NPS-AM-10-044

EXCERPT FROM THE

PROCEEDINGS

OF THE

SEVENTH ANNUAL ACQUISITION RESEARCH SYMPOSIUM WEDNESDAY SESSIONS VOLUME I

Acquisition Research

Creating Synergy for Informed Change

May 12 - 13, 2010

Published: 30 April 2010

Prepared for: Naval Postgraduate School, Monterey, California 93943

The research presented at the symposium was supported by the Acquisition Chair of the Graduate School of Business & Public Policy at the Naval Postgraduate School.

To request Defense Acquisition Research or to become a research sponsor, please contact:

NPS Acquisition Research Program
Attn: James B. Greene, RADM, USN, (Ret.)
Acquisition Chair
Graduate School of Business and Public Policy
Naval Postgraduate School
555 Dyer Road, Room 332
Monterey, CA 93943-5103
Tel: (831) 656-2092
Fax: (831) 656-2253
E-mail: jbgreene@nps.edu

Copies of the Acquisition Sponsored Research Reports may be printed from our website www.acquisitionresearch.net

The Rapid Integration and Test Environment: A Process for Achieving Software Test Acceptance

Patrick V. Mack—Commander Patrick V. Mack, US Navy is a graduate of the Naval Postgraduate School with degrees in Computer Science and Operations Research. He is the Principal Assistant Program Manager (PAPM) for the Navy's Maritime C2 Program Office (PMW 150) responsible for the development of the Global Command and Control System (GCCS) Maritime and Navy version of GCCS-Joint programs. An Engineering Duty Officer, he has served five tours at the Space and Naval Warfare Systems Command (SPAWAR): Technical Director for the DoD's Joint Simulation System–Maritime component; Flag Aide for Commander, SPAWARSYSCOM; Deputy for the APM for Naval C2 Systems, Research and Development; and PEO Integrated Warfare Systems (PEO IWS) as the Program Director for the Cooperative Engagement Capability before his current assignment. Other assignments include OIC of SPAWAR Systems Facility Pacific, Yokosuka, Japan, and a one-year tour in Baghdad, Iraq, at the Multi-National Security Transition Command, deputy Chief of Staff for reconstruction, where he was awarded the Bronze Star.

Introduction

The *Rapid Integration and Test Environment (RITE)* initiative, implemented by the Program Executive Office, Command, Control, Communications, Computers and Intelligence, Command and Control Program Office (PMW-150), was born of necessity. Existing processes for requirements definition and management, as well as those for software development, did not consistently deliver high-quality Navy command and control (C2) systems on time and within budget. Navy C2 software programs experienced an increase in software defects that were not discovered until the completion of development activities and, because of the pressure to deploy software on schedule, product releases were distributed with defects. These defects were then repaired post delivery at significant cost. This situation was untenable and required new procedures and processes to solve the programmatic and technical challenges while operating with reduced budgets.

This paper introduces a new life cycle model for Navy C2 software that places increased emphasis on early and frequent software testing, as well as on necessary software engineering practices at the source code level. *RITE* is a more structured approach to software development, taking full advantage of technology advances and open source models to automate processes and shorten development cycles—thus increasing the maintainability of the software baselines. The initiative also clarifies software delivery requirements, adding additional engineering rigor to deliverables and reducing opportunity for misunderstanding between customers and developers. Its goal is to reduce overall cost, streamline delivery of quality C2 software, and, ultimately, resource focus toward the early stages of the life cycle, where the return on investment is maximized. *RITE* provides comprehensive oversight of software development from initial product design to customer acceptance.

RITE has four foundation pillars:

- **Software Development Contracts.** The need to provide detailed system requirement specifications and acquire favorable product licensing agreements.
- **Process improvement.** The adoption of industry software engineering best practices; testing early and often to detect, track and correct software defects while the impact on project cost and schedule is minimal.

- **Infrastructure development.** The establishment of a centralized repository with web interfaces to streamline and automate product testing, information sharing, and end-product distribution.
- **Organizational change.** The alignment of technical skills and staffing levels to support new life cycle processes.

RITE was initially developed within the context of the Maritime Global Command and Control System (GCCS) Family of Systems (FoS) (MGF) project at SPAWAR Systems Center Pacific (SSC Pac). However, it is applicable to a wider range of software development programs. This paper compares and contrasts the *RITE* Life Cycle with current Navy C2 development processes, highlighting program benefits achieved through the new initiative. Also, future implementation activities are presented, along with proposed program metrics and areas for further consideration.

Current Navy C2 Development Model

Total appreciation of the benefits associated with the *RITE* Life Cycle requires an understanding of existing development activities and how they have been adapted under the *RITE* initiative. The Navy C2 release life cycle is a subset of the overarching Department of Defense (DoD) Acquisition System descripted in *DoD Instruction 5000.2* (USD(AT&L), 2008). It takes place within the Engineering and Manufacturing Development (EMD) Phase and follows the Evolutionary Acquisition (EA) model used for rapid acquisition of mature technology by implementing a spiral development approach.

This section describes the current release life cycle and presents limitations inherent in existing processes that prevent effective EA performance.

Current Release Life Cycle

The current life cycle consists of the five stages shown in Figure 1. These stages run serially and are scheduled annually. The percentages associated with each life cycle stage are work-years of the level of effort, and to some extent project timelines, expended during a complete project life cycle. Projects spend a majority of the total ownership cost (TOC) after software development is completed. Because the model produces software components with "audidable" defects, there is a self-perpetuating cycle of allotting little time, or funds, for upfront requirements, design, and development, causing the majority of the budget expenditure in the later release stages on defect detection and fixes. Usually, multiple large scale development tests (DTs) are required, resulting in schedule creep and installation delays. Historically, few programs make it through operational test (OT) with a deficiency-free report.

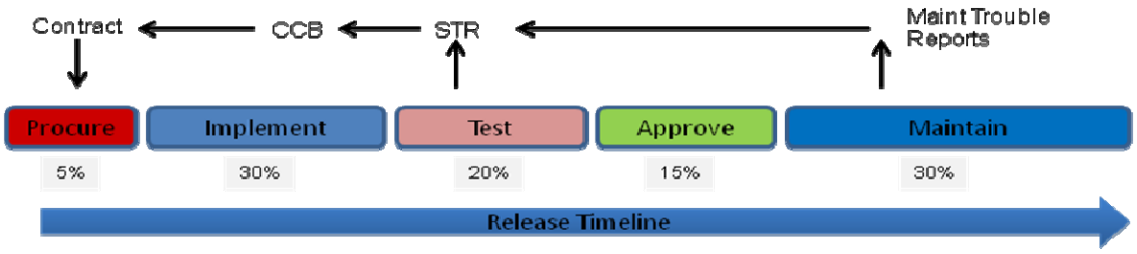


Figure 1. Current Release Life Cycle

Procurement Stage. The product release life cycle begins with the contracting officer's preparation of the contract request for proposal (RFP). The RFP is based on the list of specifications (product features) provided by the Acquisition Program Manager (APM). New specifications are based upon key performance parameters (KPP) that are derived from operational requirements and are influenced by corrective actions required as a result of the software trouble report (STRs) process. New features may be often designed to fix problems discovered either during the previous product testing or as a result of fielded system trouble reports. The final specifications are the result of a trade-off between the prioritized list of specifications and the allotted RDT&E budget, operational schedules, and established product release date. It is important to note that many of the detailed product and software documentation requirements are not clearly established in contract language under this model. The Procurement stage output is the award of an executed contract.

Implementation Stage. After contract award, the developer conducts a modified product design review and develops the software to contractual specifications. There tends to be limited interaction during this stage between the contractor development team and the Government's project team because under the terms of the contract, the contractor has proprietary ownership of the software product and sole responsibility for product delivery. Outputs from this stage are the executable software segment and any contractually required documentation.

Test Stage. The project team accepts delivery and assumes responsibility for the integration and several levels of testing. Software defects discovered during this stage are reported to the Configuration Control Board (CCB) via the STR process, where corrective action (fix) and prioritization decisions are made. By contract, the developer is required to fix critical and high-priority defects (referred to as Priority 1 and 2, respectively) prior to undergoing final testing. Lower priority category defects may be repaired, if time and budget permits. Once the software product successfully passes a final testing, a recommendation is made to support a fielding decision. Exit criteria include a demonstration that the software has matured to an acceptable level of Fleet readiness and the software meets systems integration and interface standards.

Approval Stage. The Approval stage involves many approval steps, including security certification and accreditation (C&A), successful operational test (OT), and the formal release approval. These activities are primarily conducted by outside certification agencies, such as the Commander Operational Test and Evaluation Force (COMOPTEVFOR). The output from this stage is the final fielding decision and the granting of final release approval by the Milestone Decision Authority (MDA).

Maintenance Stage. The final stage includes installation, training and continued maintenance of the C2 system.

Life Cycle Limitations

There are many program limitations inherent in the current life cycle model. In previous efforts to streamline and shorten the development cycle, the Government allowed the software developer to assume too much responsibility for the project's success. There were insufficient checks and balances built into the model to ensure that the Government received a quality product on schedule and within budget. Major limitations are highlighted below and were the drivers for the *RITE* initiative.

Limited Requirements Definition and Detailed Design. The previous life cycle model allowed the selected software developer to assume responsibility for detailed system design. Detailed system requirements should be developed by the Government and specified to the developer as part of the contracting process. Additionally, requirements need to be based upon end user (warfighter) inputs and prioritized to meet the most pressing operational needs. Contracts lacked the detailed design specificity needed to fully define the end product. Developers need to have specifications to build to, and test and evaluation (T&E) personnel need those specifications to test against. A frequently cited study, conducted by the Standish Group in 2000, reported that American companies spent $84 billion for cancelled software projects. Another $192 billion was spent on software projects that significantly exceeded their time and budget estimates. The Standish Group, and other studies, list three top reasons why software projects fail:

- Requirements and specifications are incomplete
- Requirements and specifications change too often
- There is a lack of user input (to requirements)

The cost of cancelled and failed projects is likely to have increased since the initial study but indications are that the reasons for failure have not changed. Under the current release life cycle model, making Navy software programs suffer from all three of these shortfalls.

Insufficient Noncommercial Computer Software Rights. Previous Navy C2 contracts failed to acquire the appropriate rights to the software products, thereby allowing the developer to control the product and, essentially, all future enhancements, citing proprietary ownership. As defined within the *Defense Federal Acquisition Regulation Supplement (DFARS)*, the Government can receive "Unlimited Rights" for noncommercial computer software, including source code, whenever the product is developed solely with Government funding. When Government and industry team in the research and development (R&D) effort, using both Government funds and industry funds, the Government is able to retain "Government Purpose Rights" which still affords the authority to receive, assess, and modify the source code of noncommercial computer software products. The lack of Government control of the software source code prevents the Government from ensuring the quality of the software products. Without the source code, product reviews are conducted at too high of a level to determine the condition of the deliverable. It is too easy for defects to go undetected and even for defects to find their way back into new builds after having been previously repaired. Without the rights to the source code, the true quality of the released product is often not known until receipt of user trouble reports.

Limited Schedule Control. Under the existing life cycle model, the SPM only has direct responsibility for the Test stage activities. All other stages are under external ownership and, from a project viewpoint, are essentially fixed in duration and effort. Even during the Test stage, the SPM has limited control because the level and schedule of the integration and testing conducted is primarily dependent upon the quality of the executable software and associated work products delivered by the developer. The Government needs to have greater involvement in the Implementation stage, working in partnership with the developer, to ensure the quality of the delivered product. Doing so allows the Government to monitor and influence the development schedule and to have better control of the subsequent Test stage.

Insufficient Government Technical Oversight. As stated above, current contracts do not provide rights to the noncommercial computer software source code at the level necessary for effective Government technical oversight. However, just obtaining "Unlimited Rights" will not, in itself, solve the technical oversight shortfall. The historical lack of direct Government responsibility for software development has taken its toll on the quantity and quality of resident software development skill sets. Professionally trained software developers were faced with the career decision to either attain new skill sets or transfer to private industry and write software code. Therefore, for the Navy to provide effective technical oversight and serve as "trusted agents" requires a retooling of its work force. New software engineers will need to be recruited or current staff will need to be trained in new software development techniques and tools.

Reduced Competitive Environment. Lastly, many software development contracts have essentially become sole source contracts. Because incumbent developers have proprietary ownership of the software source code, new contractors attempting to compete for follow-on development contracts basically have to "rewrite" the code because the incumbent is not generally required to relinquish control of their work products. Further, much of the current contract language may not require detailed documentation of the software, making it difficult for anyone other than the original developer to understand or modify the delivered code. These barriers-to-entry greatly reduce the competitive landscape and afford the incumbent a significant competitive advantage over its competition. Also, with little or no true competition, the Government experiences a reduction in pricing power and control over the final contract.

The *RITE* Initiative

The implementation of the *RITE* Life Cycle by PMW 150 represents a dramatic shift in the way the Navy C2 Program Office develops noncommercial computer software. *RITE* provides a software oriented set of engineering standards, processes and guidance, tools, and contract language, all available through a software development, test and distribution infrastructure. It impacts all stages of the life cycle and facilitates Government control of the various stages, reducing project costs and improving schedule performance.

The *RITE* Life Cycle Model

As previously stated, a major goal of *RITE* is to reduce overall cost and streamline delivery of quality Navy C2 software. It promotes an open standards-based culture of modularity and reuse to keep pace with evolving technology. The national security implications of open technology development are clear: increased technological agility for warfighters, more robust and competitive options for program managers, and higher levels of accountability in the defense industrial base. Technologically advanced Navy C2 systems are vital to the warfighter's ability to plan and execute missions. The *RITE* initiative entails a parallel shift in acquisition methodologies and business processes to accelerate the delivery of advanced C2 systems to the operational forces.

This section describes *RITE's* open architecture approach and how information will be accessed, used, reused, applied, distributed, and managed under the new initiative. *RITE* involves changes in organizational structure, processes, strategies, policies, and business practices, including the shifts in traditional Government and contractor software development roles. It provides the necessary guidance to organize, manage, and employ a

distributed, interoperable, and scalable net-centric, collaborative development and distribution environment.

RITE Pillars

The *RITE* initiative is designed around four functional pillars.

RITE Contract. A baseline requirement for *RITE's* implementation is the adoption of specific contract language that changes the existing relationship between the prime software developer and the Government project team. New contracts address the following contract stipulations.

- **Requirement Definition.** The Government assumes responsibility for developing the system requirements and baseline design specifications used by the software developer and the Government project team for contract performance. These requirements are based upon operational requirements and are at a level of specificity that provides developers and testers product acceptance criteria. Requirements definition involves the interaction of all stakeholders early in the Procurement stage. This Government engineering task is a vital part of preparing the contract language to insure the Government gets the desired product from the developer.
- **Licensing Agreement.** The Government obtains either Government Purpose Rights or Unlimited Rights, as defined in *DFARS* and applicable agency supplements, for all noncommercial computer software items developed with Government funding. This includes the delivery of software source code and related software version design documentation.
- **Process Adherence.** The Government mandates that use of the *RITE* Life Cycle be processed through the Statement of Work (SOW). New SOW language includes:
    - Contract Data Requirements Lists (CDRLs) and Data Item Descriptions (DIDs) that define an expanded set of delivered software work products, including source code and software version documentation;
    - Streamlined test processes, requiring the use of automated and focused testing procedures;
    - Contractor Performance Acceptance Reporting System (CPARS) metrics that satisfy *RITE* entrance and exit acceptance criteria;
    - Specified Quality Management (QM) procedures;
    - Specified Configuration Management (CM) to the source code level;
    - Implementation of disaster recovery techniques; and
    - Software auto-installation capability.

RITE Process. The *RITE* Life Cycle is shown in Figure 2. A major change is the coupling of the Implementation and Test stages and the direct involvement of the SSA project team in software development. There are a number of test events (engineering drops) of the software as it is being developed. Similarly, developers integrate *RITE* processes, techniques and tools into their development process. Both stages now take place seamlessly as part of the *RITE* process, aligning early defect detection, tracking and resolution with development activities. The *RITE* Life Cycle includes the implementation of front-end engineering, source code quality management, a distributed development environment, and automated development and test tools. A key assumption of *RITE* is that software development projects will always contain bugs and defects regardless of the skill and diligence of the development team. *RITE* has been designed to mitigate the overall program impact of software problems by the use of early and frequent software assessments.

Also shown in Figure 2 are the adjusted levels of effort (LOE) associated with the each life cycle stage. *RITE* places an increased emphasis on the early stages in an effort to detect and correct errors in the product design and code while the cost to correct is relatively inexpensive. Therefore, the Procurement stage has been expanded to allow for additional upfront Government effort needed for the development of requirement specifications and detailed designs. Additionally, the Implementation and Test stages have merged, signifying closer continuity between the two stages. Frequent testing of incremental software builds, referred to as "engineering drops," during the Implementation stage has been accommodated by an increase in development schedule time. The increase in LOE during the early stages is offset by a reduction in the time needed to complete the formal Test, Approval and Maintenance stages, respectively. Under *RITE*, the software release exits the Implementation stage with fewer defects, thereby reducing the uncertainty, and the project duration, associated with the latter stages. *RITE* improves the overall life cycle process to the extent that TOC is expected to be reduced while the frequency and quality of the product releases increase.

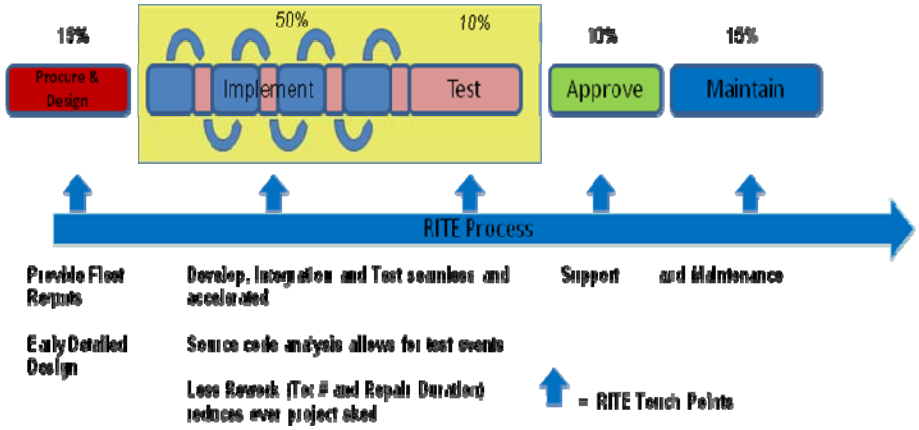


Figure 2. *RITE* Life Cycle Model

Automated and Focused Testing. Testing is critical to the overall development process success and is the means to validate and drive software quality improvement. *RITE*'s testing philosophy is based upon the need for early and frequent testing of software source code. Software development and defect detection activities begin almost simultaneously during the Implementation stage. Therefore, *RITE* mandates additional

incremental testing throughout the Implementation stage, not waiting until the end-of-program to test the product. Quality needs to be built-in (and validated with incremental integration and testing), not tested for at the end.

The RITE concept of testing is based on testing to a level of acceptable risk. Since it is not feasible to test 100% of the software code, available resources, such as funding and personnel, time or urgency, and expertise of the test team, influence the extent of acceptable risk. To minimize the level of risk, the RITE uses automated inspection and test tools wherever possible, thereby increasing test coverage and allowing faster discovery of defects remaining after requirements and design inspection and assessment. The automated tools range from simple scripts to complex commercial tools and exercise the software and identify outstanding defects. Testers use the tools to measure software complexity and assign quality ratings to segments entering the integration process. They also use automated test tools to perform time-consuming repetitive procedures, such as executing a test case multiple times under a variety of conditions, over an extended period of time, or both. Automated tools also are useful to simulate large numbers of users for performance or load testing, or exercise software that does not have a graphical user interface, such as device drivers or software libraries.

Where manual testing is necessary, the *RITE* test team follows a rigorous test methodology that focuses on predetermined test cases derived from real world situations. Testers prepare and execute detailed test procedures that provide clear and concise test steps and expected outcomes.

Testers document test results derived from automated and manual testing in standardized test reports that incorporate quality metrics indicating the level of software maturity (lack of defects). Sponsors use these reports to reduce risk and determine when the software is ready for release.

- Detection and Acceptance Process. *RITE* implements a systematic integration and test process to maximize efficiency in defect detection, thereby accelerating the release of high-quality software products to the Fleet. This systematic approach to testing allows more coverage per unit of test time, leverages automated testing to help identify bugs early, and uses Navy use-cases for test scenarios. Figure 3 illustrates the *RITE* Gated Acceptance and Detection process performed on each delivered incremental software build. This process is a part of the overall *RITE* Process, as shown in Figure 2, and is conducted repeatedly throughout the various life cycle stages to identify product defects and to validate product development milestones. Inputs to the process are the engineering drops that include the executable segments and associated software work products such as Software Version Description (SVD), test plans, test procedures, test reports, and load and installation instructions. Process outputs include a qualified system, system test reports, installation instructions, and training plans. The process gates are described below. Note that these gates are serial. Regardless of gate, whenever a drop fails to pass an inspection or test, team members notify configuration management (CM), who, in turn, notify the development contractor. If resolution of the root cause for the failure requires a software modification, the process starts over.

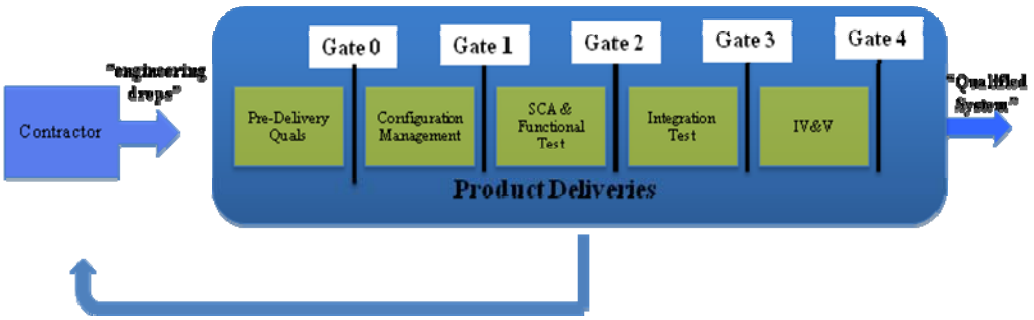


Figure 3. *RITE* Gated Detection and Acceptance Process

- *Gate 0–Pre-Delivery Qualifications*. Prior to the delivery of a new software component from the contractor, candidates are required to meet specified criteria that include contractor conducted pre-delivery testing and the development of test reports.

- *Gate 1–Configuration Management*. The contractor delivers an engineering drop to the *RITE* CM team. The team reviews the delivery to ensure all contractually required work products are present. Upon validation, the CM team delivers the software to the *RITE* Acceptance Test team.

- *Gate 2–Source Code Analysis (SCA) and Functional Testing*. The *RITE* Acceptance Test team schedules an Acceptance Readiness Review during which they check the delivery against the acceptance checklist to ensure the delivery media is readable and that the media and documentation are correct and complete. This process includes checking that all required licenses are present and current, and that test plans, procedures, and installation instructions are included. Following the review, the Acceptance Test team performs an installation test to ensure the segment installs correctly. Automated tools are used to perform an analysis at the source code level. Exit criteria for this phase are a readable and correct segment, correct documentation, a successful installation test, and a quick look test report. The Acceptance Test team notifies CM to deliver the software to the Integration team.

- *Gate 3–Integration with Baseline System*. After CM delivers the software, the Integration team reviews the segment installation procedures and attempts to integrate the segment with other C2 system segments into a complete system or "build." If they can successfully build the complete system, they perform high-level checks to ensure the build starts up correctly and major functionality is present. Exit criterion for this phase is a successful Independent Verification and Validation (IV&V) Test Readiness Review indicating that the system is ready for more in-depth testing.

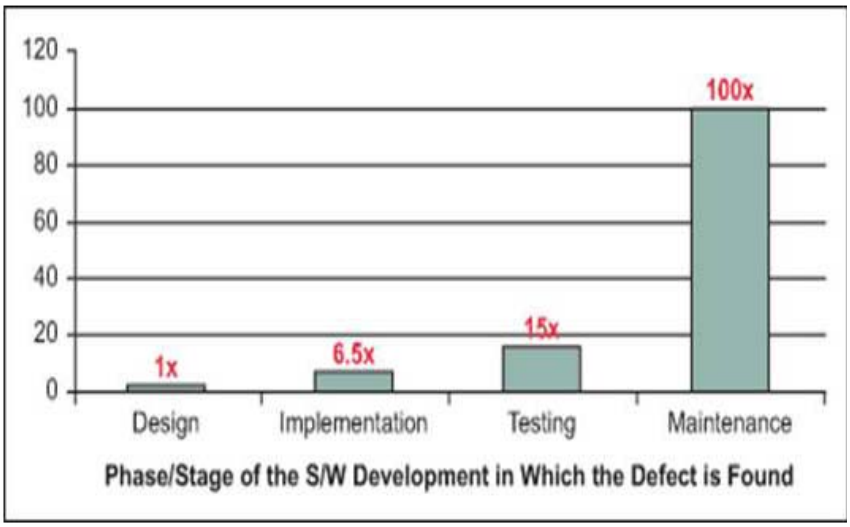
- o *Gate 4–Independent Verification and Validation (IV&V) Test*. The IV&V team develops test plans and procedures covering all types of functional testing. They perform functional testing to verify that the build meets specified requirements and validates that it achieves the desired functionality, and perform interface testing to ensure that the build meets external interface requirements. If both these tests are successful, the IV&V team performs system-level and stress testing in an environment that closely simulates the operational environment. Exit criteria for this phase are a successful IV&V Test Review and delivery of the IV&V Test Report to PMW-150 or other sponsors.

Once PMW-150 accepts the IV&V Test Report, the *RITE* team supports a Developmental Test (DT) by performing laboratory DT testing at the operational site. The *RITE* team provides on-site training to shipboard operators and resolves issues that occur during DT testing. Exit criterion for this phase is a qualified system that is ready to enter the Approval stage.

- **Early Defect Detection.** Since the foundation of the testing process is based upon the early identification of software defects, research supporting this principle is provided in this section. Software defects have different names in different agencies but, for the purposes of this paper, a software defect is any development error, issue, bug, defect or incident.

In an Internet article, Mukesh Soni (2010) states that the "Prevention is better than the cure" adage applies to software defects as well as medicine. Potential software defects detected during the early stages of software development, such as during requirements specification, are easier and cheaper to resolve than during later stages presented in the IBM Systems Science Institute study chart shown in Figure 4. Defects introduced during the requirements and design phase are not only more probable but also more severe and more difficult to remove during later stages of development, test and maintenance. This is because of the increasing number of interfaces and dependencies that exist in the code as well as the time it takes for developers to refresh their knowledge of the specific code being repaired the further removed they are from the original development. Pre-test reviews and inspections are the most efficient way to detect errors in requirements and design.

A result of early detection is the reduction in the number of defects released with delivered systems. This will reduce the need for expensive software maintenance programs and free up future budget dollars for increased RDT&E expenditures. It is a *RITE* goal to demonstrate the ROI associated with the new life cycle processes by validating the improved system performance of fielded systems. A premise is that over time, budget allocations can be adjusted to allocate more money to the early stages (Procure and Implement) where the ROI is the greatest.



Source: IBM Systems Sciences Institute

Figure 4. Relative Cost Required to Fix Errors During Software Development

RITE Infrastructure. A Distributed Development Environment (DDE) is a virtual collaborative environment that spans multiple organizations and/or multiple physical locations. In a DDE, project members share ideas, information and resources, and actively collaborate to achieve a common goal. They may not see each other face to face, but are all working collaboratively toward the project outcome. This may be accomplished through e-mail, the Internet and other forms of long-distance communications. The primary advantage of DDE is availability of resources and access to software development tools from different locations. The objective is to lower development costs, increase productivity, decrease time-to-release, and improve product quality.

Software development in the Navy is transitioning to geographically distributed development environments. Distributed development is one of the highest forms of collaboration in the development environment, but many challenges face project managers responsible for the success of distributed teams. Four characteristics common to many of today's collaborative failures include:

- Cultural incompatibility,
- Leadership struggles,
- Lack of trust, and
- Inbred notions of competition.

RITE DDE strives to overcome collaboration challenges. Success requires understanding relationships and taking practical and affordable actions to achieving successful virtual operations. These include building an organization that supports working in a distributed development, with the right incentive systems that reward collaboration. It requires urbane management and oversight, a highly efficient infrastructure, a well-developed organization, and daily interaction with open communication.

The hub of the *RITE* DDE infrastructure is the Development and Distribution (D2) Center. The D2 Center allows access to, and sharing of, applicable Navy C2 program software, test tools, program governance and guidance documentation and other project

technical documentation generated as part of the *RITE* Life Cycle process. Developers, testers, and other stakeholders have access to the Center through a private cloud using a web-based interface and a set of intuitive tools for locating and extracting desired components and associated work products. The D2 Center provides strong configuration control of the various project artifacts and assures that contractor and Government teams are working from a common set of project components.

Project members, using the RITE D2 Center, have the ability to specify and validate requirements using interactive simulations and a collaborative process that involves all stakeholders. Project members take ownership for achieving the overall project goal. No one can succeed without everyone being successful. Accurately identifying software requirements and effectively managing those requirements throughout the life cycle are keys to reducing rework activity and creating applications that accurately reflect end users' needs.

The infrastructure architecture is shown in Figure 5 and takes advantage of an open architecture to support the following project functions:

- Government management of key project artifacts,
- Management of source code,
- Definition and management of the development and integration environment,
- Configuration Management (CM) for validation and control of software deliveries,
- Support tool development,
- Architecture, and
- Guidance and governance documentation.

RITE D2 products are available on the site for sharing by all stakeholders and improve project communication and coordination while providing a common set of standards and tools for use throughout the project. Examples of how the D2 Center might facilitate software development and distribution include:

- Development. Using the D2 Center, a developer can log into the site and, by reviewing the available service catalog, discover that a new multi-tactical data links capability (MTC) component exists. Coding changes to the software component can then be made and automated acceptance tests can be re-run, all done remotely.
- Distribution. Similarly, Fleet users can access the site to download new Navy C2 components and automatically upgrade and test their systems without requiring the assistance of outside installation teams.

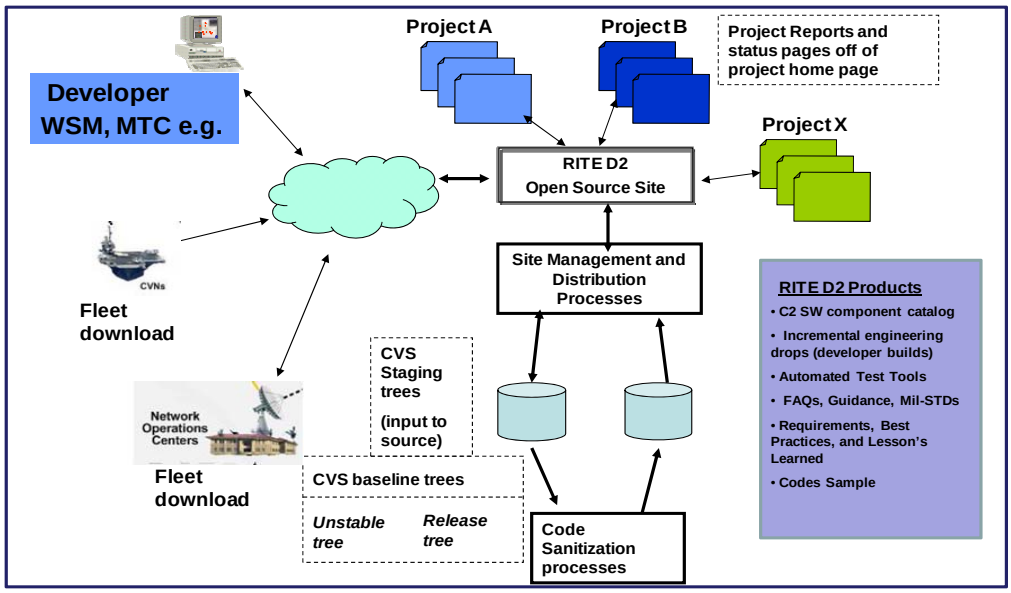


Figure 5. *RITE* Infrastructure Architecture

RITE Organization. Lastly, one of the key components of the *RITE* initiative is the organizational change needed to efficiently and effectively perform within the new life cycle model. Current project structure evolved to support existing processes and personnel skill sets were optimized for the needed job task capabilities. As the model changes, increasing the need for more software engineers and reducing the number of fleet installation teams, the organizational core competencies need to change. The projected changes include:

- Project Manager Performance Measures. New performance metrics are needed for the Government management team. In a distributed work environment where success is dependent upon frequent communication and collaboration, success factors need to reduce the current competitive environment. Additionally, success needs to be measured by program efficiencies and effectiveness that result in budget optimization, not the overall program budget size. Therefore, additional metrics associated with product quality improvement and the ability to meet Fleet operational requirements need to be captured and used for overall performance assessment.
- Personnel Qualifications. As highlighted previously, the Government lacks the number of qualified personnel, either educated or trained in the software engineering disciplines, to perform the job task functions required by RITE. These technical qualifications include knowledge of current operating systems, databases, and functional applications. Of importance are skills associated with open architecture development and web design. The Government needs to begin the transition, selecting a cadre of technically qualified software engineers to lead the workforce shift from current methods and processes while initiating focused recruitment and training programs.
- Organizational Structure. In addition to the personnel qualifications, the project organizational structure needs to evolve to meet the changing life cycle model. Under RITE, the staffing levels associated with software development and testing will need to grow to meet the increased level of effort and product throughput associated with those stages. Conversely, although not immediate, there will

need to be a reduction in staffing associated with installation and integration activities performed during the Maintenance stage. As the ability to remotely control the distribution of new software releases through the D2 Center becomes widely accepted, the need for installation teams is reduced. Therefore, the organization needs to be modified to reflect these changes in staff levels to free up budget dollars for use elsewhere.

Case Study—Multi-Tactical Data Links Capability

Casualty Description

The casualty to the multi-tactical data links capability (MTC) on the USS John C. Stennis (CVN 74) was first reported in May 2008 via the casualty report (CASREP) system. The stated problem was a "channel crash" that prevented the use of XXXX and degraded the Stennis ability to perform in mission areas

Contractor Approach

As a result of the CASREP, the development contractor was tasked with the responsibility of troubleshooting and repairing the problem. Their initial response was to form a technical "fly-away" team and travel to the forward deployed ship location to conduct an onboard investigation. In August 2008, the team boarded the Stennis, while on deployment, and began troubleshooting the reported problems, working alongside shipboard technicians. While onboard, the team did attempt to repair the MTC by reloading the current software release version (v4.5.9.14) but this did not resolve the problem. Upon return to San Diego, the contractor team continued troubleshooting at their facility as part of the future product release version (v4.5.9.15)) but achieved little or no success. By November 2009, unable to isolate and repair the MTC problems, an STR was written to more thoroughly document the problems and provide a current status update. Subsequently, in January 2009, seven months after the problems were initially reported, the contractor again sent a team of system experts to the ship to further investigate the issues. Their actions included the installation of the new MTC software release. Initial indications were that the new release corrected the channel crash but further investigation determined that the problem persisted. In April 2009, the program office called for a review with the contractor to determine a course of action. The decision was made to use the *RITE* process to aid in a timely repair. This effort was implemented in phases to monitor the effectiveness of the approach. Throughout this initial phase, the SSC Pac SSA team had limited government oversight involvement in the software maintenance activities by only tracking activities via the STR process and providing laboratory support, as needed. The initial Phase milestones are outlined in Figure 6.

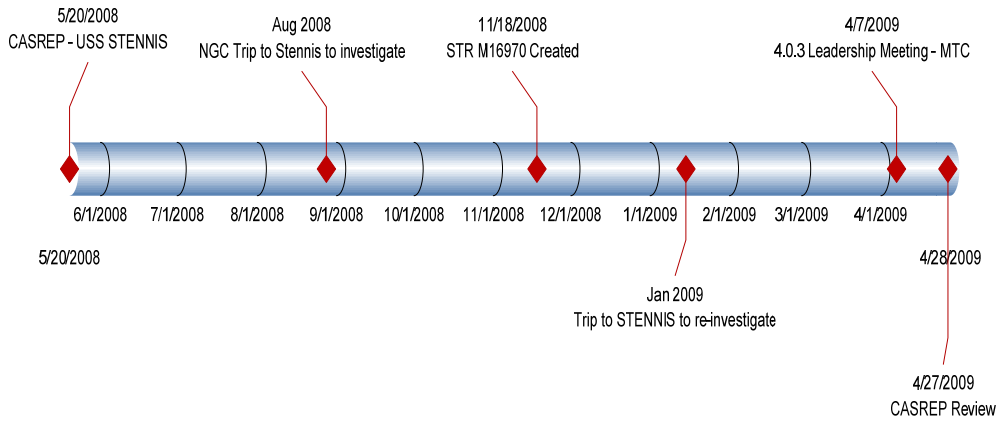


Figure 6. Phase 1 Milestones

In Phase 2, the next level the *RITE* process implementation was instituted as a software repair had not be identified. Phase 2 instituted the Engineering drop process consisting of engineering reviews and MTC assessments conducted by the SSA team for cause and effect. The engineering drop process by itself did not yield a repair. A modification to the existing process instituted a lower level STR investigation by conducting source code analysis using a set of automated source code analysis tools and peer reviews. The analysis identified key potential failure areas within the code.

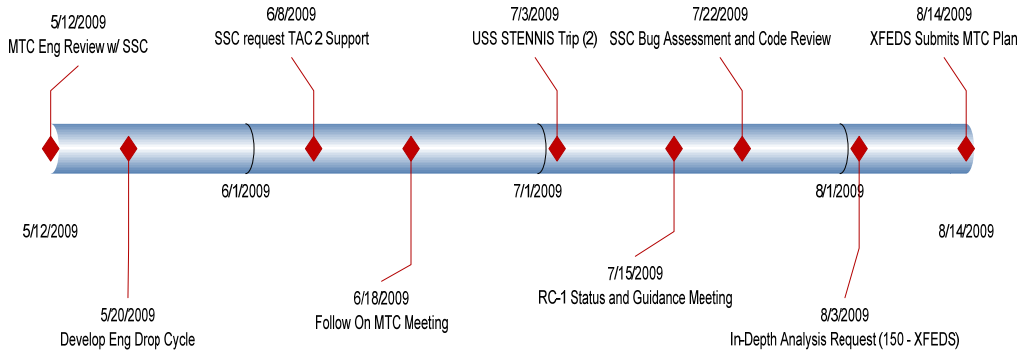


Figure 7. Phase 2 Milestones

The *RITE* Solution

In August 2009, Phase 3 was initiated. The activities undertaken in this phase were the combination of several independent testing process changes previously implemented in segments of the MGF program. Key actions taken to successfully correct the MTC repair included:

- Incorporating the use of automated code assessments, memory usage analysis, and debuggers.
- Establishing a tiger team (2 team: contractor and SSC-Pac), working from a common work list. Responsibilities were divided between the teams with the SSA team primarily responsible for engineering oversight and source code

analysis. The SSA became the source code authority and a repository was established at the government site.

- Implementing one-week build cycles with relevant sections of current code (RC) being distributed from the repository each week. The weekly schedule was fixed with specific functions being performed (and monitored) daily.
- Testing of each incremental build was conducted by the SSA and incorporated into the baseline release version.
- Working from the existing STRs, STRs were reviewed and updated to accurately explain the root causes of general symptoms. No new MTC STRs were to be generated unless new symptoms were observed and were not already covered.

MTC Lessons Learned

By implementing the activities discussed above, the SSA was able to identify, track and repair the causes of the long-standing Stennis MTC problems. Using the *RITE* processes, problems that had lingered for over fifteen months were satisfactorily repaired in less than three. Using an integrated Government/industry team and employing code level assessments, strong configuration control, and diligent development oversight, software code issues that had gone undetected throughout many versions of the software release were repaired. Key lessons learned have been incorporated into the evolving *RITE* initiative and are highlighted below.

- **Delivery code assessments**
  - Established standardized process for acceptance product delivery are accepted
- **Identified STR causes in code**
  - Debugging and defect isolation
  - Recommended fixes passed back to developer as necessary
- **STR fix management and oversight**
  - Was correct thing fixed?
  - Reduce churn with fix attempts
  - Used SECP (Software Engineering Change Proposals)
  - Final fix incorporated code from both the government and contractor team SSC, XFEDS, NGMS

RITE Benefits

Although *RITE* is a relatively new initiative, it is achieving positive results in the Navy's C2 development activities and providing significant benefits to the program office. Benefits include the following:

Budget Multiplier

By allocating time and budget dollars to the earlier stages of software development, the Government is getting "more bang for its buck." As shown in Figure 4, for every dollar

that is spent in the implementation phase, you get more than a 10 times multiplication effect when compared to dollars spent during the maintenance stage. Therefore, *RITE's* early and frequent defect detection activities are ultimately saving overall Navy C2 program dollars for the Government.

Increased Product Features or Reduced Cost

By reducing the TOC of Navy C2 programs, the Navy is theoretically faced with the decision to either reduce its overall program budgets or increase the number of component features assigned to future programs. It is recognized that because the budget categories (RDT&E verses OMN) are distinctly different, the ability to move money between the categories is not as simplistic as the author is suggesting. However, it is believed that over time, reductions in software rework will allow the OMN (maintenance) budgets to be reduced in order to increase RDT&E expenditures. Much like the RDT&E, budgets have shrunk to cover the costs associated with the rework needed for previous systems.

Improved Schedule Performance

The ability to accurately predict Navy C2 software delivery schedules, a weakness under the existing release cycle process, has been improved with *RITE*. Because the *RITE* model is based upon early and frequent interaction between the developer and the Government project team, there are fewer project surprises. The Program Manager has the opportunity to adjust the project execution, including the allocation of additional development staff, the adjustment to key milestone delivery dates, or even the reduction of product features if project issues are discovered early. Also, because *RITE's* automated and focused testing identifies software development problems earlier in the cycle, there is less corrective action required during the later stages, allowing the software development team to more accurately assess the impact of the defects and the time it will take to correct. "Focused testing" allows for testing and retesting of specific problem areas in a surgical precision model and does not require the (re) testing of the total deliverable whenever defects are repaired. This makes it less manpower intensive and therefore less expensive to conduct.

RITE is not only about highlighting issues; it also provides continual status updates to the Program Manager, demonstrating positive tangible results of project successes. The ability to repeatedly integrate and compile the binary code into executable software validates that the system is performing as expected. *RITE*, through its use of automated testing tools, also provides solid development metrics that can be used to track the project progress and process improvement.

Lastly, because the *RITE* process ensures that the software being developed is monitored and corrected throughout the development cycle, when the system enters the Approval stage, including the security accreditation and certification requirements and the operational testing (OT) program, the success rate is expected to be higher, thereby reducing the time and cost of performing this stage.

Improved Product Quality

A major benefit of the *RITE* process is that the released system (end product) is delivered to the warfighter with fewer defects, thereby reducing the need for continual software rework using the trouble reporting process. This is due to the improved testing

processes, as well as the detailed acceptance criteria derived from the design specifications. The Government is able to hold the developers accountable for the quality of the software being delivered and not just for their ability to submit a "deliverable" at a designated delivery date. The importance of delivering a quality product cannot be overstated. Besides the benefit of reducing the costs associated with the product rework, the inconvenience to the user caused by numerous system errors and the loss of credibility and confidence in the delivered product has long-term ramifications to future development programs. Studies have shown that most customers place a higher importance on quality than on timely delivery. As critical operational components of the US Navy, it is paramount that Navy C2 systems perform satisfactorily when released for operational use.

Shortened Release Life Cycle

Navy C2 is able to deliver new functionality to the end user sooner due to the reduction in timely and costly defect repair. One of the major game changers for *RITE* has been the ability to manage and control the software source code by acquiring "unlimited licensing rights" for the Government. This licensing agreement has established a partnership between the software developer and the Government's project team, giving the project team authority to require more frequent submission (drops) of the development product. By implementing a more frequent validation and inspection program, which requires the integration and testing of the software at more frequent intervals, the team has identified potential software defects earlier in the development cycle. Additionally, being able to view the development program down to the source level has allowed the project team to more accurately project program schedule and cost and, ultimately, has led to more realistic schedule development. In many programs, deliverable due dates were arbitrarily set early in the contracting stage and rarely adjusted to accommodate development progress.

Contract Competition

Lastly, and perhaps most importantly, having unlimited rights to the noncommercial computer software source code and the ability to share this code with 3rd party software vendors, greatly improves the Government's ability to implement a true competitive contracting environment, and will ultimately help improve product quality while reducing the overall program cost. The Government has begun to lower the barriers to entry into this market and has reduced the program risk, thereby improving its negotiation position.

2011 And Beyond

Future Implementation Activities

RITE is a dynamic program with much left to accomplish. PMW-150 has taken an aggressive approach to changing its software development life cycle management and is currently reviewing plans for FY 2011 and beyond. Areas for consideration are shown in Table 1 and final decisions will be made as part of the annual budget process.

Table 1. Future Program Considerations

Pillar	*Next Steps*
Contract	▪ ***Detailed Arch and Design specs as part of contract. Include stakeholders in process*** ▪ ***Refine acceptance criteria*** ▪ ***Establish defined stages for deliverables/testing*** ▪ ***Define how to manage large software library and to resolve legacy issues through future maint contracts*** ▪ ***Refine CDRLs/DIDs***
Process	▪ ***Determine correct/applicable perf Metrics for Project*** ▪ ***Determine metrics for "acceptable risk" to assist with test exit criteria*** ▪ ***Establish Performance Scorecards*** ▪ ***Measure respective Stage Level of Effort (LOE)*** ▪ ***Validate cost savings assoc with "rework" reduction*** ▪ ***Improve tool set (dependency tool)*** ▪ ***Focus testing on "what changed" not total build*** ▪ ***Increase number (and fidelity) of operationally based "test cases"***
Infrastructure	▪ ***Implement RITE Conops into MGF POR*** ▪ ***Establish Applications Store as part of D2*** ▪ ***Coordinate/Share with Industry (3rd Party developers)*** ▪ ***Establish partnerships with other testing facilities*** ▪ ***Expand RITE to Team SPAWAR and DISA—provide programmatic support***
Organization	▪ ***Personnel technical qualifications—"Trusted Agent" role requires diff tech skills*** ▪ ***Institutionalize RITE development model for all s/w dev*** ▪ ***Conduct staffing assessment—compare to competency requirements***

Conclusion

It is widely accepted in the software development industry that early detection and repair of software defects is most cost effective. Early detection also contributes to enhanced software quality and better schedule performance. However, to achieve this requires key changes in the way the Navy C2 program office conducts its software development programs. Fundamentally, the relationship between the development contractor and the Government project team needs to change. The Government needs to exercise its oversight responsibility throughout all stages of the life cycle. To achieve the needed changes, PMW-150 has implemented the *RITE* initiative based upon four pillars consisting of contract standards establishing the Government's responsibilities in the development processes; life cycle process changes to implement the automated and focused integration and testing needed to reduce defect rework requirements; infrastructure enhancements required to expand the communication, cooperation, and collaboration amongst all stakeholders in the Navy C2 program; and, lastly, organizational changes to ensure that the Government has the requisite skills to monitor and support the development contractors in the performance of their contractual obligations.

Although additional capture and analysis of *RITE* metrics is needed to fully validate the total program benefits, early indications are that changes implemented with the *RITE* initiative provides the Navy C2 Program Office a potentially significant return on its investment and should be considered for broader Navy software program adoption.

References

USD (AT&L). (2008, December 22). *Operation of the defense acquisition system* (DoD Instruction 5000.2). Washington, DC: Author.

Soni, M. (2010). *Defect prevention: Reducing costs and enhancing quality*. Retrieved from http://software.isixsigma.com/library/content/c060719b.asp

2003 - 2010 Sponsored Research Topics

Acquisition Management

- Acquiring Combat Capability via Public-Private Partnerships (PPPs)
- BCA: Contractor vs. Organic Growth
- Defense Industry Consolidation
- EU-US Defense Industrial Relationships
- Knowledge Value Added (KVA) + Real Options (RO) Applied to Shipyard Planning Processes
- Managing the Services Supply Chain
- MOSA Contracting Implications
- Portfolio Optimization via KVA + RO
- Private Military Sector
- Software Requirements for OA
- Spiral Development
- Strategy for Defense Acquisition Research
- The Software, Hardware Asset Reuse Enterprise (SHARE) repository

Contract Management

- Commodity Sourcing Strategies
- Contracting Government Procurement Functions
- Contractors in 21st-century Combat Zone
- Joint Contingency Contracting
- Model for Optimizing Contingency Contracting, Planning and Execution
- Navy Contract Writing Guide
- Past Performance in Source Selection
- Strategic Contingency Contracting
- Transforming DoD Contract Closeout
- USAF Energy Savings Performance Contracts
- USAF IT Commodity Council
- USMC Contingency Contracting

Financial Management

- Acquisitions via Leasing: MPS case
- Budget Scoring
- Budgeting for Capabilities-based Planning

- Capital Budgeting for the DoD
- Energy Saving Contracts/DoD Mobile Assets
- Financing DoD Budget via PPPs
- Lessons from Private Sector Capital Budgeting for DoD Acquisition Budgeting Reform
- PPPs and Government Financing
- ROI of Information Warfare Systems
- Special Termination Liability in MDAPs
- Strategic Sourcing
- Transaction Cost Economics (TCE) to Improve Cost Estimates

Human Resources

- Indefinite Reenlistment
- Individual Augmentation
- Learning Management Systems
- Moral Conduct Waivers and First-tem Attrition
- Retention
- The Navy's Selective Reenlistment Bonus (SRB) Management System
- Tuition Assistance

Logistics Management

- Analysis of LAV Depot Maintenance
- Army LOG MOD
- ASDS Product Support Analysis
- Cold-chain Logistics
- Contractors Supporting Military Operations
- Diffusion/Variability on Vendor Performance Evaluation
- Evolutionary Acquisition
- Lean Six Sigma to Reduce Costs and Improve Readiness
- Naval Aviation Maintenance and Process Improvement (2)
- Optimizing CIWS Lifecycle Support (LCS)
- Outsourcing the Pearl Harbor MK-48 Intermediate Maintenance Activity
- Pallet Management System
- PBL (4)
- Privatization-NOSL/NAWCI
- RFID (6)

- Risk Analysis for Performance-based Logistics
- R-TOC AEGIS Microwave Power Tubes
- Sense-and-Respond Logistics Network
- Strategic Sourcing

Program Management

- Building Collaborative Capacity
- Business Process Reengineering (BPR) for LCS Mission Module Acquisition
- Collaborative IT Tools Leveraging Competence
- Contractor vs. Organic Support
- Knowledge, Responsibilities and Decision Rights in MDAPs
- KVA Applied to AEGIS and SSDS
- Managing the Service Supply Chain
- Measuring Uncertainty in Earned Value
- Organizational Modeling and Simulation
- Public-Private Partnership
- Terminating Your Own Program
- Utilizing Collaborative and Three-dimensional Imaging Technology

A complete listing and electronic copies of published research are available on our website: www.acquisitionresearch.org

THIS PAGE INTENTIONALLY LEFT BLANK

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY
NAVAL POSTGRADUATE SCHOOL
555 DYER ROAD, INGERSOLL HALL
MONTEREY, CALIFORNIA 93943

www.acquisitionresearch.org