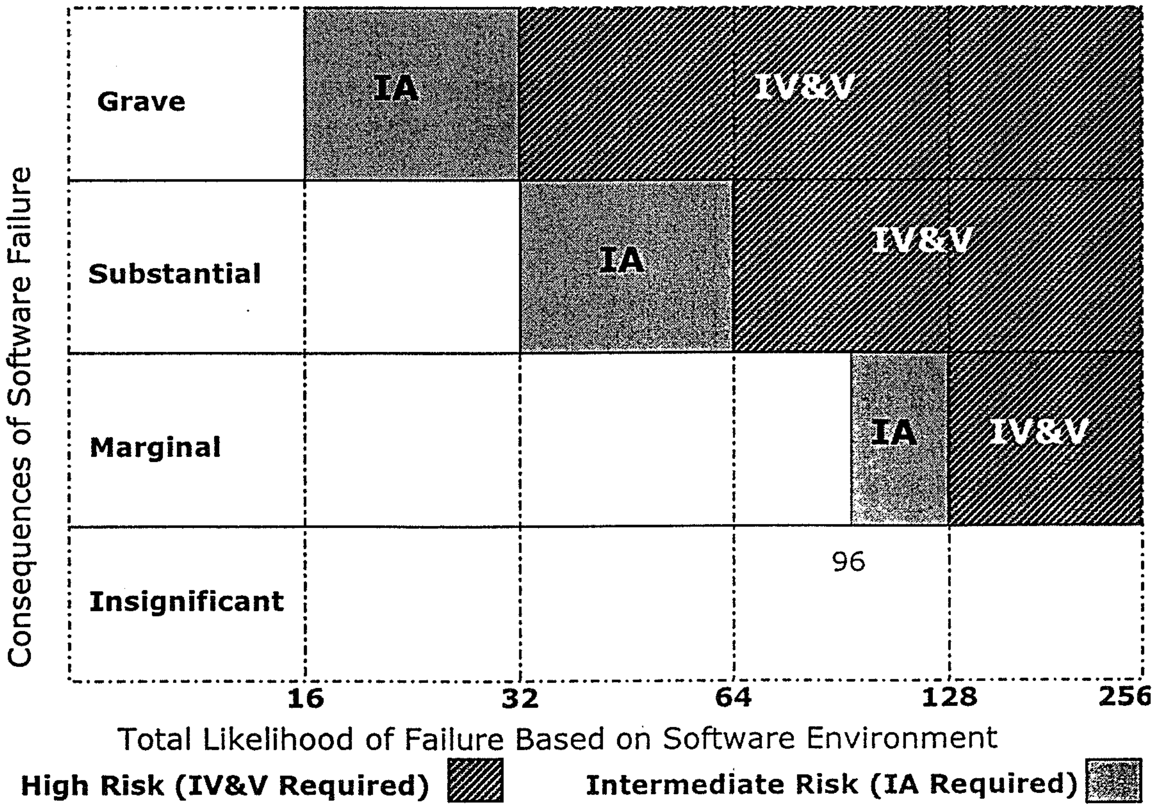

Figure 1 Software Risk