Contents

(click on the GREEN box to view)

MIL-STD-498 Points of Contact

MIL-STD-498 and Harmonization Working Group

Dr Raghu Singh, SPAWAR, 703-602-4491
singhr@smtp-gw.spawar.navy.mil

MIL-STD-498 Guidebooks

Ms. Norma A. Stopyra, SPAWAR, 703-602-3178
stopyran@smtp-gw.spawar.navy.mil

OFFICE OF THE ASSISTANT SECRETARY OF DEFENSE
3300 DEFENSE PENTAGON
WASHINGTON, DC 20301-3300

08 NOV 1994

MEMORANDUM FOR ASSISTANT SECRETARY OF THE NAVY (RESEARCH, DEVELOPMENT AND ACQUISITION)

SUBJECT: MIL-STD-498 “Software Development and Documentation”

The Defense Standards Improvement Council has approved the adoption of MIL-STD-498 for an interim period of two years. This approval is based on acceptance of the rationale for adoption, including a clear plan for transitioning to a nongovernment standard, contained in your October 17 memo. MIL-STD-498 must be replaced by a suitable nongovernment standard within two years, or be submitted to the Council for reapproval.

/Brad Bergman/

Walter B. Bergman II
Chairman, Defense
Standards Improvement Council

cc: DUSD (Acquisition Reform)
DASD (Industrial Affairs)
Co-Chairs, Software Management Review Board

DEPARTMENT OF THE NAVY

OFFICE OF THE ASSISTANT SECRETARY
(Research, Development and Acquisition)
WASHINGTON, D.C. 20360-1000

FEB 14 1995

MEMORANDUM FOR DISTRIBUTION

Subj: APPROVAL OF DEPARTMENT WIDE WAIVERS

Ref: (a) DOD Standards Improvement Program Plan

In accordance with the provisions of reference (a), Department-wide waivers have been approved on the following military standards without any restrictions for a maximum period of two years. The Department-wide waivers serve as the only justification needed when citing these military standards as requirements in solicitations.

MIL-STD-461, Electromagnetic Emissions and Susceptibility Requirements

MIL-STD-462, Measurement of Electromagnetic Interference Characteristics

MIL-STD-498, Software Development and Documentation

/D. E. Porter/

Navy Standards Improvement Executive
Acquisition Reform Office

MIL-STD-498

Jane Radatz, Myrna Olson, and Stuart Campbell, Logicon

(based on February 1995 CrossTalk article published by STSC at Hill AFB, UT)

MIL-STD-498, Software Development and Documentation, was approved on 8 November 1994. What is this new standard? How was it developed? What is its relationship to commercial standards? What are its key features? How can it be used in today's new environment of acquisition reform? This article attempts to answer these questions.

What is MIL-STD-498?

MIL-STD-498 is the DoD's new software development standard. It was developed with four primary objectives:

- Merge DOD-STD-2167A, used for weapon systems, with DOD-STD-7935A, used for automated information systems, creating a single software development standard for DoD
- Resolve issues raised in the use of these standards
- Ensure compatibility with current DoD directives, instructions, and other standards
- Provide a basis for U.S. implementation of ISO/IEC 12207, Software Life Cycle Processes

How was MIL-STD-498 Developed?

To meet these objectives, OUSD(A)DR&E and the Joint Logistics Commanders established a Harmonization Working Group (HWG) in October 1991. The HWG is made up of representatives from the Army, Navy, Air Force, and Marine Corps; DoD Agencies; other Federal Agencies; the Ministries of Defense of Germany, Canada, and the United Kingdom; and U.S. Industry. The group contains representatives from both the mission critical and automated information system communities. Figure 1 identifies the members of the HWG.

The HWG met bimonthly during the 2½ years the standard was in development. Logicon, the group's contractor, had responsibility for identifying

issues for discussion, researching those issues, presenting them to the HWG, developing drafts of the standard and DIDs, analyzing comments on the draft, presenting proposed resolutions to the HWG, and acting on the final decisions. HWG members had responsibility for reviewing drafts, coordinating the drafts within their organizations, and representing their organization's views at HWG meetings. This approach resulted in early visibility and broad consensus for the standard.

Relationship to Commercial Standards

A key reason for MIL-STD-498's approval in today's environment discouraging the use of military standards was that no commercial alternative existed or was expected for several years. MIL-STD-498 was issued for an interim period of two years, at which time an alternative, commercial standard is expected.

The Electronic Industries Association (EIA) and Institute of Electrical and Electronics Engineers (IEEE) formed a joint working group in October 1994 to develop a commercial standard based on MIL-STD-498. When the resulting commercial standard is issued, the status of MIL-STD-498 will be revisited.

Key Features of MIL-STD-498

The MIL-STD-498 package consists of the standard and 22 Data Item Descriptions (DIDs). The paragraphs the follow highlight key features.

Accommodating Incremental and Evolutionary Development

Both DODI 5000.2 and DODI 8120.2 define multiple program strategies for acquiring systems and software. Included are “Grand Design,” “Incremental” (also called “Preplanned Product Improvement”), and “Evolutionary” strategies. A key goal of MIL-STD-498 is accommodating each of these strategies.

To meet this goal, MIL-STD-498 is written in terms of developing software in multiple “builds.” Each build incorporates a specified subset of the planned capabilities of the software. The builds might be prototypes, versions offering partial functionality, or other partial or complete versions of the software. Figures and notes in the standard tell how to interpret the standard on projects involving multiple builds. Figure 2 identifies the key activities in MIL-STD-498 and shows how each activity might be applied in one or more builds.

Alternatives to Formal Reviews and Audits

One of the biggest distractions from "real work" on a software development project is preparing for formal reviews and audits. A common horror story is that everyone stops "real work" six weeks before a formal review and starts generating viewgraphs for the review. The result can be tremendous expenditure of time and energy on a review that is an overly detailed snapshot of where the project was six weeks before. The developer spends hundreds of staff-hours preparing, and the acquirer is swamped by information overload.

Instead of formal reviews and audits, MIL-STD-498 calls for more frequent, low-key joint (acquirer/developer) technical and management reviews, focusing on natural work products rather than viewgraphs or other specially prepared materials. These reviews include informal discussions of status, ideas, approaches, risks, etc. The idea of these reviews is ongoing communication between the acquirer and developer with minimum time wasted. An appendix in MIL-STD-498 suggests candidate reviews to hold.

Decreased Emphasis on Documentation and Greater Compatibility with CASE Tools

A frequent complaint about software development under military standards is that it is document driven. Rather than focusing on the real work to be done, the developer must generate an endless series of documents, some of which have little to do with getting the job done. The developer may have to translate real work products, such as data in computer-aided software engineering (CASE) tools, into paper documents or translate work products from the actual format in which work is being done to an artificial format imposed by DIDs.

MIL-STD-498 addresses this problem in several ways. First, the MIL-STD-498 activities that generate information require the developer to "define and record" information, not to "prepare a given document." Notes in the standard emphasize that this wording is designed to accommodate project information in its natural form, for example, in CASE tools, rather than requiring the developer to prepare traditional documents.

Second, MIL-STD-498 distinguishes between the planning and engineering activities that are the "real work" of a project and the activity of preparing a deliverable. This point may seem the same as the one just above, but is not. The paragraph above is about allowing the developer to record project information in the most natural form, for example, in a CASE tool vs a paper document. This

paragraph is about not making work products, whatever their form, deliverable without good reason. In the past, acquirers often thought that the way to make the developer do the work was to require deliverables containing evidence of the work (the "homework" approach to acquisition management). But preparing information for delivery takes extra time and should not be required without good reason. MIL-STD-498 requires the work to take place, regardless of whether the results are made deliverable, and gives the acquirer access to the results in the developer's facility. When deliverables are needed, they are called for in the CDRL.

Third, many of the activities in MIL-STD-498 rely on DIDs to fully define them. For example, the activity of preparing a Software Development Plan (SDP) consists of defining and recording all information called for by the Software Development Plan DID. This requirement applies regardless of whether the SDP is in the form of document and regardless of whether the resulting plan is deliverable. The DID is used as a checklist of information to be defined in carrying out the required activity. This is a new role for DIDs. Figure 3 illustrates the role of DIDs and software products in MIL-STD-498, emphasizing the distinction made between natural work products, the form in which they reside, and their status as deliverables or non-deliverables. Figure 4 identifies the MIL-STD-498 DIDs.

Improved Links to Systems Engineering

A key aspect of the merger of DOD-STD-2167A and DOD-STD-7935A was that the resulting standard had to cover two types of systems: (1) hardware-software systems (such as radar systems) for which the standard covers only the software portion, and (2) software systems (such as payroll systems) for which the standard governs overall system development.

MIL-STD-498 covers this duality by including system-level activities, requiring the developer to "participate" in them, and stating that: 1) If the software to be developed is part of a hardware-software system for which the standard covers only the software portion, the term "participate" is to be interpreted as "take part in, as described in the Software Development Plan," and 2) If the software (possibly with its computers) is considered to constitute a system, the term "participate" is to be interpreted as "be responsible for." This use of "participate" covers both cases.

Use of Software Management Indicators

There is increasing emphasis in DoD on quantitative measurement of software progress, size, quality, and other attributes. The idea behind these initiatives is that only by obtaining quantitative measures will it be possible to make real progress in improving the software development process.

MIL-STD-498 takes a cautious step into this area by requiring the developer to define and apply software management indicators and providing a set of candidate management indicators to serve as a starting point. It is left to the developer to propose in the Software Development Plan the indicators to be used, the data to be collected, the approach to interpreting the data, and the reporting approach. The acquirer can provide feedback on the proposed approach in reviewing the Software Development Plan.

Improved Coverage of Modification, Reuse, and Reengineering

There is increasing emphasis in DoD on meeting user needs by modifying, reusing, and reengineering existing software products rather than developing everything “from scratch.” These initiatives apply not just to the software itself, but to associated software products such as architectures and design.

MIL-STD-498 supports these initiatives. Each activity in the standard may be performed by modifying, reusing, or reengineering existing items, as well as by developing something new. The standard requires the developer to identify and evaluate reusable software products for use in fulfilling contract requirements, and to incorporate those that meet the criteria established for the project. It provides criteria for evaluating software products for reuse and tells how to interpret the standard when applied to reused items. It requires the developer to identify and analyze opportunities for developing software products for reuse and to notify the acquirer of those that provide cost benefits and are compatible with program objectives. Finally, it provides a definition of reengineering and its constituent activities, and provides a diagram showing how MIL-STD-498 can be applied to a reengineering project.

Compatibility with Object-Oriented and Other Methods

Although not intended to do so, DOD-STD-2167A has been perceived to favor top-down functional decomposition over other development methods. Most frequently cited are DOD-STD-2167A's requirements regarding software architecture, in which Computer Software Configuration Items (CSCIs) are

decomposed into computer software components (CSCs), which are decomposed into other CSCs, which are finally decomposed into computer software units (CSUs), which are associated with physical code entities. While some users of DOD-STD-2167A have no problem with this framework, others find it inhibiting to their preferred methods.

MIL-STD-498 redefines software architecture by requiring simply that CSCIs be decomposed into software units, which may or may not be related to each other in a hierarchical manner. A software unit is any element in the design of a CSCI, for example, a major subdivision of a CSCI, a component of that subdivision, a class, object, module, function, routine, or database. Software units may occur at different levels of a hierarchy and may consist of other software units. MIL-STD-498 further acknowledges that software units in the design may or may not have a one-to-one relationship with the code and data entities that implement them or with the computer files containing those entities. This generalization of software units and separation of logical from physical entities provides more flexibility to employ a variety of software development methods.

Applying MIL-STD-498 in Today's Environment

Under current policy, MIL-STD-498 can be used in four basic ways, illustrated in Figure 5 and described in the following paragraphs.

The first way is to cite MIL-STD-498 as a required standard in Requests for Proposal (RFPs) and in contracts, that is, the traditional way of applying military standards. This approach requires a waiver to the Perry memo of 29 June 1994. Individual Services and Agencies are considering blanket waivers for MIL-STD-498. The status of these waivers is not known at this time.

A second way to apply MIL-STD-498 is to make its provisions mandatory in the RFP (but not the contract), to request a Software Development Plan (SDP) in proposals, and to put the SDP of the winning bidder on contract. This approach also requires a waiver.

A third way is to cite MIL-STD-498 as guidance in the RFP, to require an SDP based on the guidance, and to put the SDP of the winning bidder on contract. This approach does not require a waiver.

A fourth way is based on the Perry memo's policy that bidders are to be encouraged to propose alternative standards to those cited in the RFP. If a bidder

proposes to follow MIL-STD-498 and that bidder is selected, no waiver is needed to put the standard on contract.

Conclusion

Development of a commercial standard based on MIL-STD-498 has begun. This standard will be the U.S. implementation of ISO/IEC 12207, Software Life Cycle Processes. In the meantime, MIL-STD-498 is in place, providing a single, harmonized approach to software development in DoD, a solid improvement over its predecessor standards, and a basis for the commercial standard. A guidebook for MIL-STD-498 will be released in January 1995. We will keep you informed of other developments.

Jane Radatz, Myrna Olson, Stuart Campbell
Logicon, MS 600
8910 University Center Lane
San Diego, CA 92122
Voice: 619-455-7663 Fax: 619-587-4781
Internet: mil-std-498@logicon.com

About the Authors

Jane Radatz is Logicon's technical and lead author for MIL-STD-498. She also led Logicon's work in the development of DOD-STD-2167A and DOD-STD-2168, and designed the associated TAILOR/2167A tools. She has 20 years experience in software engineering and technical management.

Myrna Olson is Logicon's project manager and associate author for MIL-STD-498. She has 10 years experience in software engineering and technical management and holds master's degrees in both business administration and software engineering.

Stuart Campbell is Logicon's associate author for MIL-STD-498. He has 15 years experience in software engineering and technical management of real-time simulation systems.

Dr. Raghu Singh, Chairman, USN

Mr. Paul Anderson, USN (MCCR)

Mr. Robert Beermann, USA (MCCR)

Ms. Jonilynn Benarick, DoD IG

Mr. Ron Berlack, IEEE (Industry)

Ms. Linda Brown, OASD (C3I)

Mr. Perry DeWeese, CODSIA (Industry)

Dr. Ken Eyre, Canada (Industry)

Mr. Robert Gagnon, OASD (ES)

Mr. Dung Ha, USMC (MCCR)

Mr. Robert Hegland, USA (IPSC)

Mr. Alan Huguley, DMA

Dr. Helmut Hummel, Fed Rep of Germany

Mr. Tim Janes, UK MoD

Mr. Don Kosco, USAF (MCCR)

Mr. Robert Mac Kissock, USMC (IPSC)

Ms. Donna McCloud, DLA

LtC. Dixie McNeme, HQDA

Mr. Fred Moxley, DISA (JIEO/CIM)

Maj. George Newberry, USAF (MCCR)

Mr. Burt Newlin, OASD (C3I)

Mr. Randy Paul, CIA

Mr. Mark Paulk, SEI (Process)

Mr. Glenn Plonk, NSA

Ms. Francine Portenier, Canada DND

Mr. Bill Price, USCG

Mr. Robert Schwenk, USA (IPSC)

Ms. Deborah Sery, FAA

Mr. Paul Shebalin, DSMC

Ms. Linda Sheets, USAF (IPSC)

Mr. Don Sova, NASA

Ms. Norma Stopyra, USN (MCCR)

Ms. Rebecca Wade, USN (IPSC)

Ms. Dolores Wallace, NIST

Mr. Dick Wesley, DNA

Contractor:

Ms. Myrna Olson, Logicon

Ms. Jane Radatz, Logicon

Mr. Stuart Campbell, Logicon

MCCR =Mission Critical Systems

IPSC = Information Systems

Figure 1. Members of the Harmonization Working Group.

Activity	Builds			
	Build 1	Build 2	Build 3	Build 4
5.1 Project planning and oversight	x	x	x	x
5.2 Establishing a software development environment	x	x	x	x
5.3 System requirements analysis	x	x		
5.4 System design	x	x	x	
5.5 Software requirements analysis	x	x	x	x
5.6 Software design	x	x	x	x
5.7 Software implementation and unit testing	x	x	x	x
5.8 Unit integration and testing	x	x	x	x
5.9 CSCI qualification testing		x	x	x
5.10 CSCI/HWCI integration and testing		x	x	x
5.11 System qualification testing			x	x
5.12 Preparing for software use	x	x	x	x
5.13 Preparing for software transition				x
Integral processes:				
5.14 Software configuration management	x	x	x	x
5.15 Software product evaluation	x	x	x	x
5.16 Software quality assurance	x	x	x	x
5.17 Corrective action	x	x	x	x
5.18 Joint technical and management reviews	x	x	x	x
5.19 Other activities	x	x	x	x

Figure 2. One Possible Mapping of MIL-STD-498 Activities to Multiple Builds.

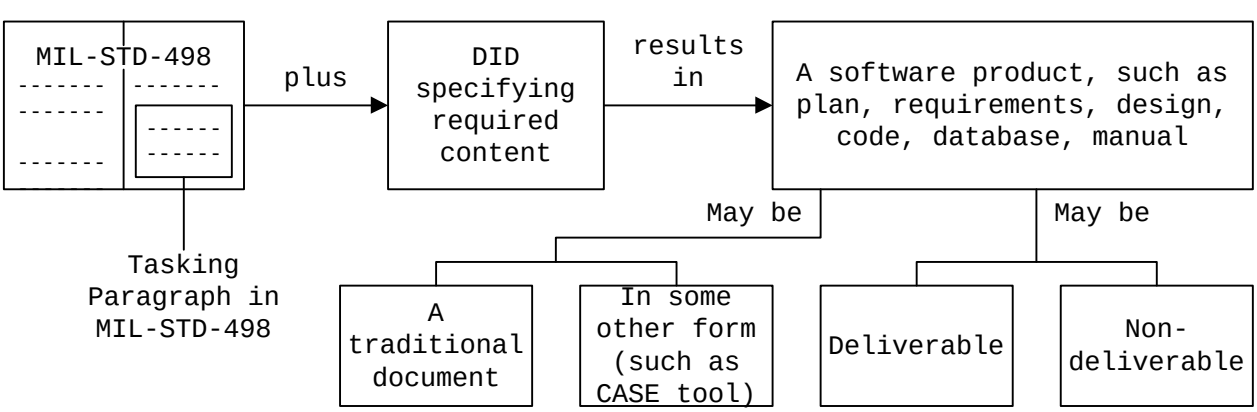

Figure 3. The Role of DIDs and Software Products in MIL-STD-498.

MIL-STD-498 DID	Resulting Software Product
Software Development Plan (SDP)	A plan for performing the software development
Software Test Plan (STP)	A plan for conducting qualification testing
Software Installation Plan (SIP)	A plan for installing the software at user sites
Software Transition Plan (STrP)	A plan for transitioning to the support agency
Operational Concept Description (OCD)	The operational concept for the system
System/Subsystem Specification (SSS)	The requirements to be met by the system
Software Requirements Specification (SRS)	The requirements to be met by a CSCI
Interface Requirements Specification (IRS)	The requirements for one more interfaces
System/Subsystem Design Description (SSDD)	The design of the system
Software Design Description (SDD)	The design of a CSCI
Interface Design Description (IDD)	The design of one or more interfaces
Database Design Description (DBDD)	The design of a database
Software Test Description (STD)	Test cases/procedures for qualification testing
Software Test Report (STR)	Test results of qualification testing
Software Product Specification (SPS)	The executable software, the source files, and information to be used for support
Software Version Description (SVD)	A list of delivered files and related information
Software User Manual (SUM)	Instructions for hands-on users of software
Software Input/Output Manual (SIOM)	Instructions for users depending on SW operators
Software Center Operator Manual (SCOM)	Instructions for SW operators supporting users
Computer Operation Manual (COM)	Instructions for operating a computer
Computer Programming Manual (CPM)	Instructions for programming a computer
Firmware Support Manual (FSM)	Instructions for programming firmware devices

Figure 4. The MIL-STD-498 DIDs.

Four basic ways of applying MIL-STD-498			
Cite as a requirement in the RFP and contract in the traditional manner (waiver required)	Cite as a requirement in the RFP; rely on resulting SDP to award and manage the contract (waiver required)	Cite as guidance in the RFP; rely on resulting SDP to award and manage the contract (no waiver required)	Cite as a requirement in the contract as a result of the winning bidder having proposed it (no waiver required)

Figure 5. Ways of Applying MIL-STD-498.

Changes from DOD-STD-2167A to MIL-STD-498

Maj. George A. Newberry, USAF

(based on April 1995 CrossTalk article published by STSC at Hill AFB, UT)

Introduction

As DOD-STD-2167A (Defense Systems Software Development) is the *de facto* software development standard in the United States and throughout the world, the software community has begun to ask how this standard compares with its replacement, MIL-STD-498 (Software Development and Documentation). This article focuses on the differences you will see between the old and the new standards.

By the last 1980s, DOD-STD-2167A had become well accepted for use in software development. However, a few concerns began to surface in regard to its perceived imposition of the waterfall development model, inflexibility related to object-oriented design, excessive documentation, no guidance on management indicators, and the need to incorporate new development techniques such as reuse and reengineering. In addition, DoD began to eliminate unnecessary standards and consolidate others where appropriate.

In 1991, the Joint Logistics Commanders (JLC) established the Harmonization Working Group (HWG), a group of more than 50 individuals from the software communities within the services, defense agencies, federal agencies, the U.S. commercial industry, as well as the defense ministries of Canada, Germany, and the United Kingdom. This group was tasked to resolve the concerns raised about DOD-STD-2167A and combine it with the other DoD software standards: DOD-STD-7935 (Automated Information Systems (AIS) Documentation Standards), and DOD-STD-1703 (NS) (Product Standards). The resultant document, MIL-STD-498, was completed August 15, 1994 and approved by the Office of the Secretary of Defense on November 8, 1994 for an interim period of two years. This interim period was established to allow a commercial equivalent to be developed.

Removing the Waterfall Bias

Although DOD-STD-2167A never dictated the use of a specific software development technique, it was perceived by many to impose the waterfall model

(known in the AIS community as "Grand Design"). This model has been used in a restrictive manner. Developers felt they could only perform each step of the software development process one time, must perform the steps in sequence, and finish each step before beginning the next. To add more flexibility to the development process, iterative techniques began to be used that involved prototyping and incremental development. These new techniques are often termed the *evolutionary* or *spiral* model.

MIL-STD-498 describes software development in one or more incremental *builds*. Each build implements a specified subset of the planned capabilities. The process steps are repeated for each build, and within each build, steps may be overlapping and iterative. Figure 1 in Section 5 of MIL-STD-498 shows an example of one possible mapping of MIL-STD-498 activities to multiple builds. A word of caution needs to be given here. Since MIL-STD-498 provides no *default* process to follow, the skill level required to use it is higher than the old standard. For this reason, a *quick* guidebook for MIL-STD-498 has been developed. The guidebook is in final draft status and should be published shortly. The guidebook is intended to help DoD and other acquisition program managers implement the new standard. For more information regarding the guidebook, please contact Norma Stopyra at SPAWAR, at 703-602-3178 or over the Internet at `stopyran@smtp-gw.spawar.navy.mil`.

Alternatives to Formal Reviews and Audits

Another noteworthy difference between the standards is in reviews and audits. DOD-STD-2167A imposes formal reviews and audits that emphasize the waterfall model and are often nonproductive "dog and pony shows." The developer spends thousands of staff-hours preparing special materials for these meetings, and the acquirer is then swamped by information overload.

MIL-STD-498 relaxes this formality and simply requires joint technical and management reviews. The standard states the reviews should be frequent and informal and focus on the project's status, approach, and risks. The biggest change in this area is probably an emphasis on using existing work products rather than special materials generated simply for the review. The objective is to move the communication between acquirer and developer from formal presentations to a continuous exchange of information.

Compatibility with Nonhierarchical Methods

DOD-STD-2167A shows software development as a top-down functional decomposition. Computer software configuration items (CSCIs), which are decomposed into other CSCs, etc., which are finally decomposed into computer software units (CSUs). Design, testing, configuration management (CM), and other activities are based on this decomposition. This has caused problems for some developers, specifically when performing object-oriented analysis and design.

MIL-STD-498 removes this limitation. CSCIs are decomposed into software units, which may or may not be related to each other in a hierarchical manner. Design, testing, CM, etc., are based on the developer-designated software units. This change provides more flexibility to use methods best suited to the specific project, such as object-oriented analysis and design.

Less Emphasis on Documentation

DOD-STD-2167A was written in terms of producing documents, e.g., "These plans shall be documented in ... ." The implication was the developer needed to prepare and deliver a series of documents related to every piece of the project. In addition, the standard has been interpreted to discourage use of computer aided software engineering (CASE) tools by referencing only traditional documents. The data item descriptions (DIDs) also were seen to enforce this interpretation.

MIL-STD-498 was written in terms of defining and recording information, e.g., "Develop and record plans for ... ." This information may or may not be in the form of a traditional document and may or may not be deliverable. The wording in both the standard and DIDs encourages the use of CASE tools by simply specifying the required information, regardless of the form it takes. The objective is to reduce cost by simplifying how information is recorded and to eliminate unnecessary documentation.

Improved Links to Systems Engineering

DOD-STD-2167A assumes software is embedded in a hardware-software system. It also assumes someone else performs system-level activities. The document does not acknowledge software engineering's participation in upfront systems engineering.

MIL-STD-498 acknowledges both software-only systems and systems that contain software as one element, i.e., *embedded* systems. It also contains system-level requirements for software-only systems and requires participation of software engineering in system-level activities for embedded systems.

Use of Software Management Indicators

DOD-STD-2167A does not require use of software management indicators and offers no guidance on the subject.

MIL-STD-498 requires the developer to define and apply software management indicators and provides a set of candidate indicators to serve as a starting point.

Improved Coverage of Databases

DOD-STD-2167A was developed for use on weapons systems. It was difficult to use on AIS systems because it largely ignores databases—key elements of AIS.

MIL-STD-498 was written for use on development of all systems. It defines software as computer programs and computer databases (consistent with the FAR), adds a database design description DID, and uses the term *implementation* vice *coding* to include data. Thus, the new standard covers databases in all stages—requirements, design, and implementation.

Better Coverage of Modifications, Reuse, and Reengineering

DOD-STD-2167A was written in terms of new development. This means it took interpretation and extensive tailoring to apply the standard to modification, reuse, and reengineering. Although it requires the developer to consider incorporating non-developmental software, it leaves unclear what criteria to use in the consideration or even how the standard is to be applied when incorporating reusable software.

MIL-STD-498 explicitly acknowledges that each step may involve modifying, reusing, or reengineering existing items vice new development. It expands the reuse requirement to cover all software products, e.g., reusable architectures, not just the software, and Section 4.2.3 tells how to apply the standard to reused items. Appendix B provides mandatory and nonmandatory criteria to be used in evaluating items for reuse and explains how to interpret each

requirement when applied to reusable software. In addition, Figure 12 in Appendix G provides a model that shows how to apply the standard to a reengineering project.

Increased Emphasis on Supportability

Although DOD-STD-2167A is strong on supportability it does leave some loopholes. It is written as though testing is the final activity. The standard also fails to make clear the considerable tasks of preparing the completed software for delivery to users and the support agency. In addition, it doesn't clearly distinguish between preparing for software use and preparing for software transition.

MIL-STD-498 requires identification of all resources used or generated during development that will be needed by the support agency. Sections 5.12 and 5.13 cover the hardware, software, data, and documentation necessary for support and also require a demonstration be conducted showing that the delivered software can be supported given the identified resources. It also requires the rationale for key decisions, which may be useful to the support agency, be recorded. The standard also includes separate activities to prepare for software use and transition and distinguishes the tasks that constitute each of these activities.

Improved Evaluation and Review Criteria

DOD-STD-2167A defines criteria for software product evaluations but applies these criteria only to deliverables. It also relies on MIL-STD-1521B for formal review criteria.

MIL-STD-498 strengthens the criteria for software product evaluations by making them applicable to in-process work products, not just to draft and final deliverables (see Section 5.15). It uses the same criteria for evaluations and joint technical reviews, thus integrating both activities.

Clearer Distinction Between Requirements and Design

DOD-STD-2167A defined requirements as *what* the system or software must do and design as *how* it is implemented. This traditional distinction has led to argument, confusion, and numerous program delays.

MIL-STD-498 seeks to eliminate this confusion by modifying the definitions. Requirements are defined as what the acquirer cares enough about to make conditions for acceptance (may be *what* or *how*). Design is defined as the set

of decisions made by the developer in response to the stated requirements (may be *what* or *how*).

Inclusion of Software Quality Assurance

DOD-STD-2167A requires the developer to perform software product evaluations. However, it relies on DOD-STD-2168 for software quality assurance (SQA). The information contained in the latter document had originally been part of DOD-STD-2167 but was separated during development of DOD-STD-2167A. This was done so the SQA information could be incorporated in an update to MIL-Q-9858A (System Quality), but the update never occurred. This separate quality standard was interpreted by developers as a requirement for a separate quality organization. This was never the government's intent. The intent was to have a group of individuals who were not part of the development team (e.g., engineers working on a different project, evaluate the development effort. The two documents only succeeded in causing confusion over the difference between product evaluations and SQA and which organization had responsibility for what functions.

MIL-STD-498 basically returns us, from a quality perspective, to the self-contained document we had with DOD-STD-2167. It still requires the developer to perform software product evaluations and incorporates key points from DOD-STD-2168 regarding SQA. In addition, Section 5.16 eliminates the inference regarding a separate SQA organization and clarifies the scope of SQA in such a way that the overlap with software product evaluations is removed.

Clarification of CM Requirements

DOD-STD-2167A uses the concept of "developmental configuration," which has caused confusion. It does not acknowledge that computer files are often the entities placed under CM, rather than CSUs, which may be conceptual vice physical. It also limits configuration control to deliverables and to just before delivery.

MIL-STD-498 eliminates the concept of "developmental configuration," and requires identification of entities at the level at which they will actually be controlled, e.g., computer files. In addition, the standard does not dictate how the developer organizes his CM function. Section 5.14 simply states developers shall participate in selecting CSCIs, establish and implement configuration control procedures, prepare and maintain configuration status accounting, and support acquirer-conducted audits as specified in the contract. The object is to use the

developers' in-house CM organization and not require them to provide a separate one solely for DoD development.

Applicability to More Types of Projects

DOD-STD-2167A is written in terms of *government* vs. *contractor*. This has led to confusion regarding how to apply the standard to government *in-house* development projects and its use in prime contractor to subcontractor relationships.

MIL-STD-498 clears up this ambiguity by using the terms *acquirer* and *developer*. In-house development efforts should find this standard easier to use as it defines contractual terms in ways usable in the absence of a contract and clarifies its applicability in prime contractor to subcontractor relationships by generalizing usability of the standard.

Improved Treatment of the Software

DOD-STD-2167A assumes all software is to be ordered using a contract line item number (CLIN), which is not the case. It offers no means to order the executable software, source code, or data files via a contract data requirements list (CDRL). Its biggest weakness is it incorrectly mimics hardware development by treating the final design as the end product of software development.

MIL-STD-498 established a software product specification DID as a means to order the executable software and the source code and data via a CDRL. It treats the software, not the final design, as the final product of software development.

Amended Set of DIDs

DOD-STD-2167A, DOD-STD-7935, and DOD-STD-1703 contained 52 DIDs. Furthermore, many of the DIDs are inconsistent in their treatment of interfaces, data descriptions, system vice software requirements, and traceability. In addition, many individuals have inferred a single DID meant a separate document.

MIL-STD-498 reduces the number of DIDs to 22 (17 from DOD-STD-2167A, two restored [in response to user requests] from DOD-STD-2167 [the operational concept description and the database design description], three incorporated from DOD-STD-7935A [the software installation plan—for cases where the developer installs software, the software center operator manual—for

computer center staff, and the software input/output manual—for users of the computer center]).

Very little change was made to the content of those DIDs incorporated from DOD-STD-2167A. However, title modifications were made to a few of the DIDs to clarify ambiguities in the old titles and make them more consistent with other titles. Table 1 shows the affected DIDs. Two clarifications are noteworthy: "subsystem" was substituted for "segment" as it more clearly defines the scope of the document, and "description" replaced "document" since it decreases the implication of traditional documentation. Finally, this standard provides consistent treatment of interfaces, regardless of level or type of data, i.e., inputs, output, stored data, interface data, and messages; makes system and software specifications parallel and consistent; and provides consistent treatment of traceability. You have always been able to use multiple DIDs to request a single document, e.g., if you wanted to combine the software test plan with the software test description; however, experience at doing so has always been lacking. Although MIL-STD-498 does not address this problem, the guidebook details how to use multiple CDRLs to order a single document.

Title in DOD-STD-2167A	Title in MIL-STD-498
System/**Segment** Specification	System/**Subsystem** Specification
System/**Segment** Design **Document**	System/**Subsystem** Design **Description**
Software Design **Document**	Software Design **Description**
Interface Design **Document**	Interface Design **Description**
Computer **System Operator's** Manual	Computer **Operator** Manual
Software Programmer's Manual	**Computer Programming** Manual
Software **User's** Manual	Software **User** Manual
Computer Resources Integrated Support Document	**Software Transition Plan**
Version Description **Document**	**Software** Version Description

* **Bold** type denotes changed wording.

Table 1: Name Changes to Selected DIDs.

Conclusion

MIL-STD-498 corrects the problems reported in the use of DOD-STD-2167A and reflects current advances in the state-of-the art in software development. It is also applicable to more types of systems than the old standard and discusses current DoD initiatives such as reuse and reengineering. This makes MIL-STD-498 a clear improvement over DOD-STD-2167A.

I hope this article has provided some insight regarding the contents of MIL-STD-498 as well as the rationale for why certain changes were made. Since I will be directly involved in the commercial version of this standard, I welcome your comments and criticisms as you begin to use MIL-STD-498.

Acknowledgments

I wish to thank Logicon, specifically Myrna Olson and Jane Radatz, for providing the compilation of changes that are cited in the article. I would also like to thank Dr. Raghu Singh (SPAWAR) for his insightful comments concerning this article.

Maj. George A. Newberry
SAF/AQKS
1060 Air Force Pentagon
Washington, DC 20330-1060
Voice: 703-697-3108 DSN 227-3108
Fax: 703-697-8313 DSN 227-8313
Internet: newberrg@aqpo.hq.af.mil

About the Author

Maj. Newberry was the Air Force representative for embedded issues to the Harmonization Working Group and was responsible for consolidation and resolution of all Air Force comments. He is currently assigned as special assistant for software quality to the Air Force deputy assistant secretary for communications, computers, and support systems.