REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
09/30/2015		Oct 2008-Sep 2015

4. TITLE AND SUBTITLE

HPCMP CREATE™-AV Quality Assurance: Best Practices for Validating and Supporting Computation-Based Engineering Software

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Benjamin P. Hallissy, Joseph P. Laiosa, Theresa C. Shafer
David H. Hine, James R. Forsythe, Jennifer Abras,
NAVAIR, Patuxent River NAS, MD
Nathan S. Hariharan, Cynthia Dahl
HPCMP, Lorton VA

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

HPCMP CREATE
10501 Furnace Road Ste 101
Lorton, VA 22079-2624

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

HPCMP CREATE
10501 Furnace Road Ste 101
Lorton, VA 22079-2624

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Distribution A

13. SUPPLEMENTARY NOTES

To be published in Computing in Science and Engineering Vol 18, No. 1 Jan/Feb 2016

14. ABSTRACT

From the perspective of a user of computational-based engineering software, the two most relevant questions are: (1) "Does this tool adequately perform the analysis I need?" and (2) "Does this tool adequately perform any and all advertised capabilities?" This paper will describe how the HPCMP CREATE Air Vehicles (AV) project ensures positive answers to these questions through the functions performed by the Quality Assurance group. Industry quality standards will be discussed and their strengths and weaknesses within the CREATE-AV framework addressed. Work toward the

15. SUBJECT TERMS

HPCMP CREATE, Quality Assurance, Aviation, Acquisition Engineering, Digital Engineering

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**	UU	13	Kevin Newmeyer
UU	UU	UU			**19b. TELEPHONE NUMBER** *(include area code)* 703-812-4417

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
30 SEP 2015		**00-10-2008 to 00-09-2015**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
HPCMP CREATE (Trade Mark)-AV Quality Assurance: Best Practices for Validating and Supporting Computation-Based Engineering Software	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **HPCMP CREATE,,10501 Furnace Road Ste 101,,Lorton, ,VA, 22079**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
To be published in Computing in Science and Engineering Vol 18, No. 1 Jan/Feb 2016

14. ABSTRACT
From the perspective of a user of computational-based engineering software, the two most relevant questions are: (1) Does this tool adequately perform the analysis I need and (2) ???Does this tool adequately perform any and all advertised capabilities???? This paper will describe how the HPCMP CREATE Air Vehicles (AV) project ensures positive answers to these questions through the functions performed by the Quality Assurance group. Industry quality standards will be discussed and their strengths and weaknesses within the CREATE-AV framework addressed. Work toward the

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a REPORT **unclassified**	b ABSTRACT **unclassified**	c THIS PAGE **unclassified**	**Same as Report (SAR)**	**14**	

HPCMP CREATE™-AV Quality Assurance: Best Practices for Validating and Supporting Computation-Based Engineering Software

Benjamin P. Hallissy, Joseph P. Laiosa, Theresa C. Shafer
David H. Hine, James R. Forsythe, Jennifer Abras,
NAVAIR, Patuxent River NAS, MD

Nathan S. Hariharan, Cynthia Dahl
HPCMP, Lorton VA

Abstract

From the perspective of a user of computational-based engineering software, the two most relevant questions are: (1) "Does this tool adequately perform the analysis I need?" and (2) "Does this tool adequately perform any and all advertised capabilities?" This paper will describe how the HPCMP CREATE Air Vehicles (AV) project ensures positive answers to these questions through the functions performed by the Quality Assurance group. Industry quality standards will be discussed and their strengths and weaknesses within the CREATE-AV framework addressed. Work toward the verification and validation of the CREATE-AV suite of tools will be presented. The paper will conclude with Quality Assurance lessons and best practices learned over the course of an extended (7+ year) effort to field next-generation multiphysics aviation design tools to the DoD community.

I. The Need for Quality Assurance

QUALITY, specifically *software quality*, affects each of our lives on a daily basis - but we usually don't notice. One rarely has to worry about the workings of the engine control module in a sedan, the autofocus algorithms in a digital SLR, or the autopilot on an airliner, unless out of simple curiosity. The software that we interact with each day tends to perform as we expect, and failures are a rare exception. That those software products work, almost without fail, to the consumer's satisfaction is the result of a concerted effort to build quality software. The same concerted effort should be present in the production of software designed for computation-based engineering (CBE).

Quality may be defined in very different ways, depending on one's perspective. Kitchenham and Pfleeger [1] quoting David Garvin, explain that there are five views on quality. (1) The transcendental view which takes quality as something that can be recognized and sought but not defined, (2) the user view, which sees quality as satisfactorily performing a desired function, (3) the manufacturing view sees quality as conformance to specification, (4) the product view sees quality as an inherent characteristic of the product or software, and (5) the value-based view, which determines quality based on the products perceived value to a customer. When discussing quality within this paper, the user and manufacturing views will be assumed. Therefore, the relevant questions that must be answered when evaluating quality are: (1) "Does this CBE tool adequately perform the analysis I (the user) need?" and (2) "Does this CBE tool adequately perform any and all advertised capabilities?" respectively.

Within the HPCMP CREATE umbrella (see also: The US Department of Defense (DoD) High Performance Computing Modernization Program (HPCMP) Computational Research Engineering Acquisition Tools and Environments (CREATE) Program, D. Post et al, this issue), the CREATE Air Vehicles (AV) project aims to improve the DoD acquisition process by developing and deploying advanced computational engineering design and simulation tools for military aircraft. While quality in newly developed software should be built in from the ground up, this paper will describe how the CREATE-AV project also answers the above questions through the functions performed by the independent CREATE-AV Quality Assurance group. Industry standards will be discussed and

their strengths and weaknesses within the CREATE-AV framework addressed. Work toward the verification and validation of the CREATE-AV suite of tools will be presented. The paper will conclude with Quality Assurance lessons and best practices learned over the course of an extended (7+ year) effort to field next-generation multiphysics aviation design tools to the DoD community.

II. Industry Standards and Practical Considerations

A. Quality Management Standards

Two standards for quality management are commonly used today, the ISO 9000 series (and to a lesser, although more software-relevant extent, ISO 10000) [2], and Carnegie Mellon's Capability Maturity Model Integration (CMMI) [2]. These guidelines tend to focus on processes, and for good reason. The quality of a software package is heavily influenced by the quality of the processes used to develop, test and maintain it. An external certification of these quality processes has great value in a large commercial environment, but in a (relatively) small software development effort, the cost and time of acquiring these certifications can be prohibitive.

B. Verification and Validation Standards

The American Institute of Aeronautics and Astronautics (AIAA) and American Society of Mechanical Engineers (ASME) have both published guides for the verification and validation of software relevant to their respective disciplines. The AIAA, concerned primarily with computational fluid dynamics, published first in 1998 [3] and ASME, concerned primarily with computational structural mechanics, followed with a guide in 2006 [4]. These guides provide a solid foundation for the subject of Verification, Validation, and Uncertainty Quantification (VV&UQ), but as the AIAA guide acknowledges, the required accuracy and certainty of modeling and simulation is not (and cannot be) a rigid standard. Each new application will have a different accuracy requirement depending on the project's budget, schedule and risk (probability × consequence) of failure. CREATE V&V best practices build on these standards, and also align with the recently released National Academies guidelines in [5].

C. Limitations in Practice

For the CBE community, Roache [6] suggests that quality management standards like ISO9000 are "expensive, cumbersome and not conducive to creative work, e.g. in algorithm development." (pp. 367) Industry V&V standards, on the other hand, are extremely valuable guidelines, but are necessarily flexible, and application-dependent, so we return once more to the user perspective of quality. Most expectations of CBE software users can be grouped into 5 categories:

1. Robustness across applicable problem space
2. Accuracy across applicable problem space
3. Efficiency (short time to solution)
4. Usability (user interface, documentation, training)
5. Consistency (no regression in 1-4)

The first three categories of customer expectations are deceptively simple. In fact, these expectations introduce considerable challenges to any QA program, because they ironically depend heavily on the experience level and behavior of the user. Robustness and accuracy can be challenged by introduction of a complex geometry or a poorly generated mesh. Application of a software product to unvalidated and/or uncalibrated scenarios (e.g. hypersonics) can generate results that should not be trusted without additional validation. For every application, there are likely ideal points in the solver parameter space that maximize robustness, accuracy, or efficiency, but the point that maximizes one, is not likely the same as the point that maximizes the other. Therefore, the selection of default solver parameters is both difficult and critical to creating a satisfying out-of-the-box solver experience. Strazisar and Denton [7], in a blind comparison of turbomachinery prediction using CFD, noted that three users, each using the same CFD code (but running independently), generated three different predictions! In their words, "It is clear that even tighter controls of the CFD parameter space are necessary [in future comparative exercises]." Another way to think about this, says Roache [6], is that QA or certification of the user base, as well as the code itself, is required, to ensure that validation results are independently reproducible. Yet, to the authors' knowledge, no commercial or governmental software package requires the user to have a certification to operate, so the user will remain an unknown influence in the "quality" of a CBE software product.

Usability is likewise a function of the user, and is subjective to a large extent. Measuring usability is somewhat of an art, but several proprietary techniques have been developed. One open-source option is the USE Questionnaire **[8]**, which stands for Usefulness, Satisfaction, and Ease of Use (and to a lesser extent, Ease of Learning). Administering, gathering, and analyzing the USE results would hopefully bring some science to the question of usability. However, in order for the results to be statistically significant, a large number of surveys must be collected, and a relatively small user base combined with typically low response rates presents a further complication.

The well-known saying "consistency is key," also holds true for software development. Having a large number of distributed developers (or, cooks in the kitchen) can cause cross-contamination of code and lead to the introduction of faults. Yet, from version to version, users expect to only move forward in categories 1-4, i.e. version N+1 should be MORE robust, MORE accurate, MORE efficient, and MORE usable, than version N. To ensure this, QA must have a complete set of regression tests (automated if possible), but that is not all. Consistency requires manual testing, and there is no way around this fact. GUI exercise and output examination (flow visualization, structural deformation, etc.) are critical. Consistent and helpful support resources are also play a role in an evaluation of quality. As a code matures, the support infrastructure must at least keep pace with the code capability and growing user base.

III. CREATE-AV Quality Assurance

The following section describes how CREATE-AV Quality Assurance addresses both the ideal and the practical views of software quality assurance.

A. User Base and QA Team Status

In order to best serve the customer, QA frequently considers the makeup of our customer base. CREATE-AV currently (Feb 2015) has about 350 registered software users who have actively participated in the program within the past calendar year. Currently, CREATE-AV users come from each of the three services (Army, Navy, and Air Force), NASA, industry and academia, and the breakdown by affiliation is given in Figure 1. All DoD employees and others involved with a current DoD contract are eligible to access and run CREATE-AV products, which creates a geographically diverse user base, shown in Figure 2.

The QA Team itself is currently headquartered at Patuxent River Naval Air Station, Patuxent River MD, has representatives in both Florida and Connecticut, and interacts closely with the AV development teams across the nation, based primarily at NASA Ames, Wright Patterson AFB, and Eglin AFB. Interacting with these distributed organizations requires close communication and tools for efficient collaborative work. HPCMP CREATE has enabled this through the use of Atlassian JIRA, discussed in detail in [9].

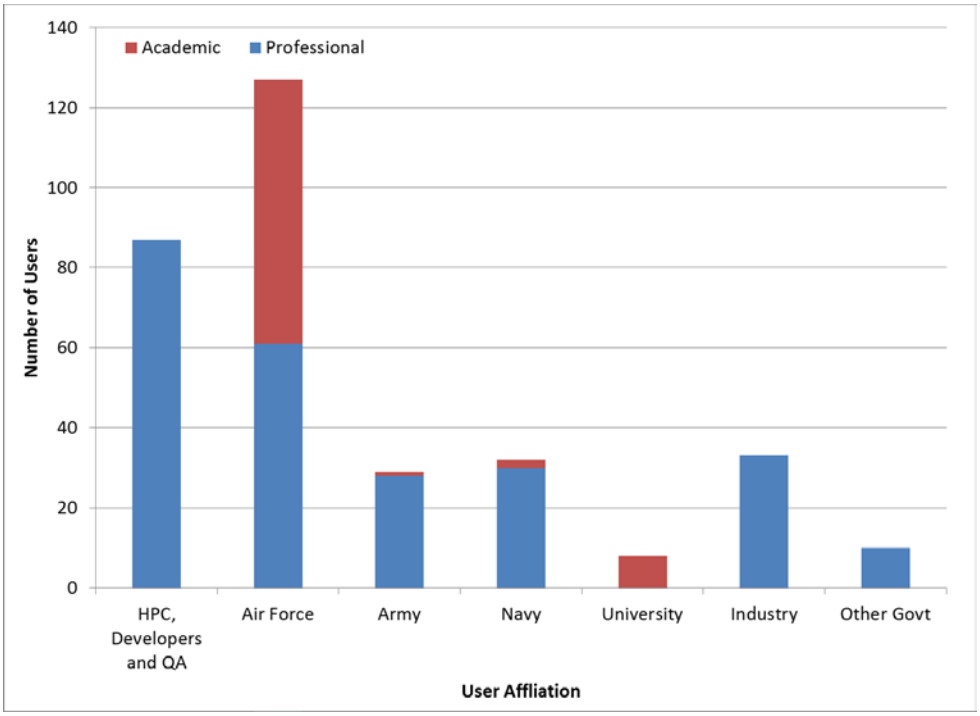

Figure 1. User base affiliation

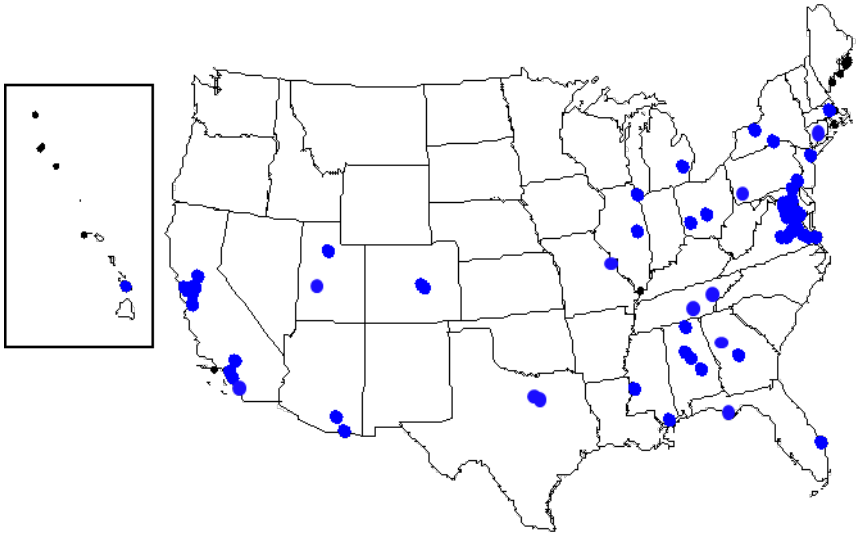

Figure 2. Geographical user base distribution (calculated from user forum activity)

B. Verification, Validation, and Uncertainty Quantification

W. Edwards Demming, winner of the National Medal of Technology and Innovation in 1987, once said "In God we trust, all others bring data." Trust in a computational tool of any kind is a rare and difficult thing to develop. The mutually dependent exercises of verification, validation, and uncertainty quantification are a prerequisite in order to gain the trust of potential scientist and engineer users.

Extensive verification and validation has been performed on the CREATE-AV tools Kestrel and Helios. Recent V&V efforts include studies on 2D and 3D external aerodynamics configurations [10, 11, 12], unsteady weapons bay aeroacoustics [13], fluid-structure interaction [14], validations on Navy and Air Force-specific applications [15,

16], low speed rotorcraft fuselage aerodynamics [17], and rotorcraft hover calculations [18]. In FY14, CREATE-AV sponsored a broad agency announcement to the aerospace industry to encourage evaluation of CREATE tools in a production environment with its associated challenges: restrictive security, established processes, and different (often older) computational environments. These evaluations, by Boeing [19, 20], Lockheed [21], and Sikorsky [22], generated substantial workflow feedback which has been incorporated into each product's roadmap. Overall the BAA participants' response was overwhelmingly positive. With respect to technical support, one participant said "QA support is excellent, as responsive as and more competent than front-line support of most vendors we interact with."

The spirit of CREATE-AV Quality Assurance testing (evidenced above and discussed below in part C) is closely aligned with the recommendations summarized in the National Academies report on Verification, Validation and Uncertainty Quantification [5] . CREATE tools will be applied to solve problems across a large variety of flow regimes, and applications. In order to ensure that the basic physics modeled in each application is fundamentally sound, a set of phenomena of interest (PoI) should be defined, and validation efforts should be focused in these areas. Only after confidence is generated in solutions to the simple problems, can complex applications be trusted. A sampling of PoI (red boxes) and validation cases used to investigate those phenomena (bulleted lists) for CREATE Quality Assurance is given in Figure 3 below.

Figure 3. V&V Phenomena of Interest

C. Quality Assurance Testing

In the CREATE-AV program, computational tools endure six levels of testing prior to general release. These are listed in chronological order below, and their relationships and owners are depicted left-to-right in Figure 4.

1. Unit Tests
2. Integration Tests
3. Alpha Tests
4. Quality Assurance Test (QAT) / Regression Tests
5. Exit Tests (Figure 6)
6. Beta Tests (Omitted for interim releases)

A good summary discussion of levels 1 through 3 can be found in [23] and also (more generally) in most software development texts. The first two levels (Unit and Integration) are largely automated and are performed

automatically on a nightly basis by each development team. Level 3 (Alpha) consists of both regression testing and check out V&V for any newly introduced capabilities. Responsibility for levels 4 and 5 falls to the independent quality assurance team and these levels will be detailed in the current section and the section on Software Release and Deployment, respectively. Level 6, or Beta testing, is a common procedure and is well known in the software development world, and to any early adopters of commercial software.

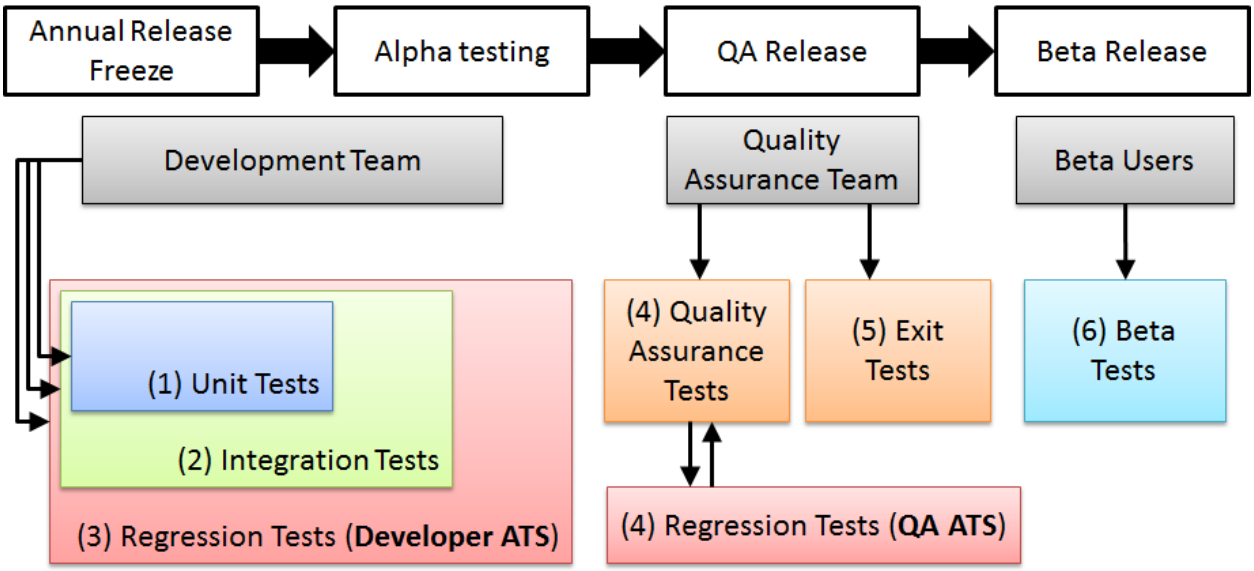

Figure 4. Levels of testing within CREATE-AV

1. Quality Assurance Test Selection

QA test selection occurs concurrently with solver development, and the vast array of sources for these tests is outlined in Figure 5. As shown in the figure, when a capability and its corresponding test(s) are mature, they are integrated into the automatic Acceptance Test System(ATS), and serve as capability regression tests. Several common sources of tests are described in more detail below:

1. *User Support Issues*: Support requests can reveal shortcomings in both the product itself, or the associated training/documentation. These support tickets can be converted into feature requests, or bug reports, and the cases (provided by the user) which demonstrate the errors can then be used to verify that a fix has been found. Afterward, these cases may prove useful for regression testing.
2. *Strategic TARgeting* (*STAR) Project Requirements*: STAR Projects are research efforts funded jointly by CREATE-AV and DoD program offices. These projects often push the boundaries of CREATE product capabilities, and in doing so suggest areas for future effort. STAR Projects are monitored closely by the Quality Assurance group in order to support the researchers, which allows QA direct access to the user wants, needs and frustrations uncovered when using CREATE tools.
3. *Annual Requirements Reconciliation*: This meeting involves the CREATE-AV planning team, which represents the interests of stakeholders across DoD. Inputs gathered here include the large capability gaps that CREATE-AV software must fill. Whenever a new capability is added to a CREATE-AV product, a QA test is created to validate that capability, to the extent possible.
4. *Classic Fundamental Physics Validation*: These cases are omnipresent in the literature, and they serve to both validate and benchmark CREATE-AV products relative to the current state of the art. Due to the large body of literature surrounding these validation cases, both code-to-code comparisons, as well as code-to-data comparisons can often be made.
5. *Code Weaknesses*: Often, execution of a prior QA test reveals an area of potential weakness in a solver, for example, a lack of robustness for a certain application or undue sensitivity to an input parameter. In this case, new tests are created or old tests modified (in collaboration with the development teams) in order to exercise and harden the weak capability.

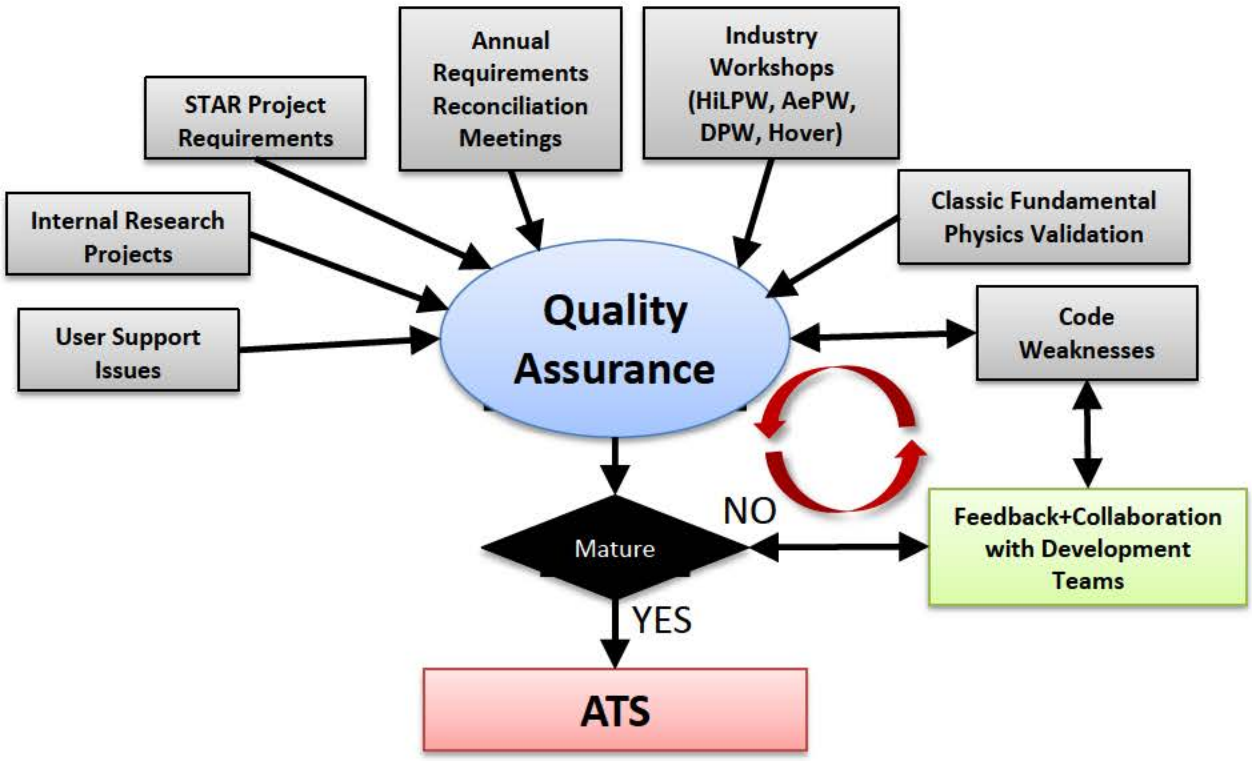

Figure 5. Sources of Quality Assurance Tests

2. Quality Assurance Test Execution

In the early days of CREATE, the Quality Assurance Test (QAT) was called the Product Acceptance Test (PAT). The idea (similar to DoD acquisition process) was that all software products would have to pass a final set of pass/fail tests performed by the independent QA group prior to a yearly release. As releases became more frequent, and as capabilities began to be introduced and matured over several interim versions, the sense of a massive, one-shot-per-year test gave way to a collaborative testing process in which QA is engaged continually with the development teams year-round. QATs represent a broader perspective than the simple "pass/fail" tests of old. QATs aim to:

1. *Inject User Perspective*: Design test cases that challenge what has been demonstrated. Push the boundaries in the applications that users require.
2. *Go Beyond High Level Outputs*: Investigate the physics, numerics, and underlying accuracy of CREATE-AV tools. Many users do not venture beyond integrated load values, like lift and drag, and developers may not have the time.
3. *Complement Alpha Tests*: Work with the development teams to fill in any gaps. Query and analyze what the development teams haven't focused upon.
4. *Investigate Weaknesses*: Find and probe weakness found from QA, STAR Projects, and user interactions.
5. *Understand Model Dependencies*: Identify and quantify effects of grid and solver parameters. Calibrate solvers to problems within the domain of interest, and for the relevant quantities of interest.
6. *Generate Independent Results*: Ensure that prior results are reproducible by an independent group.

D. Software Release and Deployment

The software release process (levels 4 and 5 of software testing) is highlighted in Figure 6. On the far left-hand side, a new product version is delivered to QA, and on the far right hand side, a product is approved by QA for release. The process may involve several iterations with the development team. For instance, if any test in the fundamental

physics or applications test suite fails, the code goes back to the development team to be fixed and retagged as a new version. If a bug or slowdown has been introduced into the GUI, the code goes back to the development team to be fixed and retagged. In the case of interim releases, only light exercise of the updated GUI is performed, because this is a costly step in man-hours. For major releases, the entire QA team will be hands-on with the GUI, working to root out any deficiencies.

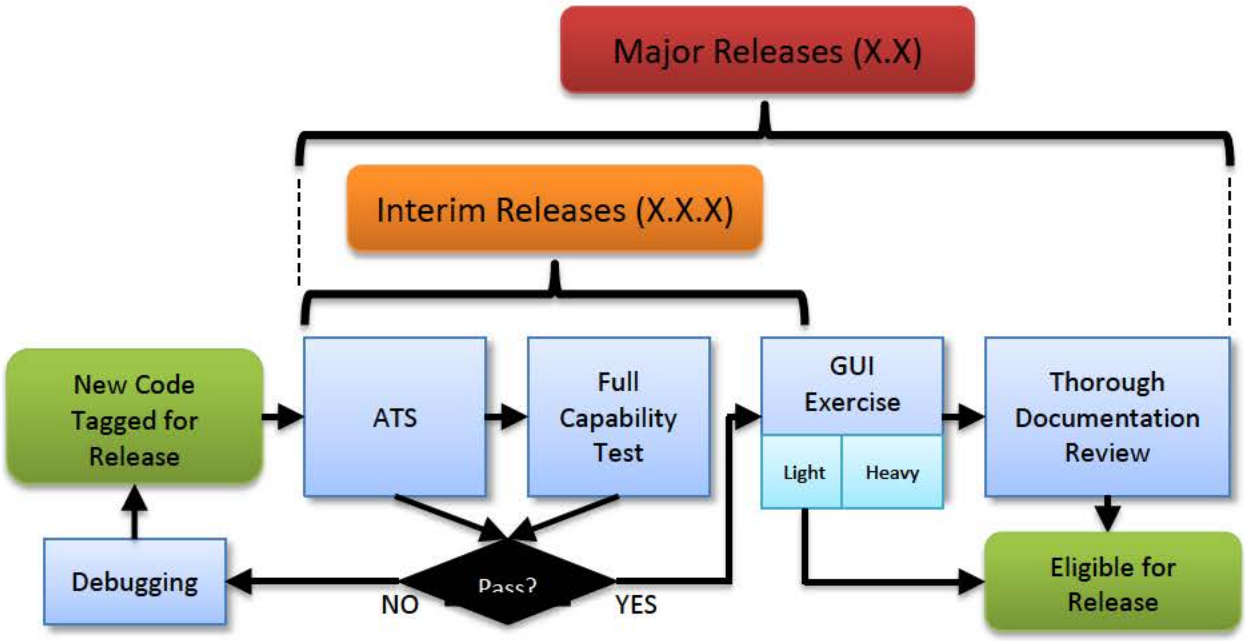

Figure 6. Exit Testing Procedure for New Releases

Every release, both major and interim goes through a final QA inspection which confirms several things are true:

1. There are no outstanding, unresolved "Blocker" issues, those of the severity which would delay release. Note that this does not mean there are NO bugs, or NO feature requests left to complete (if we're being pragmatic, there always are).
2. All software documentation is present and current
3. Version release notes are present and current
4. The CREATE-AV Buildmaster and QA have developed a release schedule (Which HPC systems will the software be deployed to? In what order? Any risks present (e.g. new HPC machines) or mitigations needed?)
5. (For general release) Training material has be updated and is ready to accompany the release.

Following completion of the inspection, the software is deployed remotely by the CREATE-AV build masters in three ways. (1) Binary executables – which include the GUI, solver, and any accompanying utilities for use on Linux workstations, (2) HPC executables – located in centralized locations on as many as 15 remote Defense Supercomputing Resource Centers (DSRCs), and (3) In-house clusters at both government research centers and industry (if applicable).

Finally, even though a code version is eligible for release, it may not necessarily be introduced to the user base. The most common reason for suppressing a release at this stage is to prevent something we call *user whiplash*, where users are pulled from version to version in pursuit of minor improvements. Releases must pass a final (and admittedly subjective) "urgency" screening, to ensure that the potential benefit to the user outweighs the potential time cost and whiplash.

E. Software Training

On-site training sessions are held around the country as requested by DoD organizations. The size of these training sessions has ranged from 3 to 30 users per session, with varying levels of formality, but in each case students have hands-on access to the software. Hands-on practice combined with instructor guidance is consistently cited as the most beneficial aspect of training. We've found that an instructor-student ratio of 1:5 is generally sufficient. To date, as a broad measure of training quality, over 93% of participants said they would recommend the training to a friend.

The training material is detailed and flexible however, and can be used for self-paced instruction. In addition, the user community forums (one for each CREATE-AV product) are an active learning environment with users posting suggestions, bugs, and workarounds. We have also developed several short video tutorials which supplement the written training material and provide an efficient refresher when returning to the CREATE toolset.

F. User Support and Issue Triage

Once the software moves out the door and into the user's hands, CREATE-AV's multi-tiered support structure, depicted in Figure 7 takes over. The users are first provided with several self-help options – tutorials, documentation, and quick-start guides. If no resolution is found following a self-guided investigation, or if the user prefers direct communication, they may contact CREATE-AV quality assurance by forum, phone, or an online ticket submission system. QA will then perform an analysis of the issue and act to facilitate a resolution, based on the nature of the problem. Importantly, the support ticket is tracked by QA during the process, and the user is kept informed through all stages of issue resolution.

When confusion is caused by a user error, a lack of training, or a lack of documentation, QA recommends a fix or workaround to the user, modifies the training/documentation as needed, and resolves the ticket. In the case of bug reports, QA attempts to duplicate the bug on a simple test case (which can be especially important when dealing with proprietary or sensitive simulations) and narrow down the potential scope of the bug (e.g. answer the question: "In which version was it introduced?") before handing the ticket off to the product development team.

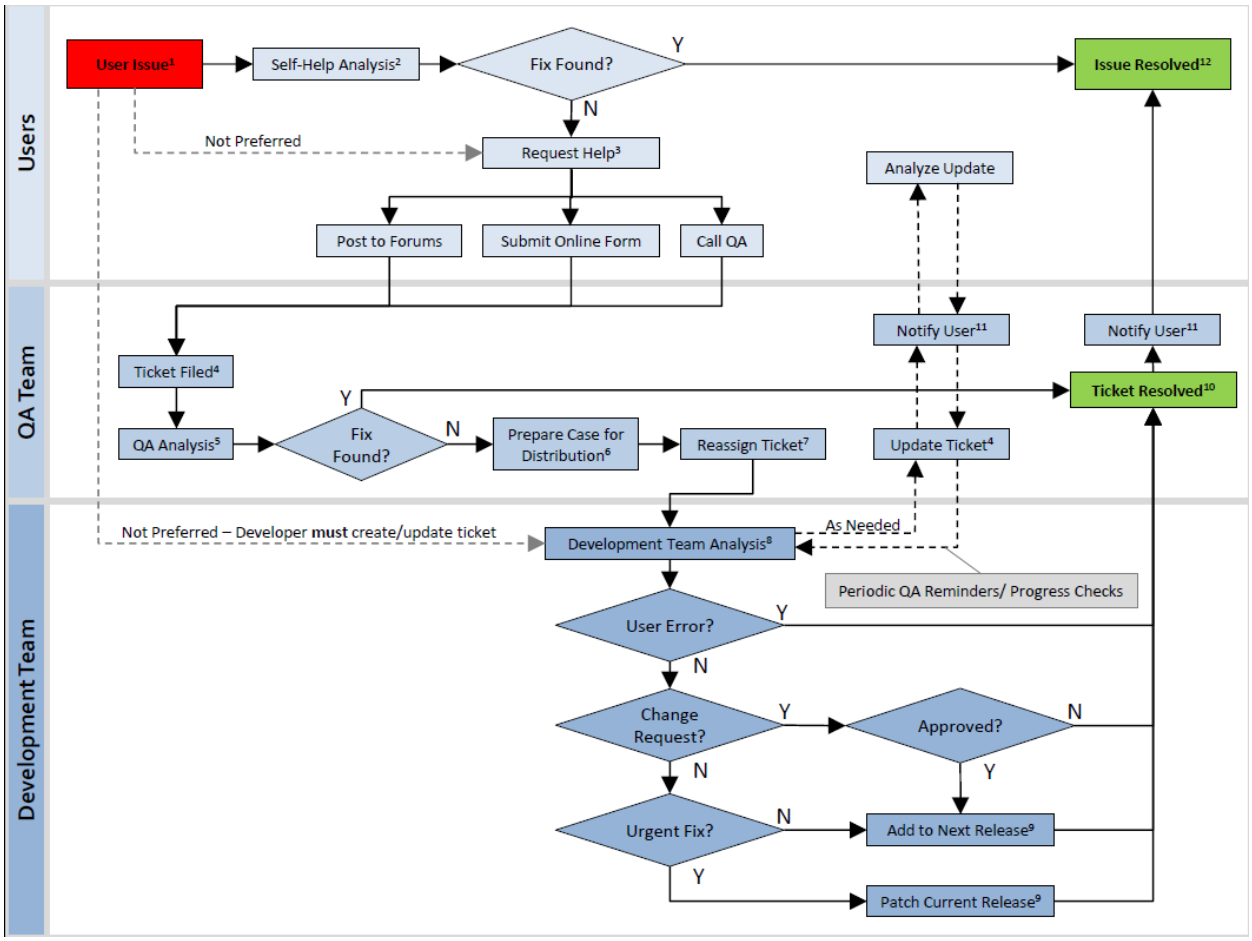

Figure 7. CREATE-AV Quality Assurance User Support Model

At the time of the hand-off, QA will assign a priority level to the bug or feature request to help the development team prioritize their work. Assigning consistent priorities across product lines and individual QA team members presents several difficulties. First, the priority level should represent the importance of the issue to the user or users, and not QA's analysis of difficulty or scope. Second, if all else is equal, the priority of a bug (known deficiency) should always outweigh that of a feature request, so the rating system needs to be slightly biased. Third, the priority level should not necessarily favor newer issues over older – but the need to "put out" the latest fires can make this challenging. Finally, the priorities should be minimally subjective, despite the fact that they are chosen by human beings who have (possibly strong) opinions. We have recently developed a rubric [24] that addresses each of these concerns using 5 categores, each scored on a scale of 0 to 2. Summing the scores across categories, the total rating – which we call the NAVIS rating (for **N**umber, **A**lternatives, **V**alue, **I**mpact, and **S**ubjective Analysis) – can range from 0 (lowest priority) to 10 (highest priority). Table 1 explains the prioritization of the resulting NAVIS ratings. Note that 20% of the NAVIS rating is intentionally still subjective. This percentage is used to break ties if need be, and to promote issues that are in line with the CREATE-AV mission: reduce cost and improve the effectiveness of DoD acquisition through the use of CBE software.

Table 1. NAVIS Prioritization Levels

NAVIS Rating	JIRA Priority	Summary

0 to 2	**Trivial**	Low priority issue. Required developer workload not assessed.
3 to 4	**Minor**	Medium priority issue. Workload may be assessed, but issue NOT a contender for current development team work.
5 to 6	**Major**	High priority issue. Workload assessed. Issue is a contender for current development team work.
7 to 8	**Critical**	Highest priority issue that is being worked on a daily basis. Created when similar requests come in from multiple (or high-profile) users, or when bugs are found with painful/limited workarounds.
9 to 10	**Blocker**	Used sparingly, in cases where important work is stopped with no workaround, or a large bug is unearthed that could have big consequences.

IV. Best Practices / Lessons Learned

Below is a summary of the quality assurance "lessons learned" throughout CREATE's 7-year (ongoing) experience in deploying CBE tools to the DoD workforce.

A. User Training

- Clearly define the scope of all training, both in advertisements and the day of. The most effective training is tailored to experience level, both level of subject expertise (e.g. CFD) and experience with a particular code (Kestrel or Helios).
- Emphasize the best-practices of your code, and then emphasize them again. Provide easily accessible reference material. Otherwise, if a new user is familiar with a different type of simulation or a different software package, they will tend to revert to parameters and settings that are familiar.
- Train the folks who will be using the code, not their managers (no matter how technical). If managers insist on training, present a high-level capability overview.
- Without *knowing* your users, you can not *train* them effectively

B. User Support

- Gather as much data about users as reasonable. (What is their interest in the code, how did they hear about it, do they have experience with these types of codes, do they want training, are any of their colleagues using it, etc.)
- Keep users informed as bugs are fixed and feature requests implemented.
- Without *knowing* your users, you can not *support* them effectively.
- Clearly explain which changes or updates are contained in each new product version to prevent unnecessary user whiplash.

C. Quality Assurance

- Maintain close communication with development teams. Weekly or bi-weekly stand-up meetings go along way toward maintaining that relationship.
- QA cannot be seen as or positioned as the adversary of the developer. Rather, QA can be thought of as an aid – able to remove (to some extent) the support and training burden from the overtaxed developer, and able to pass user needs and feedback directly to the development team – valuable grassroots feedback that can be missed with your head buried in code.
- Despite its clear importance, QA's focus should not be *entirely* on VV&UQ. Consistent (up to date) and thorough documentation is critical in order to minimize user frustration. Training must correctly set the expectations of users.

D. VV&UQ

- Automate basic testing phases (unit, integration, system tests).
- Automate regression and validation tests for each advertised capability (e.g. relative mesh motion, moving control surfaces, actuator disc, etc.)
- VV&UQ is computationally and human resource intensive – must prioritize efforts based on immediate software requirements and use cases.
- To the extent possible, limit manual testing to three primary efforts:
    - GUI exercising
    - Closely inspecting the results (spot checks) of automatic tests
    - Development of tests for NEW capabilities, or known weaknesses in automatic test suite.

V. Conclusion

The definition of quality assurance from Rechard et al [25] succinctly describes the products that CREATE-AV strives to produce: "*QA as related to computer software means ensuring that the software consistently does what it is supposed to do to meet the expectations of the customer (i.e. recipient, purchaser, or benficiary of the work)*". This paper has explained how the CREATE-AV program meets this all-important expectation, through the lenses of VV&UQ, quality assurance testing, software deployment, user training, and user support.

Acknowledgments

Material presented in this paper is a product of the CREATE-AV Element of the Computational Research and Engineering for Acquisition Tools and Environments (CREATE) Program sponsored by the U.S. Department of Defense HPC Modernization Program Office. The authors would like to thank the Kestrel, Helios, Firebolt, DaVinci and CREATE-MG Capstone teams for their rigorous development work, and the CREATE-AV management team for their guidance and support.

References

[1] B. Kitchenham and S. L. Pfleeger, "Software Quality: The Elusive Target," *IEEE Software,* pp. 12-21, January 1996.

[2] D. S. Craft, "CMMI vs ISO," EDS, April 2007. [Online]. [Accessed May 2013].

[3] AIAA Committee on Standards, "Guide for the Verification and Validation of Computational Fluid Dynamics Simulations," American Institute of Aeronautics and Astronautics, Reston, VA, 1998.

[4] ASME, "Guide for Verification and Validation in Computational Solid Mechanics," American Society of Mechanical Engineers, 2006.

[5] National Research Council, Assessing the Reliability of Complex Models: Mathematical and Statistical Foundations of Verification, Validation, and Uncertainty Quantification, Washington, D.C.: National Academies Press, 2012.

[6] P. J. Roache, Verification and Validation in Computational Science and Engineering, Albuquerque, New Mexico: Hermosa Publishers, 1998.

[7] A. J. Strazisar and J. D. Denton, "CFD Code Assessment in Turbomachinery - A Progress Report," *ASME Global Gas Turbine News*, pp. 12-14, May/June 1995.

[8] A. M. Lund, "Measuring Usability with the USE Questionnaire," [Online]. Available: http://www.stcsig.org/usability/newsletter/0110_measuring_with_use html. [Accessed 15 November 2013].

[9] C. Atwood and M. Murphy, "Building Large-Scale Software in the DoD," *DoD HPCMP HPCinsights*, pp. 7-11, Spring 2013.

[10] J. R. Forsythe, B. P. Hallissy, D. Hine, J. Laiosa and T. Shafer, "Fundamental Physics Validation Using HPCMP CREATE-AV Kestrel: Part II," in *AIAA Science and Technology Forum and Exposition*, National Harbor, MD, 2014.

[11] D. Hine, J. Forsythe, B. P. Hallissy, T. Shafer and J. Laiosa, "Fundamental Physics Validation Using HPCMP CREATE-AV Kestrel: Part I," in *AIAA Science and Technology Forum and Exposition*, National Harbor, MD, 2014.

[12] T. Shafer, T. A. Eymann, J. R. Forsythe, B. P. Hallissy and D. Hine, "Applications of CREATE-AV Kestrel v5 with Cartesian Adaptive Mesh Refinement," in *AIAA Science and Technology Forum and Exposition*, Orlando, FL, 2015.

[13] B. P. Hallissy and N. Hariharan, "Prediction of Unsteady Flow in UCAV Weapon's Bay Using CREATE-AV's Kestrel," in *AIAA Science and Technology Forum and Exposition*, National Harbor, MD, 2014.

[14] S. E. Lamberson and B. P. Hallissy, "Aeroelastic Simulations with Modal and Finite-Element Structural Solvers Using CREATE-AV/Kestrel v5," in *53rd AIAA Aerospace Sciences Meeting*, Orlando, FL, 2015.

[15] T. Shafer, B. Green, B. P. Hallissy and D. Hine, "Advanced Navy Applications Using CREATE-AV Kestrel," in *AIAA Science and Technology Forum and Exposition*, National Harbor, MD, 2014.

[16] C. Lillian, "A-10 LAU-131 A/A Jettison CFD Analysis," in *AIAA Science and Technology Forum and Exposition*, National Harbor, MD, 2014.

[17] J. Abras and N. Hariharan, "CFD Solver Comparison of Low Mach Flow Over the ROBIN Fuselage," in *AIAA Science and Technology Forum and Exposition*, National Harbor, MD, 2014.

[18] J. Abras and N. S. Hariharan, "Comparison of CFD Hover Predictions on the S-76 Rotor," in *53rd AIAA Aerospace Sciences Meeting*, Orlando, FL, 2015.

[19] D. Stookesberry, "An Industry Assessment of HPCMP CREATE-AV Kestrel," in *53rd AIAA Aerospace Sciences Meeting*, Orlando, FL, 2015.

[20] R. Narducci, "An Industry Assessment of HPCMP CREATE-AV Helios," in *53rd AIAA Aerospace Sciences Meeting*, Orlando, FL, 2015.

[21] B. R. Smith, "An Assessment of CREATE-AV Kestrel for F-35 Aero/Performance Applications," in *53rd AIAA Aerospace Sciences Meeting*, Orlando, FL, 2015.

[22] E. Reed and A. T. Egolf, "Coaxial Rotor Wake and Prop Induction Impact on a Horizontal Tail Using HPCMP CREATE™-AV Helios," in *53rd AIAA Aerospace Sciences Meeting*, Orlando, FL, 2015.

[23] B. Jayaraman, A. M. Wissink, S. Shende, S. Adamec and V. Sankaran, "Extensible Software Engineering Practices for the Helios High-Fidelity Rotary-Wing Simulation Code," in *49th AIAA Aerospace Sciences Meeting*, Orlando, FL, 2011.

[24] B. P. Hallissy, D. Hine, J. P. Laiosa, T. C. Shafer, J. R. Forsythe, J. Abras, C. Dahl, C. S. Lillian and N. S. Hariharan, "CREATE-AV Quality Assurance: Best Practices for Validating and Supporting Computation-Based Engineering Software," in *52nd Aerospace Sciences Meeting*, National Harbor, MD, 2014.

[25] R. P. Rechard, P. J. Roache, R. L. Blaine, A. P. Gilkey and D. K. Rudeen, "Quality Assurance Procedures for Computer Software Supporting Performance Assessments of the Waste Isolation Pilot Plant," Albuqueque, NM, April 1991.