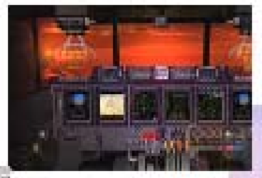
Software Acquisition Management Practical Experience

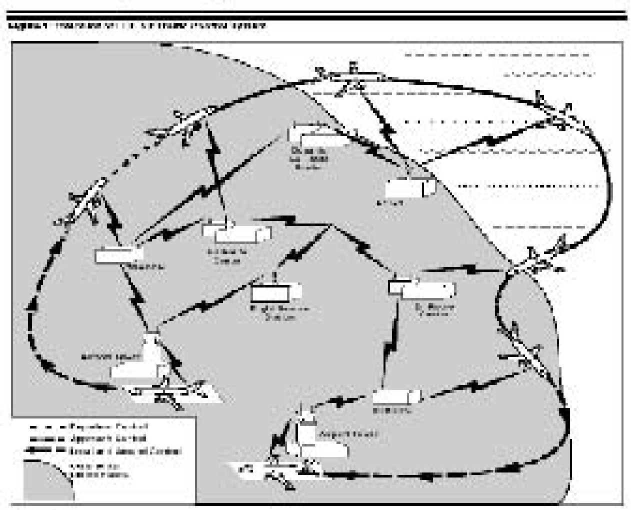
U.S. Air Force C-130 Avionics Modernization Program



Lockheed Martin C-130J Hercules Program

U.S. Department of Transportation Federal Aviation Administration National Airspace System Plan

Terminal Doppler Weather Radar



James E. Jones
Support Systems Associates, Incorporated – Warner Robins, GA 31088

CONTENTS

ABSTRACT

SECTION 1 BACKGROUND
- Air Force C-130 AMP
- LMAS C-130J Hercules Program
- FAA NAS PLAN
  - *Advanced Automation System*
  - *Microwave Landing System*
  - *Radio Control Equipment*
  - *Voice Switching and Control System Upgrade*
  - *Terminal Doppler Weather Radar*

SECTION 2 SOFTWARE ACQUISITION MANAGEMENT CHALLENGES
- *Problems in Software Acquisition Management*
- *Successes in Software Acquisition Management*

SECTION 3 THE CONTRACT
- Contract Types
  - *Fixed-Price Contract*
  - *Cost-Reimbursable Contract*
- Contracting Administration
- Contract Data
  - *Statement of Work/Statement of Objective*
  - *Contract Data Requirements List*
  - *System Specification*
  - *Data Rights*

SECTION 4 SOFTWARE ACQUISITION TEAM

SECTION 5 SOFTWAE DEVELOPMENT ENVIRONMENT
- Acquirer/Supplier Relationship
- Supplier’s Software Process Definition

SECTION 6 TECHNICAL PERFORMANCE ASSESSMENTS
- Software Process Assessments
- Progress Assessments
- Software Product Assessment

SECTION 7 SOFTWARE TEST EVALUATION
- Unit Testing
- Integration Testing
- Formal Qualification Testing
- Problem Reporting/Tracking
- How Much Testing is Enough?

SECTION 8 REQUIREMENTS MANAGEMENT

SECTION 9 RISK MANAGEMENT

SECTION 10 PERFORMANCE MEASUREMENTS
- Software Size
- Cost/Schedule Deviation
- Schedule Progress
- Software Development Progress
- Software Formal Qualification Testing Progress
- Software Requirements Stability
- Computer Resource Utilization
- Software Product Review Item Discrepancies

SECTION 11 SUMMARY

SECTION 12 REFERENCES

SECTION 13 ABOUT THE AUTHOR

ABSTRACT

Studies have shown that for software intensive systems; technical performance, cost, and schedule risks are inherent in delivering quality software products within cost and schedule constraints. Design constraints such as software size and complexity, requirements for high-integrity, reliability, safety-critical performance, and diversity of application domains make proper software acquisition and development extremely critical. However, to ensure that the supplier's performance meets contractual requirements, software acquisition management faces many challenges such as detailing the contractual requirements and selecting the supplier. The acquirer's ability to perform according to the terms of the supplier's contract, monitoring the supplier's progress and performance, and verifying the supplier's compliance with contractual requirements are extremely important. Success in software acquisition and development depends on the following key acquisition elements: (1) the Contract, (2) the Software Acquisition Team, (3) the Software Development Environment, (4) Technical Performance Assessments, (5) Software Test Evaluation, (6) Requirements Management, (7) Risk Management, and (8) Performance Measurements.

This paper provides detailed insight for acquisition organizations that are trying to enhance the effectiveness of their acquisition methods and techniques. It provides a pragmatic discussion of the key acquisition elements involved in software acquisition and development for software intensive systems. For each key acquisition element, software acquisition management and development best practices such as techniques, methods, processes, and activities are discussed. Discussions include practical experience supporting the United States Department of the Air Force's multi-billion-dollar development program for the C-130 Avionics Modernization Program (AMP) modernization of the C-130 fleet and the United States Department of Transportation Federal Aviation Administration's (FAA) multi-billion-dollar development program for the National Airspace System Plan (NAS Plan) Modernization Program. Software supplier acquisition management practical experience for the Lockheed Martin Aeronautical System's (LMAS) C-130J Hercules Program is also described. This paper also illustrates how software acquisition support helps these acquisition organizations achieve their objectives and advance the practice of software engineering.

This paper is organized into the following sections:

- Section 1, "*Background*" provides a brief description of the Air Force's C-130 AMP, LMAS' C-130J Hercules and the FAA's NAS Plan acquisition programs. Examples of the FAA NAS Plan programs are described. A brief description of the programs and acquisition highlights are discussed.
- Section 2, "*Software Acquisition Management Challenges*" describes the problems and successes in software acquisition and provides examples.
- Section 3, "*The Contract*" discusses contracting terms, contract types, and contract data. The essential elements are discussed: Statement of Work (SOW)/Statement of Objectives (SOO), Contract Data Requirements List (CDRL), System Specification, and Data Rights.
- Section 4, "*Software Acquisition Team*" describes the acquirer's software acquisition organization and qualifications.
- Section 5, "*Software Development Environment*" describes the acquirer/supplier relationship and the supplier's defined software process – software standards, procedures, tools, and methods used to develop the software product.
- Section 6, "*Technical Performance Assessments*" describes the acquirer's technical performance assessment activities to ensure that the software products are delivered within cost and schedule in accordance with the contractual requirements and the supplier's defined software process and

plans. It describes the methods and techniques for the assessment of the supplier's software process, progress, and software products (CDRL items).

- Section 7, "*Software Test Evaluation*" describes the acquirer's and supplier's activities that are necessary to ensure that the "as-built" software product meets the software requirements as specified in the software requirements specification document and the related interface requirements specification document. Each level of software testing, problems reporting and tracking, and the level of sufficient testing are described.
- Section 8, "*Requirements Management*" describes the methods and techniques for establishing and maintaining bidirectional traceability of requirements to ensure that the appropriate software product is being built.
- Section 9, "*Risk Management*" describes risk management activities for identifying, analyzing, planning, tracking, and controlling risks.
- Section 10, "*Performance Measurements*" describes the measurements used to provide detailed insight into four key acquisition areas: process, product, project, and productivity.

BACKGROUND

This section provides a brief background for the Air Force C-130 AMP, LMAS C-130J Hercules Program, and the FAA NAS Plan Modernization Program. It describes the program acquisition, provides a brief program description, and discusses the current progress. For the FAA NAS Plan Modernization Program, examples of projects are described.

AIR FORCE C-130 AMP

On 30 July 2001, the Department of the Air Force, Air Force Materiel Command (AFMC) Aeronautical Systems Center (ASC) at Wright-Patterson Air Force Base, Ohio, selected The Boeing Company over arch-competitors Raytheon, Lockheed Martin, and BAE Systems PLC to perform the C-130 AMP. The contract had a total potential value of approximately $4 billion for the Engineering, Manufacturing, and Development (EMD) Program and the Production Program. Under the $485,000,000 EMD Program Cost-Plus-Award Fee contract (F33657-01-C-0047), Boeing was tasked with design, development, integration, test, fabrication and installation of a modern, common cockpit and new avionics systems for approximately 500 C-130 aircraft. The modified C-130 aircraft features a cockpit with six digital displays, a proven flight management system from commercial aircraft, and avionics systems which provide navigation, safety and communications improvements to meet Communications Navigation Surveillance/Air Traffic Management (CNS/ATM) requirements. The CNS/ATM upgrade will allow the C-130 to be continued deployed worldwide.

The acquisition strategy was developed jointly by ASC 866th Aeronautical Systems Group (ASG) and the Warner Robins Air Logistics Center (WR-ALC) 330th Aircraft Sustainment Group (ASG) at Robins Air Force Base, Georgia. The C-130 AMP is managed by the ASC 866th ASG.

The C-130 AMP was officially designated as an Acquisition Category (ACAT)-ID program.[1] After contract award, Congress reduced the Fiscal Year (FY) 02 program by $20 million based on a late contract award date. On 20 August 2003, Boeing was awarded a $200,023,337 Cost-Plus Award-Fee contract modification to provide funding for the Restructure Engineering Change Proposal (ECP) 1302 for the C-130 AMP. The ECP re-baselined the program due to funding reductions in FYs 03/04 which resulted in a two year delay in the System Development and Demonstration program.[2] On February 7, 2003, ECP 0303 was authorized to accelerate the development and fielding activities for the Special Operations Forces (SOF) MC-130H aircraft. The goal was to complete testing and the first AMP MC-130H production unit in Fiscal Year (FY) 2008.

On September 20, 2006, the first C-130 AMP, C-130H2 aircraft (89-09101) with the Combat Delivery/Tanker Capability Block Operational Flight Program (OFP) Software, successfully completed the first C-130 AMP flight from Lackland Air Force Base in San Antonio, Texas. The test flight lasted approximately 3 hours. Boeing delivered the C-130H2 aircraft to the Air Force Flight Test Center at Edwards Air Force Base, California, in November 2006. Ground and Flight Testing was conducted at Edwards, and operational testing will begin in 2009.

---

[1] An acquisition category I program is defined as a major defense acquisition program with estimated expenditures of over $355 million in research, development, test, and evaluation, or over $2.135 billion in procurement (in fiscal year 1996 dollars). A category ID program is monitored by the defense acquisition executive, not a service executive.

[2] Department of the Air Force Research, Development, Test and Evaluation Budget Item Justification Sheet, 041115F C-130 Airlift Squadrons, Project 4726/4885, February 2006, Exhibit R-2.

On January 10, 2007, Air Force officials told *Dow Jones Newswires* that the Air Force instructed Boeing to stop work on the Air Force special operations Hercules. Under the 180-day stop work, more than 100 planes have been taken out of the upgrade plan. On January 12, 2007, the *Defense Daily* reported that "the Air Force's multi-billion-dollar program to modernize the cockpits of its older C-130 Hercules transport aircraft faces unit-cost growth more than 25 percent above the current program baseline that will breach congressionally imposed cost thresholds." Service officials said that the Air Force is preparing to support the Office of the Secretary of Defense in a review to certify to the Congress that this project, the Boeing-led C-130 AMP, merits continuation despite the anticipated increase of its current $4.9 billion baseline. The program recently notified Congress of a critical Nunn-McCurdy breach concerning its unit cost increases.[3] On June 6, 2007, Boeing reported that the U.S. Air Force C-130 AMP has been recertified by the U.S. Department of Defense to continue upgrading 222 C-130 aircraft.[4]

LMAS C-130J HERCULES PROGRAM

In September 1992, the Lockheed Aeronautical Systems Company (now Lockheed Martin Aeronautical System -LMAS) started with the Lockheed 382J, a commercial aircraft that was created specifically to achieve FAA Order 8110.4A Type Certification [FAA 1995]. FAA Type Certification was at Level A (the highest level) of the RTCA/DO-178B standard [RTCA 1992]. The C-130J aircraft is an integrated collection of software systems produced by more that 25 suppliers. These systems, which are developed in compliance with the LMAS C-130J Tier I Software Development Plan, are integrated with the devices on the aircraft such as the engines, pneumatics, flight station displays, and radar.

In late 1994, LMAS received the launch order for the C-130J from the United Kingdom (UK) Ministry of Defense for the Royal Air Force (RAF), who ordered 25 aircraft. The C-130J aircraft was a LMAS initiated improvement for the C-130H3. The Department of Defense (DoD) created a C-130J aircraft acquisition program to provide the Air Force oversight of aircraft development. Eventually, the Air Force procured the C-130J under a commercial acquisition strategy. In October 1995, the Air Force contracted for the first two C-130J aircraft in a modification to the C-130H aircraft contract. The Air Force designated the C-130J acquisition as an ACAT IC Program. The Air Force contracted for the aircraft under a commercial acquisition strategy based on claims by LMAS that the new C-130J was a commercially-designed and available aircraft. LMAS reported only minor modifications were needed to bring the aircraft up to military specifications. LMAS originally planned to deliver the initial aircraft in July 1997, but did not deliver the aircraft until February 1999.

First flight of the C-130J was in April 1996 with a minimum of onboard OFP Software. Delivery came approximately two years later than expected on August 24, 1998, when the RAF became the first customer for the advanced C-130J to replace the C-130K model originally bought in the 1960s. The aircraft was assigned first to the Defence Evaluation and Research Agency (DERA) for an initial test program before being transferred to the Royal Air Force. LMAS, along with customer pilots from the Royal Air Force, the Royal Australian Air Force, and the U.S. Air Force, conducted nearly 5,000-hour flight test program of the new C-130J for FAA certification. The DERA flight test program tested the new system in RAF-specific operational scenarios.

[3] 10 U.S.C. § 2433 establishes the requirement for unit cost reports if certain thresholds for program costs are exceeded (known as unit cost or Nunn-McCurdy breaches). DoD is required to report to Congress and, if applicable, certify the program to Congress.

[4] Boeing News Release: http://www.boeing.com/news/releases/2007/q2/070606d_nr.html

In January 1999, the Air Force became aware that LMAS could not build a C-130J that met its advertised capabilities. Instead they agreed to a contractor-initiated, three-phase, block upgrade program, consisting of block upgrades 5.1, 5.2, and 5.3. However, the Air Force continued to contract for additional aircraft and exercised options for more aircraft before the first aircraft was delivered. The first two C-130J aircraft arrived two years late due to design and testing problems.

In September 1999, LMAS was granted FAA type certification of the commercial variant of the C-130J-30 configuration formally known as the 382J. Concurrent flight tests were accomplished for five other configurations required by initial customers. The flight test program for FAA certification and customer qualification called for performance of more than 30,000 test points covering flying qualities, avionics performance, system reliability and functionality, and safety systems.

The C-130J flew with a complete mission computer software suite in March 2001. On October 16, 2006, the Air Force Air Mobility Command declared Initial Operational Capability for the C-130J.[5]

FAA NAS PLAN

In January 1982, the NAS Plan, now known as the *Capital Investment Plan (CIP)*, was released by the FAA to modernize the facilities and equipment that make up the air traffic control (ATC) system for improvement in capacity, safety, and timeliness through the use of new technology. The ATC permits air traffic controllers to view key information, such as aircraft location, aircraft flight plans, and prevailing weather conditions and to communicate with pilots by providing automated information processing and display, communication, navigation, surveillance, and weather resources. These resources reside at, or are associated with, several ATC facilities: flight service stations, air traffic control towers, terminal radar approach control (TRACON) facilities, and air route traffic control centers (en route centers).

The NAS Plan is a multi-billion-dollar investment comprising over 200 separate programs. Between 1982 and 1998, Congress appropriated over $25 billion (GAO/T-RCED/AIMD-98-93, February 26, 1998). The expenditures were as follows: 1) $5.3 billion on 81 completed programs, 2) $15.7 billion on about 130 ongoing programs, 3) $2.6 billion on programs that have been cancelled or restructured, and 4) $1.6 billion on personnel-related expenses associated with system acquisition.

In 2004, the General Accounting Office (GAO) reported that since 1982, the FAA's ATC modernization programs have consistently experienced cost, schedule, and performance problems that GAO and others have attributed to systemic management issues. Initially, the FAA estimated that its ATC modernization efforts would cost $12 billion and could be completed over 10 years. As of October 30, 2003, two decades and $35 billion later, the FAA expects to need another $16 billion through 2007 to complete key projects, for a total cost of $51 billion [GAO-04-227T (www.gao.gov/cgi-bin/getrpt?GAO-04-227T)].

As of 2005, the GAO reported that the FAA has made progress, but continues to face challenges in acquiring major Air Traffic Control Systems. The GAO found that the FAA's performance-based Air Traffic Organization (ATO), created in February 2004 to address legacy challenges, and had met its acquisition goal for fiscal year 2004. However, the GAO reported that 13 of the 16 major system acquisitions experienced cost, schedule, and/or performance shortfalls when assessed against their

[5] All timeline information is from: Department of Defense: Office of the Inspector General. Acquisition: Contracting for and Performance of the C-130J Aircraft (D-2004-102), July 23, 2004. Available online at: http://www.dodig.osd.mil/audit/reports/fy04/04-102.pdf

original milestones [GAO-05-331 www.gao.gov/cgi-bin/getrpt?GAO-05-331]. In March 2007, the FAA's 2007 2nd Quarter Performance Report indicated for the Organizational Excellence category that the Performance Targets: ***Critical Acquisitions on Budget*** and ***Critical Acquisitions on Schedule*** index range were GREEN.
http://www.faa.gov/about/plans_reports/Performance/quarter_scorecard/media/FPP_Scorecard.pdf

Examples of the FAA's NAS Plan Modernization Programs are described:

Advanced Automation System (AAS) – In 1984, the IBM Federal Systems Company and Hughes Aircraft Company were selected as finalists for the $276.7 million competitive design phase contract to replace computer hardware and software at all three air traffic control facilities – airport towers, terminal facilities, and en-route centers.

After three years and $500 million spent on prototypes, in July 1988, the FAA awarded an acquisition phase fixed-cost contract of $3.5 billion to IBM. The program cost, including supporting efforts, was estimated by the FAA to be $4.8 billion. In 1994, the FAA estimated that the program would cost $7 billion, with key segments as much as eight years behind schedule. The main causes of development failure were reported to be (1) overambitious plans, (2) poor oversight of software development, (3) the FAA's inability to stabilize requirements, and (4) a poor statement of work in the original contract [US DOT, 1998] [OIG 1998].

Microwave Landing System (MLS) - The FAA awarded the $90.6 million first production contract for 178 of the 1250 planned microwave landing systems to the Hazeltine Corporation in Commack, New York, in January 1984 to begin producing a radically advanced type of landing aid that will enable planes to fly a wide variety of approach paths to airport runways. Hazeltine Corporation had promised it would begin delivering the MLS 18 months after contract award. The company ran into problems with software; however, and the delivery date was pushed back repeatedly. Four years later, Hazeltine had only delivered two systems, and the FAA terminated the contract in 1989. The two Hazeltine systems are currently being used only for testing at the FAA's Technical Center in Atlantic City.

Radio Control Equipment (RCE) – On August 7, 1986, the FAA awarded American Telephone and Telegraph Company (AT&T) – Federal Systems Advanced Technologies of Greensboro, North Carolina, the Radio Control Equipment contract (DTFA01-86-C-00034). The schedule specified *Commence First Article Testing* 17 Months-After-Contract (MAC) award (Jan 1988) and *Complete First Article Testing* 21 MAC (June 1988). On September 28, 1989, bilateral Modification 21 was issued to restructure the contract (incorporate a revised specification, schedule and CLIN structure, update Section I clauses, and establish a ceiling price of $105, 286, 000 which includes all negotiated equitable adjustments resulting from changed conditions and the firm items). On March 29, 1990, bilateral Modification 22 was issued to further extend the delivery schedule. In July 1990, *First Article Testing* was started, but was suspended approximately ten days later because of the unacceptably high failure rate being experienced.

During February and March 1991, an extensive audit of the RCE program past, present, and future was conducted. The audit findings indicated that inadequate systems engineering was the root cause of project failure; that AT&T's project organization is inadequate to support a "systems engineering-driven" solution to the problem; and that the recovery effort proposed by AT&T is considered to be high-risk, unrealistic and unmanageable with respect to both schedule and technical accomplishments. The findings and conclusions of this audit were presented to AT&T on March 14, 1991. In May 1991, the FAA terminated the contract for default pursuant to Paragraph 9a (1) (ii) of the Default clause (FAR 52.249-8), which was incorporated by reference in the contract.

Voice Switching and Control System (VSCS) Upgrade - In 1991, the FAA awarded Harris Corporation of Melbourne, Florida, a $1.3 billion contract to develop and install VSCS. The VSCS allows air traffic controllers to establish all air-to-ground and ground-to-ground communications with pilots and other air traffic controllers at 23 commercial Air Route Traffic Control Centers (ARTCCs) in the U.S.

The VSCS is based on independent distributed processors and switches, fault-tolerant databases, redundant high-speed bus interconnections, and extensive switching for real-time reconfiguration and redundancy to achieve an operational availability of 0.9999999.

On November 5, 2001, Harris announced the successful completion of the Functional Acceptance Test (FAT). Currently, VSCS is fully operational in all 21 air traffic control centers around the country, as well as the testing center in New Jersey and the training facility in Oklahoma.

Harris achieved 100% on-time system delivery, installation, test, and acceptance of all systems. Harris received the FAA Contractor of the Year Award and the Human Factors Engineering Society award for excellence in human-machine interface design.

Terminal Doppler Weather Radar (TDWR) - In November 1988, the FAA awarded a Firm Fixed Price Incentive (FFPI) contract (DTFA01-89-C-00002) (GAO/RCED-99-25, FAA's Modernization Program) to the Raytheon Systems Company to develop, produce, and install 47 TDWR systems at 45 airport sites. The Final Acceptance of the First System Testing was scheduled for August 1993. The Incentive Target Date at Memphis Airport (MEM) was scheduled for February 1993.

The TDWR detects and reports hazardous weather in and around airport terminal approach and departure zones. The TDWR identifies and warns air traffic controllers (ATCs) of low altitude wind shear hazards caused by micro-bursts and their associated gust fronts, in addition to reporting on precipitation intensities and providing advanced warning of wind shifts. The ATCs use the TDWR reports to warn pilots who are potentially affected by the hazardous weather patterns.

The First Production System was delivered at Memphis, TN (MEM) six months early. The TDWR is currently installed in 47 areas in the United States – currently operational at 45 airports per *Aviation Week & Space Technology*, January 27, 1992, "*TDWR Installation Begins, Sizable Fuel Saving Expected*".

Raytheon Electronic Systems received the IEEE Computer Society Award for outstanding achievement in improving system processes. In 1991, Raytheon's software process was evaluated at Level 3 against the Carnegie Mellon University Software Engineering Institute (SEI) Capability Maturity Model for Software Version 1.1. It was identified that the TDWR software development played a key role.

Carnegie Mellon University Software Engineering Institute (SEI) reported in 1995 that Raytheon Electronic Systems had implemented a process improvement program in 1988 which had reduced its rework costs from about 40 percent to about 10 percent of the total project cost, increased staff productivity by 170 percent, and reduced defects by about 75 percent over a seven-year period [Haley 1995].

The key acquisition elements of the TDWR are documented by the author in:

- *Successful Acquisition of FAA Terminal Doppler Weather Radar* [Jones 2004]
- *Software Metrics Effectiveness in Software Acquisition* [Jones 1993]
- *Software Acquisition Management: Managing The Acquisition of Terminal Doppler Weather Radar (TDWR) System Software Design* [Jones 1990]

- *Software Acquisition Management: Managing the Acquisition of Computer Software Using DOD-STD-2167A* [Jones 1990]

SOFTWARE ACQUISITION MANAGEMENT CHALLENGES

Software acquisition management is the process to ensure that the supplier's performance meets contractual requirements and that the acquirer performs according to the terms of the supplier contract and their defined software process. Effective management of software acquisition and development is unquestionably one of the greatest challenges in the application of new technologies. Design constraints such as software size and complexity, the requirements for high-integrity, reliability, safety-critical performance, and diversity in systems applications make proper software acquisition extremely critical.

PROBLEMS IN SOFTWARE ACQUISITION MANAGEMENT

The history of software intensive systems acquisition and development has been plagued with technical performance, cost, and schedule problems. Studies have shown that for software intensive systems; technical performance, cost, and schedule risks are inherent in programs tasked with delivering high-quality, highly-reliable software products within cost and schedule constraints [GAO 1999]. Studies have shown that one-half of all software projects double their original cost estimates, projects slip an average of 36 months and one-third of all software projects are even canceled before any products are delivered.

Table 1 depicts some examples of problems in software acquisition management with programs the author experienced. As shown in Table 1, three **FAA NAS Plan** programs were terminated.

Table 1: Examples of Software Acquisition Management Issues

Air Force			
C-130 AMP	Aircraft cockpit modernization	Source Line of Code (SLOC) increased from 60K to 900K	Cost overruns Cost more than 50% higher that initial estimates Cost breached Nunn-McCurdy provision
LMAS			
C-130J	Aircraft cockpit modernization	SLOC increased from 489K at Critical Design Review (CDR) to 761K at Test Readiness Review (TRR)	Cost and Schedule overruns - cost per aircraft increased 32.6% Performance issues-flight safety tests
FAA NAS Plan			
AAS	Advance Automation System		Cost and Schedule overruns Restructured in 1994 after estimated contract tripled from $2.5 billion to $7.6 billion
NADIN II	National Airspace Data Interchange Network		Cost and Schedule overruns
MCC	Maintenance Control Center		Termination for Convenience
MLS	Microwave Landing System	SLOC increased 16K to 70K at TRR	Termination for Default
RCE	Radio Control Equipment	SLOC increased 30K to 175K	Termination for Default

As shown in Table 1, cost overrun is the single biggest problem in software development because it represents time, money and missed opportunities.

For the **Air Force C-130 AMP**, it was reported that "The government and industry both underestimated the complexity of the technology insertion, "said The Honorable Sue Payton-Assistant Secretary of the Air Force for Acquisition in explaining the AMP cost increases and the recent breach of the Nunn-McCurdy cost-monitoring thresholds that Congress used to gauge the health of major weapons programs (Defense Daily, Jan. 12 and Jan. 16). For example, she said, use of commercial-off-the-shelf technologies to replace the navigator proved more difficult than anticipated. "We thought we were going to have somewhere around 60,000 source lines of code," she said. "And as we got into this, for various reasons, not only for the navigation area, we ended up with more like 900,000 source lines of code."

For the **LMAS C-130J**, the Air Vehicle avionics systems (basic aircraft: 41 Line Replaceable Units) source lines of code (SLOC) increased by 56 percent from 489, 000 at the Critical Design Review (CDR) in July 1994 to 761, 000 in April 1996. The Mission Computer and Bus Interface Unit increased 80 percent from February 1995 (104, 000 SLOC) to September 1998 (190,000 SLOC) [Jones 1999].

The **FAA NAS Plan** programs also experienced major software size growth as shown in Table 1 for the MLS and RCE programs.

The difficulty in estimating costs is due to poor software size estimates and requirements growth. Poor software size estimation is one of the main reasons major programs ultimately fail. Software size is the critical factor in determining cost, schedule, and effort [Jones 2004] [Jones 1999]. Software sizing is typically driven by the supplier's agreement items (such as contract vehicle, statement of work, deliverables, and technical requirements) and the supplier's software development capability/maturity.

SUCCESSES IN SOFTWARE ACQUISITION MANAGEMENT

Table 2 depicts examples of successes in software acquisition management. The success of the TDWR was due to the match of the FAA TWDR System Program Office (SPO) software acquisition team and Raytheon TDWR software development team as far as their process capability/maturity, and level of experience. Communications also played a key role.

Table 2: Examples of Successes in Software Acquisition Management

FAA NAS Plan		
TDWR	Terminal Doppler Weather Radar	Delivered First Product Unit six months early Received IEEE Computer Society award SLOC increased 175K at CDR to 200K at Product Baseline
VSCS Upgrade	Voice Switching and Control System	Production completed 100% on-time system delivery FAA Contractor of the Year Award

According to Thomas J. Haley, manager of Raytheon's Software Engineering Laboratory and chairman of its Software Engineering Process Group (SEPG), "Software development played a key role in achieving TDWR delivery to the FAA six months ahead of schedule." In 1991, Raytheon's software process was evaluated at Software Engineering Institute (SEI™) Capability Maturity Model® (CMM®) Level 3. In 1995, Raytheon Electronics Systems received the Institute of Electrical and Electronics Engineers (IEEE) Computer Society Award for outstanding achievement in improving software processes.

The author, FAA TDWR SPO Software Lead, documented the key successful acquisition element in *Successful Acquisition of FAA Terminal Doppler Weather Radar* [Jones 2004] and *Software Acquisition Management: Managing The Acquisition of Computer Software Using DOD-STD-2167A* [Jones 1990].

Software acquisition management methods and techniques can be used to ensure compliance with techniques, control, and processes. Software acquisition management methods and techniques can also be used to verify software quality [Jones 2004-1] [Jones 1992]. The quality of any software product is the direct result of acquisition and development management techniques, controls, processes, and tools. Techniques, controls, and processes can be managed, measured, and progressively improved. All too often, software intensive systems acquirers place the blame for poor quality software on the supplier. Acquirers and suppliers are on different sides of the same system. They can engage in mutually beneficial behaviors or naturally destructive behaviors. A poor acquirer can inhibit a good supplier. History has shown that many of the problems are caused by the mismatch between the acquirer and the supplier as far as their process capability and maturity as well as their level of experience.

THE CONTRACT

Acquisition management involves obtaining products through a contractual agreement. A contract is a mutually binding legal relationship obligating the seller (supplier) to furnish supplies or services and the buyer (acquirer) to pay for them. The acquirer specifies what the system requires, when the software will be needed, and how it will be accepted. The supplier (developer) determines how the software will be developed and the resources required (people, equipment, facilities, technology, and so on). Although both parties are concerned with cost, schedule, and technical performance, each addresses these concerns differently. Acquiring software intensive systems requires that both the acquirer and the supplier to formulate effective management strategies.

This section defines contract types, contracting administration, and discusses contract data – Statement of Work (SOW)/Statement of Objective (SOO), Contract Data Requirements List (CDRL) items, System Specification, and Data Rights.

CONTRACT TYPES

The degree of interaction between the acquirer and supplier depends on the nature of the development effort and the type of contract. Although there are many variations, the two basic compensation schemes used in contracts are *fixed-price* and *cost-reimbursement*. Under a fixed-price contract, the acquirer pays the supplier a fixed sum for the agreed upon products or services. Using a fixed-price contract, the supplier assumes the risk. Profit is a direct function of the supplier's ability to deliver an acceptable product for less than the price paid. Under a cost-reimbursement contract, the risk is shared because the acquirer agrees to reimburse the supplier's allowable costs plus profit. Examples are:

Fixed- Price Contract

There are four basic types of fixed-price contracts: *firm fixed-price* (FFP), *fixed-priced with price adjustment, fixed-price incentive* (FFPI), and *fixed-price redetermination* (FPR). There are strengths and weaknesses of these four types of contracts. For example, a FFP contract requires firm design/requirements and that adequate competition exists. A FPR contract should be used when a realistic price cannot be estimated at start. Under a FFPI contract, the acquirer pays the developers a fixed sum plus an incentive for fulfilling provisions of the contract.

Cost-Reimbursable Contract

There are also four basic types of cost-reimbursable contracts: *cost and cost-sharing, cost plus incentive fee, cost plus award fee* (CPAF), and *cost plus fixed fee* (CPFF). Each of these types of contracts has inherent strengths and weaknesses in their applicability, essential elements, cost risk, and approval requirements. Cost-reimbursement contracts are to the supplier's advantage because they always reimburse the allowable costs. Therefore, the only risk is in the fee, except where the fee is fixed (CPFF contracts). This form of contract is less attractive to the acquirer. The management burdens are higher because allowable progress, costs and fees must be assessed or determined.

For example, the CPAF contract extends the concept of financial incentive into more subjective areas. The acquirer establishes a number of performance criteria that are difficult to measure quantitatively (such as quality, ease of use, etc). The acquirer and supplier structure incentives based upon subjective evaluation of performance using these factors. The fee structure is then established so that there is a base fee and an award amount. The base fee is usually fixed and does not vary as a function of performance. The award fee is used to motivate the supplier to excel in the negotiated areas using the negotiated criteria for performance. The award amount is determined by the award fee board that assesses the supplier's performance relative to the established criteria.

CONTRACTING ADMINISTRATION

Contractual authority is delegated to a contracting officer (CO)/procuring contracting officer (PCO) or the contracting officer's representative (COR). When a change is required to the contract, a Change Order or a Contract Modification is issued by the CO.

CONTRACT DATA

To provide a proper and effective software management environment, appropriate management requirements must be communicated to the supplier. The contract vehicle must be designed to clearly express a vision of final product goals and the development effort requirements. Issues important to managing a software acquisition project should be addressed in the Request-For-Proposal (RFP). Thus, the development of the RFP is the acquirer's first step toward bringing the acquirer and supplier together as a cohesive, high-performance team. The RFP also marks the culmination of the strategic planning process and represents the formal means for communicating the acquirer's requirements to the supplier. The RFP must contain clear and sufficient technical guidance so that the supplier has a definite picture of how the system is envisioned to perform when delivered. It is also important that a technical functional description of software requirements is included and clearly scoped. The success of an acquisition is directly linked to the quality of the RFP [Army 2007].

Establishing sound supplier contractual requirements is the foundation for successful software acquisition and development. During the RFP preparation, the acquisition team must have software expertise in the application domain, software acquisition management, software process, software project management, and software safety to ensure that essential technical data and data rights are acquired to meet the project needs. The RFP process should include a software acquisition team review of the acquisition package.

The RFP should address the following essential elements: 1) Statement of Work (SOW)/Statement of Objectives (SOO), 2) Contract Data Requirements List (CDRL) (DD 1423), 3) System Specification, and 4) Data Rights.

The RFP essential elements are discussed in the following paragraphs.

Statement of Work (SOW)/Statement of Objective (SOO)

The RFP Statement of Work (SOW) or Statement of Objective (SOO) is the primary document for translating management requirements into contractual tasks. It is the basis for communicating management requirements to the supplier. The SOW/SOO defines the tasks required to successfully supply the software that meets the specification requirements. The SOW/SOO must provide sufficient detail to allow the supplier to scope the effort, cost it, and provide a responsive technical solution to the requirements.

The SOW/SOO must also contain tasking information for the preparation of documentation per the *Contract Data Requirements Lists* (CDRL) items. Each tasking statement should reference any applicable CDRL item which will be delivered by that task. The CDRL will be discussed in detail later.

While the SOW/SOO states the specific tasks to be performed, it must not tell the supplier how to do the required work.

Though not specified in the SOW/SOO, the selection of major software components, *Computer Software Configuration Items* (CSCI), is a critical process in development. It provides the first step of system design and sets the system management framework.

The SOW/SOO should include but not be limited to the following key software tasking: 1) Software development process, 2) Software management, 3) Software engineering, 4) Tools and environment, 5) Risk management, 6) Technical reviews, and 7) Direct technical visibility

Contract Data Requirements List (CDRL)

Software products (data/artifact/documentation) are absolutely essential for managing the development process. Software products are a natural by-product of the development effort to capture results for each software development activity. The RFP's CDRL is the primary vehicle for acquiring software products from the supplier. The CDRL is a list of authorized data requirements for a specific procurement that forms a part of the contract. It is comprised of either a single DD Form 1423, or a series of DD Form 1423 (individual CDRL forms) containing data requirements and delivery information. The CDRL is the standard format for identifying potential data requirements in a solicitation and deliverable data requirements in a contract. **Subpart 215.470 Estimated Data Prices** of the Defense Federal Acquisition Regulation Supplement (DFARS) requires the use of a CDRL in solicitation when the contract will require delivery of data. The CDRL should be used only to acquire technical data and rights which are essential to meeting the needs of the requiring organization. All CDRL line items should be referenced in the SOW paragraphs describing the supplier software effort. The SOW tasks the preparation of data. The SOW takes precedence over the CDRL in a contract. Therefore, it is essential that the language in the SOW be consistent with and does not conflict with the CDRL in any way. The CDRL line items should be managed by the System Program Office data manager. Special data provisions (such as data rights, warranty, etc.) if required should be identified in the contract via special contract clauses (e.g., DFARS).

Each CDRL should identify the specific applicable Data Item Description (DID). This DID must have been accepted and approved by the acquirer. Assist-Quick Search should be used to access the current DIDs http://assist.daps.dla.mil/quicksearch . The DID selected should be used as is, or with non-applicable requirements tailored out (i.e., data requirements cannot be added to, only tailored out of a DID). Tailoring instruction (i.e., "BLK: Delete paragraphs…") are entered in the remarks section (Block 16). The DID should be referenced by the exact identifier and title with reference to any issue or revision identifier. The DID defines the data that the supplier is required to provide, along with delivery instruction.

CDRL submission should be associated with technical review milestones such as Software Specification Review (SSR), Preliminary Design Review (PDR), and Critical Design Review (CDR). This does not mean that other types of data such as software work products will not be required to be prepared. Non-deliverable data must be prepared, but will not require acquirer consent to change it. Data not identified as deliverable should be prepared and evaluated to the established software processes defined in the software plans (i.e., development, configuration management, and quality assurance).

CDRL items should be delivered to the acquirer to allow significant time for the acquirer to perform a detailed review and distribution of the review comments to the supplier prior to the technical design review. *Block 6, Requiring Office* should specify the organization having primary responsibility for reviewing the data product and recommending acceptance/rejection of the data. *Block 8, Approval Code* should specify approval of a draft before preparation of the final data item. With a SOO approach, the offerors propose a CDRL list that is tailored to their design. The proposed CDRL line items are then evaluated by the acquirer during proposal evaluation.

Typically software CDRL items include: 1) Software Development Plan (SDP), 2) Software Configuration Management Plan (SCMP), 3) Software Quality Assurance Plan (SQAP), 4) Software Requirements Specification (SRS), 5) Software Detailed Description (SDD), 6) Software Test Plan (STP), 7) Software Test Description (STD), 8) Software Test Results (STR), and 9) Software Version Description (SVD).

Lesson Learned: All software-related contract data requirements on DD Form 1423 contained in the acquisition packages should be prepared by the software acquisition team, reviewed by all applicable distribution addressee organizations, and approved by either the appropriate Project Manager, Program Director or Data Requirements Review Board Chairperson. This activity should be performed prior to action by the Contracting Officer.

System Specification

The System Specification is used to establish top-level technical performance, design, development, integration, and verification requirements for the software intensive system.

Data Rights

Computer software data rights are of great importance to both the acquirer and the supplier. The acquirer must have sufficient rights to enable the use, maintenance, and replication of the computer software data. The supplier wants to ensure that its proprietary rights for computer software developed at company expense are protected in order to maintain its competitive advantage. According to the Federal Acquisition Regulation (FAR), the term "*data*" simply means *recorded information*, including software. "*Computer software*" means computer programs, computer data bases, and the documentation thereof. Policies governing the rights to these data are found in FAR Subpart 27.4-Rights in Data and Copyrights, DFARS Subpart 227.72 – Rights in Computer Software and Computer Software Documentation, Revised June 21,2005, and DFARS 252.227-7014 –Rights in Noncommercial Computer Software and Noncommercial Computer Software Documentation [DPAP 01].

SOFTWARE ACQUISITION TEAM

Software acquisition management of software intensive systems involves a number of organizations, including the customer or user of the system, the contracting agency or the acquirer of the system, and the supplier (developer) or seller of development products or services. During the establishment of supplier agreements (contract) phase, the acquisition team must consist of software expertise in the application domain, software acquisition management, software process management, software project management, software engineering, and software safety, as needed. A software acquisition management manager should be designated to be responsible for establishing and managing the acquisition. The software acquisition management manager should be knowledgeable and experienced in software engineering including acquisition, development, and process improvement and should be responsible for coordinating the scope of technical software work and the terms and conditions of the contract with the affected parties. The appropriate business function groups, such as finance, contracts, and legal, should establish and monitor the terms and conditions of the contract.

Table 3 depicts examples of the author's software acquisition management roles for the C-130 AMP, FAA NAS Plan programs and LMAS C-130J.

Table 3 Examples of the Author's Software Acquisition Management Roles

Programs	***Software Acquisition Management Roles***
Air Force C-130 AMP	Systems software subject matter expertise for the following Integrated Product Team: 1) Avionics Operational Flight Program (OFP) Software, 2) Systems Integration Facility (SIF), and 3) Systems Requirements Design & Test
FAA NAS Plan (AAS)	System Development Manager responsible for software, hardware, and testing
FAA NAS Plan (TDWR)	System Program Office software lead and software formal qualification test director
FAA NAS Plan (MLS)	Software acquisition management subject matter expert
FAA NAS Plan (RCE)	Software acquisition management subject matter expert
LMAS C-130J	Software supplier management manager

The software acquisition team should have adequate resources and funding to perform the acquisition activities. The software acquisition manager and other individuals who are involved in the acquisition process should be trained to perform the acquisition activities. Examples of training should include:

- Basic Software Acquisition Management: a) Preparing and planning for software acquisition, b) Evaluating supplier's software process capability, c) Evaluating supplier's software estimates and plans, d) Selecting suppliers, e) Managing the acquisition
- Intermediate Software Acquisition Management
- Advanced Software Acquisition Management

The software acquisition team should receive orientation in the technical aspects of the project. Examples of orientation should include: 1) Application domain, 2) Software technologies being applied, 3) Software tools, 4) Methodology, and 5) Processes, Procedures, and Standards being used.

SOFTWARE DEVELOPMENT ENVIRONMENT

Software intensive systems software products result from a software engineering process that integrates all software engineering activities to produce correct, consistent software products effectively and efficiently. Software engineering has been defined as "the disciplined application of engineering, scientific, and mathematical principles, methods, and tools to the production of quality software" [Humphrey 1989]. Its domain includes activities such as planning, estimating, modeling, designing, implementing, testing, maintaining, and managing.

ACQUIRER/SUPPLIER RELATIONSHIP

The relationship of the acquirer's organization program management, software acquisition management, supplier's organization software development management, software engineering, software configuration management (CM), and software quality assurance (QA) for a software environment is shown in Figure 1.

The acquirer organization program manager (PM) is responsible for the life cycle management of the system or end-item. The PM has full authority, responsibility, and resources to execute the acquisition program. Software acquisition management is the process of assembling the software requirements for the system, planning the software activities, supporting acquiring the supplier, monitoring and controlling the software implementation.

The supplier's organization for software development management, sometimes called the Software Project Management-Software Integrated Product Team (IPT), is headed by a manager who is responsible for the software project planning, managing, tracking, and oversight. The software development manager is the single point of contact for the acquirer's software acquisition management. Software development management involves project planning which includes developing estimates for the work to be performed, establishing the necessary commitments, and defining the plans to perform the work.

The planning process includes steps to: 1) Estimate the size of the software work products and the resources needed, 2) Produce a schedule 3) Identify and assess software risks and 4) Negotiate commitments.

Software development management provides visibility into actual progress so that the supplier management and the acquirer software acquisition management can take effective actions when the software project's performance deviates significantly from the software plans. Software development management tracking and oversight tasks involve tracking and reviewing the software accomplishments and results against documented estimates, commitments, and plans; and adjusting these plans based on the actual results.

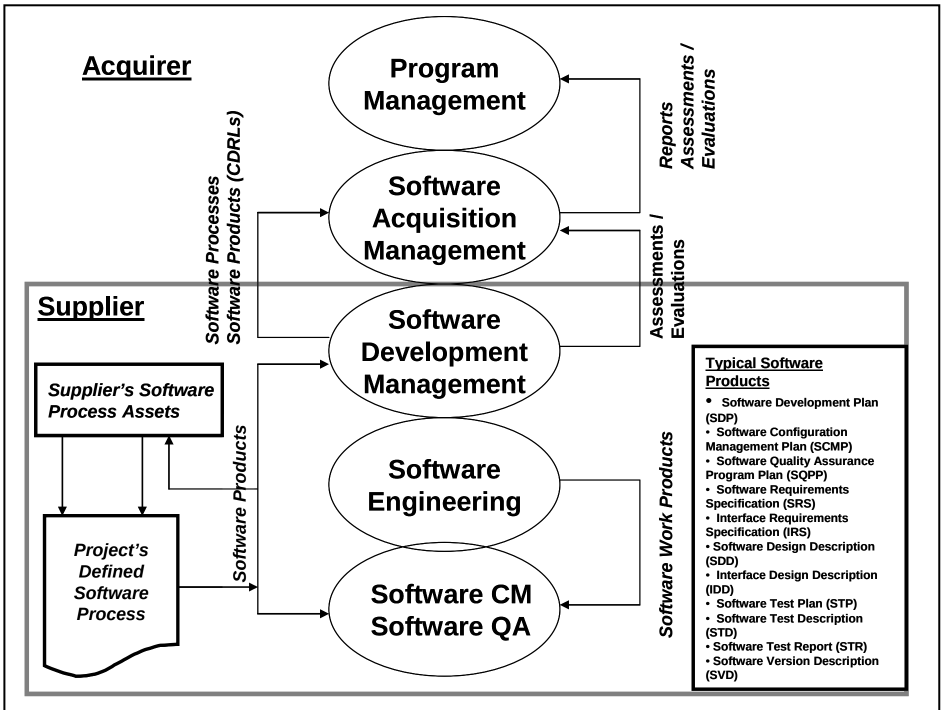


Figure 1 Software Environment

Software engineering involves building and maintaining the software product using the project's defined software process and appropriate methods/tools. The software engineering tasks include analyzing the system requirements allocated to software, developing the software requirements, developing the software architecture, designing the software, implementing the software in the code, integrating the software components, and testing the software to verify that it satisfies the specified requirements. During the software development life cycle, software products are developed. The supplier performs software product peer reviews, management review and approval, and places the software products under developmental configuration control prior to delivering the software products (CDRL Items) to the acquirer for assessment. Figure 1, depicts typical software products.

Software configuration management (CM) establishes and maintains the integrity of software work products throughout the project's software life cycle. The CM task involves software configuration identification, configuration change control and maintenance of the integrity and traceability of the configuration. The work products placed under software configuration management include the software products that are delivered to the acquirer and the items that are identified with or required to create these software products (e.g., complier, build procedures).

Software quality assurance (QA) provides staff and management with objective insight into the software processes and associated software work products. The QA task involves reviewing and auditing the software products and activities to verify compliance with the applicable procedures and standards to provide the software project and other appropriate managers with the results of these reviews and audits.

The intersection between software project manager and software acquisition is at the project management level because software acquisition includes this level as well as lateral and higher levels of management. Software acquisition management provides the software development visibility to program management. Typical work products include: supplier progress reports/performance measures and assessments/evaluation reports.

SUPPLIER SOFTWARE PROCESS DEFINITION

The supplier's organization should establish and maintain a set of software process assets. The supplier's software development project should develop a defined software process by tailoring the organization's standard software process per the supplier's documented procedure. The supplier's procedure typically should specify that a software life cycle model is selected from among those approved by the organization, to satisfy program contractual and operational constraints using the guidelines established by the organization. After the supplier's software development project has established a defined software process, the supplier should develop the project's software plans (i.e., software development plan, software configuration management plan, and software quality assurance plan), which describe the use of the project's defined software process.

The supplier's software development plan (SDP) should establish the plans for conducting a software development effort. The term "software development" is meant to include new development, modification, reuse, reengineering, and all other activities resulting in software products. The SDP should provide the acquirer with:

- Insight into the processes to be followed for software development
- A tool for monitoring the processes to be followed for software development
- Methods to be used
- Approaches to be followed for each activity
- Project schedules, organization, and resources
- Procedures for performing general software development activities

The SDP should provide general plans for software development and for performing detailed software development activities. The SDP should also include schedules, an activity network, project organization and resources.

The software development environment should also be augmented by management methods and practices such as measuring and monitoring progress, judging the quality of the product, validating the deliverable products against contractual requirements, and conducting technical reviews. Theses activities provide the information that managers need to control software acquisition. They provide a means of communication among all personnel involved in developing and managing the project. They also provide checkpoints, commonly called quality gates, by which interim deliveries can be checked and quality can be assessed. These practices ensure that the software product is properly built and satisfies the contractual requirements.

TECHNICAL PERFORMANCE ASSESSMENTS

Software acquisition is a collaborative process between the acquirer and the supplier. Gaining adequate visibility into the supplier's defined software process, plans and software products (artifacts) is key to technical performance assessments. The acquirer must have all the artifacts necessary to ensure that the program is proceeding as it should. Assessment techniques provide visibility into the quality and reliability of the software products. This section discusses the key technical performance assessment activities performed after the contract is awarded. It also discusses the essential contractual requirements that allow adequate visibility into the supplier's defined software process and products.

The key technical performance assessment activities are: Software Process Assessments, Progress Assessment, and Product Assessment. Each key technical performance assessment is discussed in the following paragraphs.

SOFTWARE PROCESS ASSESSMENTS

The acquirer should conduct software process assessment activities to verify that software management, software configuration management, and software quality assurance activities and products are in compliance with contractual requirements and in accordance with the supplier's documented defined software process and plans such as the Software Development Plan (SDP), Software Configuration Management Plan (SCMP), and Software Quality Assurance Program Plan (SQAPP). The results should be analyzed to detect issues and to identify risks to the program.

The contract should provide the mechanism to allow the acquirer to access the supplier's defined software process and artifacts to gain insight into the supplier's software management, software configuration management, and software quality assurance processes and products.

PROGRESS ASSESSMENTS

Reviews should be held to allow the acquirer to determine progress, status, surface issues, and to provide feedback to the supplier. The key focus should be "what is done and the product being built". There are normally two general types of reviews, formal and informal. Formal reviews, such as technical reviews, are those mandated by the selected development methodology or contractual requirements. Informal reviews are those conducted by the supplier such as peer reviews and walkthroughs.

Formal reviews should be structured around well defined procedures and objectives and coupled with realistic project milestones. Formal reviews should consist of: Program Management Reviews, Technical Interchange Reviews (TIM), and In-Process Reviews (IPR). TIMs should be conducted periodically by the supplier's IPTs to allow the acquirer to gain visibility into the development progress, product quality, and to discuss issues/candidate risks. Items to be considered should include, but not be limited to: 1) accomplishments, 2) issues, 3) risks, 4) upcoming events, and 5) schedule. IPRs should be conducted to review in-process work products in order to improve the process, product quality, and to provide feedback.

SOFTWARE PRODUCTS ASSESSMENT

As discussed previously, software products are essential for managing the development process and development of quality software. Software products should be prepared throughout the development lifecycle to capture the results of each software management and engineering activity. Prior to exiting each development phase, the supplier should perform software product evaluation and place the software product under configuration control prior to delivering the software product to the acquirer.

For software management and engineering tasks, the SOW should specify the preparation and delivery of software products in accordance with the CDRL item. CDRL items should be delivered to the acquirer to allow significant time for a detailed review and distribution of the review comments prior to the design review.

The acquirer should establish a process for reviewing the software products and the disposition of review comments. The review comments should identifies the discrepancy, provide a recommendation, and recommend acceptance/rejection of the data item.

SOFTWARE TEST EVALUATION

The development of software involves a series of production activities in which opportunities for human induced software errors are enormous. Errors may begin at the very inception of the process, where the software requirements may be erroneously or imperfectly specified, as well as later in the design and coding phases. Because of this likelihood of human error producing errors in software development, the development process is accompanied by a quality assurance activity – *Software Testing.* Software testing is a critical element of software quality assurance and represents the ultimate evaluation of the software requirements, design, and coding [Jones 1993-1].

As software is developed, errors are introduced due to many sources such as human mistakes, complexity, modularity, and ambiguous requirements. Studies have established conclusively that software testing can make the product more reliable and usable [Musa 1987] [Dunn 1984]. Studies have shown that between 46 percent [Endres 1975] and 60 percent [Voges 1979] of all software errors originate in the software requirements analysis phase. Software testing is the software quality assurance technique used to evaluate the "as-built" software product to ensure that the probability of failure due to latent errors is low enough for acceptance. Software testing cannot by itself provide an assurance of failure-free operations. Defects should be removed at the earliest opportunity. For example, a requirement defect (ambiguous or erroneous specification of the functions to be performed) propagated to the design phase results in designer labor expended on work that will have to be redone. Software testing is the last opportunity to remove latent defects before the product baseline is established.

There are typically three levels of software testing performed by the supplier – *Unit Testing, Integration Testing, and Formal Qualification Testing (FQT)*. Approximately 65 percent of all errors can be caught in Unit Testing, which is dominated by path testing. Software testing should be specified in the Contract Statement of Work (SOW) and in the supplier's defined software process and software development plan. Testing criteria, regression testing strategy, adequacy of testing (levels), strategy (functional, structural), and test coverage should be documented in the Software Test Plan (STP) in accordance with the Contract SOW CDRL peculiar Data Item Description (DID) and reviewed with the acquirer.

For each level of software testing, test readiness criteria should be established. Examples of criteria to determine test readiness include:

- Software "as-built" units have successfully completed a code peer review and unit testing before they enter integration testing.
- The software "as-built" has successfully completed integration testing before it enters FQT.
- A Test Readiness Review (TRR) is held.

The supplier should perform software testing with the intent of finding errors. Classes of tests such as timing tests, erroneous input tests, and maximum capacity tests should be performed. The acquirer software acquisition team should witness all formal testing

Problem Reporting/Tracking

The supplier should document problems identified during FQT and track the *problem report* (PR) to ensure closure in accordance with the supplier's software defined process. The supplier should apply a priority classification in accordance with the supplier's defined software process to all problems detected in the deliverable software and its documentation that has been placed under developmental configuration control. The supplier should collect and analyze data on problems identified during the FQT and in peer reviews in accordance with the supplier's defined software process.

The supplier's Configuration Control Board should analyze the PR to determine the impact to the work product, related work product, and schedule/cost. The acquirer and supplier should monitor the closure of PRs to determine the impact to the software release milestone. A PR will typically go through a number of states from the time it is reported until its closure such as analysis required, in-worked, and verified.

When the PR is generated, the supplier should record the PR in the *Change Control System* in accordance with the supplier's defined software process. The Change Control System should include the storage media, the procedures, and tools for recording and accessing PRs. During the PR closing process, each state should correspond to a milestone which provides the acquirer and the supplier visibility of each PR's progress, i.e., how many PRs have been reported, how many PRs are pending, and how many PRs are closed.

The supplier should report the progress of each PR and discuss the PR analysis results at the Technical Interchange Meetings (TIMs). The supplier should provide the acquirer access to the Change Control System and PR analysis. The Change Control System information should be used to determine the aspects of software engineering needing improvement and how effective previous analyses and testing have been.

How Much Testing is Enough?

Considering that complete test coverage is generally not possible [Jones 1993-1], the acquirer and supplier face a difficult question in deciding when to release the software. The acquirer and supplier should mutually agree on completion criteria such as completion of an arbitrary number of test runs with no open priority 1 (HIGH) and 2 (MEDIUM) severity problem reports. During the test planning activity, the acquirer and supplier should establish a failure intensity objective (FIO) using a software reliability growth model such as Time-Between-Failure Models or an Error-Count Model. The failures should be used with the software failure model to determine that the FIO has been met-the software is acceptable.

REQUIREMENTS MANAGEMENT

During the development life cycle, requirements change for a variety of reasons – additional requirements are derived or changes are made to the existing requirements. The supplier should manage changes to the requirements as they evolve and identify any inconsistencies that occur among the plans, work products, and requirements. It is essential to manage these additions and changes efficiently and effectively. To effectively analyze the impact of the changes, it is necessary that the source of each requirement be known and the rationale for any change be documented. The supplier should track measures of requirements volatility to determine whether new or revised controls are necessary.

Traceability is one of the essential activities of requirements management. Traceability ensures that the right products are being built at each phase of the software development life cycle to trace the progress of that development and to reduce the effort required to determine the impacts of requested changes. The supplier should establish and maintain *bidirectional traceability* between source requirements and all products in accordance with the CDRL in a requirements database using a requirements tool such as Telelogic *Dynamic Object-Oriented Requirements Systems (DOORS®)* and Serena Software, Inc. *Requirements Traceability Management (RTM).*

Forward traceability ensures proper direction of the evolving product (the right product is being built) and indicates the completeness of the subsequent implementation. For example, during system architectural design, the supplier should conduct analysis to determine the allocation of the requirements in the system specification to system components [i.e., Hardware Configuration Items (HWCIs), Computer Software Configuration Items (CSCIs), and manual operation]. Each component should be assigned a project-unique identifier. The acquirer should ensure that the requirement traceability shows forward traceability from each system requirement to the system component (i.e., CSCI) that implements that requirement. Forward traceability should be performed during the product development life cycle. The acquirer should ensure that the requirements traceability shows that all system and software requirements allocation to design, code, and test.

Backward traceability helps ensure that the evolving product is not expanding the scope of the project by adding design elements, code, test or other work products that are not specified in the requirements. For example, during software requirements analysis, the acquirer should ensure that backward requirements traceability is shown from each CSCI requirement to the system requirement that it addresses. During software design, the acquirer should ensure that the backward requirements traceability is shown:

- From each software component identified to the CSCI requirements allocated
- From each test identified to the CSCI requirements and, if applicable, the system requirements that it addresses
- From each test case to the CSCI requirements or system requirements it addresses

Benefits of bidirectional requirements traceability include the ability to:

- Analyze the impact of a change to all work products affected by a changed requirement and to all requirements affected by a change or defect in a work product
- Assess current status of the requirements and the project to identify missing requirements.

RISK MANAGEMENT

Studies have shown that technical performance, cost, and schedule risks are inherent in delivering high-quality, highly-reliable software intensive systems within cost and schedule constraints [GAO 1999]. Some projects are even canceled before any products are delivered. Programs are planned to succeed. They are planned to produce the product in accordance with the contract and within cost and schedule constraints. However, there are many obstacles to their success. One key obstacle is the inability to see cost and schedule issues as symptoms of a more fundamental problem such as unforeseen software size growth, requirements growth, the ability to determine the complexity of the product, and the ability to perform.

This underlying problem is often an unresolved technical risk. It occurs because programs are unable to cope with technical risk in the development process. In the 1986 General Accounting Office (GAO) report entitled *Technical Risk Assessment: The Current Status of DOD Efforts* [GAO1986], the GAO reported that:

® DOORS is a trademark of Telelogic AB.

> "*Technical risks are inherent in the development of new weapon systems, whose advanced performance requirements may exceed the capabilities of current technology. Not to anticipate technical risk before and during the development process creates the potential for schedule and cost problems and, more, the possibility that a system will fail to meet its design specifications and will not function as intended*."

There are two factors that comprise a risk: Probability or likelihood that it will occur and loss resulting from its occurrence. Therefore, risk is a part of any activity and can never be eliminated, nor can all risks ever be known. Risk in itself is not bad; risk is essential to progress, and failure is often a key part of learning. However, we must learn to balance the possible negative consequences of risk against the potential benefits of its associated opportunity.

Technical risk is the possibility that the application of software engineering theory, principles, and techniques will fail to yield the desired software product. Technical risk is comprised of the underlying technological factors that may cause the final product to be: 1) Overly expensive, 2) Delivered late, and 3) Unacceptable to the acquirer.

Risk management is becoming recognized as a best practice for reducing the surprise factor. There are many models for managing risk. A systematic risk management process must have a set of practices, which must be performed to manage project risks. To improve the efficiency and effectiveness of the Air Force acquisition processes and software management, the Air Force expects the acquisition communities to address Risk Management throughout the life cycle of the acquisition program [DoD 2004].

The acquiring organization should establish a risk management model to define a systematic process for managing a project's risks. The model should consist of a number of functions that are performed as continuous activities throughout a project life cycle. The risk management model practices should include: 1) Identify, 2) Analyze, 3) Plan, 4) Track, 5) Control, and 6) Communicate and Document. [Van Scoy 1992] A consistent format for risk statement should be established to allow rapid recognition of the impact or consequence to be avoided and to show causes or conditions that need to be eliminated or reduced to avoid the consequence.

PERFORMANCE MEASUREMENTS

Performance measurement is a key to managing and producing quality software and is an essential element of software development process improvement [Humphrey 1989]. Software development is often out-of-control. Mr. Thomas DeMarco (the author of *Controlling Software Projects*) asserts that "*You cannot control what you cannot measure*" [DeMarco 1982]. The acquirer and supplier should use performance measurements as software management and quality indicators (metrics) to augment conventional acquisition and development reports. As mandated by Section 804 of the National Defense Acquisition Act, "*metrics for performance measurement and continual process improvement*" is a requirement [Section 804-2003].

Performance measurements should be captured to document actual-versus-planned activities and to identify problems in development. For tracking key criteria, metrics should be selected that are directly measurable during development to evaluate progress and identify significant predictors of the final project success or failure [Jones 2004]. The acquirer and supplier should mutually agree on and implement selected performance measurements to provide management visibility into the software development and acquisition process. For example, Tom DeMarco, in his book *Controlling Software Projects*, states that metrics should be measurable, or quantifiable; independent from influence by project personnel; accountable, in that the data can be collected; and precise, in that the degree of exactness can be specified.

Watts S. Humphrey, in *Managing the Software Process*, states that metrics should be objective (versus subjective), explicit (versus derived), absolute (versus relative), and dynamic (versus static).

Performance measurements should be selected to provide insight into four key acquisition areas:
- *Process.* Provides insight into the software development processes and how it is working.
- *Product.* Measures the quality of the product (e.g., frequency of requirement changes, number of problems, number of review comment discrepancies, etc.).
- *Project.* Progress-oriented measures (e.g., schedule attainment, CDRL delivery, etc.).
- *Productivity.* The rate at which the work is progressing.

Performance measurements selected should provide a top-level overview of the software development progress and an early-warning mechanism for detecting software quality problems. These performance measurements should provided feedback to the project to refine the process and contribute to positive control. The acquirer should use performance measurements for escalating the discussion of progress and status to the supplier and the acquirer's System Program Office (SPO).

Typical acquirer and supplier performance measurements should include:
- Software Size – estimated and tracked at the CSCI level (Source-Line-Of-Code)
- Cost/Schedule Deviation –tracking and assessment using cost/schedule control system criteria
- Schedule Progress – estimate related to the size estimates of work products and major milestones
- Software Development Progress –tracks software activities (e.g., software requirements analysis, design, implementation, etc.)
- Software Formal Qualification Testing (FQT) Progress – determines the supplier's ability to maintain the software FQT progress and the degree to which the "as-built" software satisfies the requirements
- Software Requirements Stability – degree to which changes in the requirements affect the implementation effort
- Computer Resource Utilization – tracks changes in the estimated/actual use of execution time and memory utilization in a worst case processing load
- Software Product Review Item Discrepancies – number of discrepancies generated during the product evaluation

SUMMARY

Successful development and acquisition of software is paramount for acquiring software intensive system programs. The quality of any software product is the direct result of acquisition and development management techniques, controls, processes, and tools. This paper has discussed the key success elements in software acquisition and development: 1) the contract, 2) software acquisition team, 3) software development environment, 4) technical performance assessments, 5) software test evaluation, 6) requirements management, 7) risk management, and 8) performance measurements.

This paper has shown that software acquisition management techniques such as technical performance assessments can be used to ensure compliance with techniques, control, processes and to verify software product quality. Performance measurements have been shown as effective tools for monitoring cost, schedule, technical performance, and quality. As previously discussed, performance measurements are useful in identifying deficiencies in the software development processes and products, in providing a vehicle for process improvement, and as pivotal predicators of final project success or failure.

To ensure the highest probability of success, the acquirer and supplier must be comparable in software management and engineering experience, and process capability/maturity. Both must have a team risk and metric approach, and possess the ability to execute the plan. The acquirer's management processes, practices, and resultant decisions can negatively impact the supplier's processes and product quality.

This paper has shown that by proficiently detailing the contractual requirements, applying highly skilled qualified acquirer personnel, effectively assessing the supplier's technical performance through processes and products, participating in management and technical design reviews, participating in software testing, measuring performance and managing risk, the acquirer can make the supplier's software development process more efficient and effective.

There are many parallel and related efforts underway that address or mandate improvement in the acquisition of software products:

- Public Law 107-314 Section 804 of the National Defense Authorization Act, released in December 2002 [Section 804-2003]
- Clinger-Cohen Act: initiatives such as Software Assurance and Open Architecture
- The best practice model Capability Maturity Model Integration (CMMI) for Acquisition

REFERENCES

URLs are valid as of the publication date of this document.

Army 2007	*Army Source Selection Manual*, Office of the Assistant Secretary of the Army, Feb 07 http://www.acanrhq.army.mil/library/Army%20Source%20Selection%20Manual.pdf
Boehm 1987	Boehm, B., "*Industrial Software Metrics Top Ten List*", IEEE Software, September 1987.
CMMI 2006	CMMI Product Team. *CMMI for Development, Version 1.2,* CMMI-DEV, V1.2, CMU/SEI-2006-TR-008, ESC-TR-2006-008, August 2006, Carnegie Mellon Software Engineering Institute, Pittsburgh, PA http://www.sei.cmu.edu/pub/documents/06.reports/pdf/06tr008.pdf
CMMI-AM 2005	Thomas Bernard, Brian Gallagher, Roger Bate, Hal Wilson. CMMI Acquisition Module (CMMI-AM), Version 1.1, CMM/SEI-2005-TR-011, ESC-TR-2005-011, May 2005. http://www.sei.cmu.edu/pub/documents/05.reports/pdf/05tr011.pdf
DAPA 01	Defense Acquisition Performance Assessment Report, January 2006, A Report by the Assessment Panel of the Defense Acquisition Performance Assessment Project for the Deputy Secretary of Defense.
DeMarco 1982	DeMarco T., *Controlling Software Projects*, Prentice Hall, New York, 1982.
DOD-STD-2167A	*MILITARY STANDARD Defense System Software Development*, 29 February 1988 (Canceled)
DPAP 01	Office of the Under Secretary of Defense for Acquisition Technology and Logistics, Defense Procurement and Acquisition Policy, http://www.acq.osd.mil/dpap/dars/dfars/
DUNN 1984	Dunn, R. H., *Software Defect Removal*, McGraw-Hill, New York, 1984.
ENDRES 1975	Endres, A., *An Analysis of Errors and Their Causes in System Programs*, IEEE Transactions in Software Engineering SE-1, 2, June 1975.
EVMS 2002	American National Standards Institute / Electrical Industries Association, Earned Value Management System (EIA-748-A), Washington, D.C., 2002
FAA 1995	FAA Type Certification Process, March 2, 1995 http://www.airweb.faa.gov/Regulatory_and_Guidance_Library/rgOrders.nsf/0/832744411e40d71486256f8e0076a146/$FILE/Order8110-4.pdf
G490405	Software Development Plan for the Terminal Doppler Weather Radar (TDWR), G490405, Rev A, 5 March 1991.
GAO 1986	Government Accounting Office. *Technical Risk Assessment: The Current Status of DOD Efforts*, GAO/PEMD-86-5, Government Accounting Office, Washington, D.C. , 1986

GAO 1999	(U.S. General Accounting Office, "High Risk Series: An Update. "GAO/HR-99-1. Washington, D.C.:GAO, Jan. 1999. http://www.gao.gov/pas/hr99001.pdf).
Haley 1995	Haley, T., et al. "*Raytheon Electronic Systems Experience in Software Improvement.*" CMU/SEI-95-TR-017. Pittsburg, PA: Software Engineering Institute, Nov 1995.
Humphrey 1989	Humphrey, Watts S., *"Managing the Software Process"*, Addison-Wesley Publishing Company, 1989.
Jones 1990	*Software Acquisition Management: Managing the Acquisition of Terminal Doppler Weather Radar (TDWR) System Software Design*, Department of Commerce, 1990**.**
Jones 1992	*Software Acquisition Management: Managing The Acquisition of Computer Software Using DOD-STD-2167A*, Proceeding of the 37th Air Traffic Control Association Fall Conference, 1992.
Jones 1993	*Software Metrics Effectiveness in Software Acquisition Management*, Proceeding of the 38th Air Traffic Control Association Fall Conference, October, 1993.
Jones 1993-1	*Software Testing: Methods and Techniques*, Proceeding of the 38th Air Traffic Control Association Fall Conference, October, 1993.
Jones 1999	*Estimating Software Size, Cost, and Schedule: Mission Success Through Life Cycle Processes*, 1999 Joint ISPA/SCEA Conference, 1999.
Jones 2004	*Mission Success – Estimating Software Development*, 26th Annual International Society of Parametric Analysts International Conference in Frascati, Italy (Models Track, 11 May 2004) http://www.ispa-cost.org/2004_conf_toc.htm
Jones 2004-1	*Successful Acquisition of FAA Terminal Doppler Weather Radar*, Third Annual Conference on the Acquisition of Software-Intensive Systems (Experience Track, 26 January 2004) http://www.sei.cmu.edu/programs/acquisition-support/conf/2004-presentations/jones.pdf
MIL-STD-1521B	*MILITARY STANDARD Technical Reviews and Audits For Systems, Equipments, and Computer Software*, 4 June 1985 (Canceled)
MUSA 1987	Musa, J., Iamnino, A., and Okumoto, K., *Software Reliability: Measurement, Prediction, Application*, McGraw-Hill, 1987.
NDIA 2005	National Defense Industrial Association (NDIA) Program Management Systems Committee (PMSC) ANSI/EIA-748-A Standard for Earned Value Management Systems Intent Guide
OIG 1998	Office of Inspection General, *Audit Report, Advance Automation System, Federal Aviation Administration*, Report Number: AV-1998-113, Date Issued: April 15, 1998, http://www.oig.dot.gov/StreamFile?file=/data/pdfdocs/av1998113.pdf
RTCA 1992	RTCA/DO-178B, "*Software Considerations in Airborne Systems and Equipment Certification*", December 1, 1992.
Sea 1995	*IEEE Names Raytheon Leader In Software Industry*, Sea Technology Ocean Engineering, 1995
Section 804-2003	U.S. Congress. "National Defense Authorization Act for Fiscal Year 2002", Calendar No. 163, 107th Congress, 1st Session, S. 1438. Washington, C.C., 2001

	http://www.theorator.com/bills107/s1438.html
STSC 01	*Guidelines for Successful Acquisition and Management of Software-Intensive System: Weapon Systems Command and Control Systems Management Information Systems*, DEPARTMENT OF THE AIR FORCE Software Technology Support Center, Version 3.0, May 2000.
Van Scoy 1992	Van Scoy, Roger L., *Software Development Risk: Opportunity, Not Problem* (CMMU/SEI-92-TR-30, ADA258743), Software Engineering Institute, Pittsburgh, 1992.
VOGES 1979	Voges, U. and Taylor, J. R., *A Survey of Methods for the Validation of Safety-Related Software* (presented at the IFAC Workshop on SAFECOMP-1979, Stattgart: May 16, 1979)

ABOUT THE AUTHOR

Mr. James E. Jones has 40 years military and commercial software-intensive systems experience in software acquisition management, process improvement, project management, systems integration, independent validation and verification, software development, software sustainment, and formal qualification testing. Mr. Jones was a key participant in the development of MIL-STD-2167A *Defense System Software Development*, *Guide to the Software Engineering Body of Knowledge*®, and *Adapting CMMI*® *for Acquisition Organizations: A Preliminary Report*.

Currently, Mr. Jones is providing software systems engineering support to the United States Air Force for the C-130 Avionics Modernization Program (AMP) modernization of the C-130 fleet. He served as Lockheed Martin Aeronautical Systems (LMAS) Supplier Software Manager for the C-130J Hercules Air Vehicle Supplier Operational Flight Program (OFP) Software. Mr. Jones provided software acquisition management support to the FAA's multi-billion-dollar NAS Plan Modernization Program (now the *Capital Investment Plan*) under the Systems Engineering and Integration Contract for multiple programs. He served as a software subject matter expert for several FAA NAS Plan Modernization Programs and provided a deposition for contract termination.

Mr. Jones served as the Software Engineering Process Group (SEPG) lead, developed and trained organization-wide software processes and helped organizations achieve Software Engineering Institute (SEI[6]) Capability Maturity Model (CMM®) Level 3.

Mr. Jones earned a BS in Mathematics from Tuskegee Institute and pursued an MBA at Florida Institute of Technology. He is co-inventor of US patent 4451702. May 29, 1984., US patent 4479034. October 23, 1984.

SELECTED PUBLICATIONS AND PRESENTATIONS

- *Process Improvement in a Small Company*, Proceedings of the First International Research Workshop for Process Improvement in Small Settings, October 2005. http://www.sei.cmu.edu/pub/documents/06.reports/pdf/06sr001.pdf
- *Successful Acquisition of FAA Terminal Doppler Weather Radar*, Third Annual Conference on the Acquisition of Software-Intensive Systems (Experience Track, 26 January 2004). http://www.sei.cmu.edu/programs/acquisition-support/conf/2004-presentations/jones.pdf
- *Mission Success: Estimating Software Projects*, The International Society of Parametric Analysts, 26^{th} Annual Conference, May 10-12, 2004.
- *Estimating Software Size, Cost, and Schedule: Mission Success Through Life Cycle Processes*, 1999 Joint ISPA/SCEA Conference, 1999.
- *Conforming to ISO 9001: A Mission Success Solution to Product Development*, Lockheed Martin Management and Data Systems, 1997.
- *Estimating Software Development Size, Cost, and Schedule*, Lockheed Martin Engineering Process Improvement, 1994.

® Registered in U. S. Patent office
® CMMI is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.
[6] The Software Engineering Institute (SEI) is a federally funded research and development center sponsored by the U. S. Department of Defense and operated by Carnegie Mellon University.
® CMM and CMMI are registered in the U. S. Patent and Trademark Office by Carnegie Mellon University.

- *The Road to Mission Success Delivering Quality Software within Cost and Schedule*, Martin Marietta Communications Systems, Camden Connection, 1994
- *Software Metrics Effectiveness in Software Acquisition Management*, 38th Air Traffic Control Association Fall Conference, 1993.
- *Software Testing: Methods and Techniques*, 38th Air Traffic Control Association Fall Conference, 1993.
- *Software Acquisition Management: Managing The Acquisition of Computer Software Using DoD-STD-2167A*, 37th Annual Air Traffic Control Association Conference Proceedings, November, 1992.