Changes from

DOD-STD-2167A

to

MIL-STD-498

01/05/95

Background

- MIL-STD-498 was developed by:
    - Merging DOD-STD-2167A with DOD-STD-7935A
    - Resolving issues identified in applying DOD-STD-2167A
    - Responding to changes in DoD directives, instructions, and standards
- This briefing focuses on changes from DOD-STD-2167A

Removing the Waterfall Bias

- 2167A is perceived to impose the waterfall model:
  - Perform each step of the software development process one time
  - Perform the steps in sequence
  - Completely finish each step before beginning the next

- 498 describes SW development in 1 or more incremental "builds"
  - Each build implements a specified subset of the planned capabilities
  - The process steps are repeated for each build
  - Within each build, steps may be overlapping and iterative

Alternatives to Formal Reviews and Audits

- 2167A imposes formal reviews and audits
  - The reviews and audits emphasize the waterfall model
  - They are often non-productive "dog and pony shows"
    - Developer spends thousands of staff-hours preparing
    - Acquirer is swamped by information overload
- 498 requires joint technical and management reviews instead
  - Frequent and informal
  - Preference for using natural work products, not special materials
  - Informal discussions of status, approaches, risks, etc.
  - Objective: ongoing communication between acquirer and developer

Compatibility with Non-Hierarchical Methods

- 2167A is perceived to favor top-down functional decomposition
    - CSCIs are decomposed into computer software components (CSCs), which are decomposed into other CSCs, ... which are finally decomposed into computer software units (CSUs)
    - Design, testing, CM, and other activities are based on this decomposition
- 498 removes this bias
    - 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
    - Result: More flexibility to use methods best suited to the project, such as object-oriented analysis and design

Less Emphasis on Documentation

- 2167A is written in terms of producing documents
  - "These plans shall be documented in a Software Development Plan"
  - "Document these requirements in the SW Requirements Specification"
  - Implication: Prepare and deliver a series of documents
- 498 is written in terms of defining and recording information
  - "Develop and record plans for conducting the [SW devel] activities"
  - "Define and record the software requirements to be met by each CSCI"
  - This information:
    - May or may not be in the form of a traditional document
    - May or may not be deliverable

Greater Compatibility with CASE tools

- 2167A may discourage use of CASE tools:
  - Wording acknowledges only traditional documents
  - The DIDs seem to enforce this interpretation

- 498 uses wording designed to accommodate CASE tools:
  - Requirements to "define and record" information
  - Words in both standard and DIDS suggest CASE tool contents as appropriate work products and deliverables
  - DIDs specify required information, regardless of the form it takes

Improved Links to Systems Engineering

- 2167A:
  - Assumes software is embedded in a hardware-software system
  - Assumes someone else performs system-level activities
  - Does not acknowledge software engineering’s participation in up-front systems engineering

- 498:
  - Acknowledges both software-only systems and systems that contain software as one element ("embedded" systems)
  - Contains system-level requirements for software-only systems
  - Requires participation of software engineering in system level activities for embedded systems

Use of Software Management Indicators

- 2167A:
  - Does not require use of software management indicators
  - Offers no guidance on this subject
- 498:
  - Requires the developer to define and apply software management indicators
  - Provides a set of candidate indicators to serve as a starting point

Improved Coverage of Databases

- 2167A:
  - Focuses on weapons systems (vs automated information systems (AIS))
  - Largely ignores databases -- key elements of AIS
- 498:
  - Covers both weapons systems and automated information systems
  - Defines software as computer programs and computer databases (consistent with the FAR)
  - Adds a Database Design Description DID
  - Uses the term "implementation" vs "coding" to include data
  - Covers databases in all stages -- requirements, design, implementation

Better Coverage of Modification, Reuse, Reengineering

- 2167A:
  - Is written in terms of new development
  - Takes interpretation/tailoring to apply to modification, reuse, reengineering
- 498:
  - Explicitly acknowledges that each step may involve modifying, reusing, or reengineering existing items vs new development
  - Provides an appendix telling how to interpret each requirement when applied to reused software
  - Provides a model showing application to a reengineering project

Improved Requirements for Reuse

- 2167A:
  - Requires the developer to consider incorporating non-developmental software
  - Leaves unclear what criteria to use in the consideration
  - Leaves unclear how the standard applies when software is reused
- 498:
  - Expands the reuse requirement to cover all software products, not just the software itself (such as reusable architectures)
  - Provides mandatory and non-mandatory criteria to be used in evaluating items for reuse
  - Tells how to apply the standard to reused items

Increased Emphasis on Supportability

- 2167A:
  - Is strong on supportability, but leaves some loopholes
- 498:
  - Requires identification of all resources used or generated during development that will be needed by the support agency
  - Covers hardware, software, data, documentation that may be needed
  - Requires a demonstration that the delivered software can be supported given those resources
  - Requires the recording of rationale for key decisions that may be useful to the support agency

Improved Evaluation/Review Criteria

- 2167A:
  - Defines criteria for software product evaluations
  - Applies the evaluations and criteria only to deliverables
  - Relies on MIL-STD-1521B for criteria for formal reviews

- 498:
  - Strengthens the criteria for software product evaluations
  - Makes the evaluations applicable to in-process work products, not just to draft and final deliverables
  - Uses the same criteria as the basis for joint technical reviews, thus integrating these activities

Clearer Distinction Between Requirements and Design

- 2167A uses the rule:
  - Requirements are "what" the system or software must do
  - Design is "how" it does it
    - This traditional distinction causes argument and confusion

- 498 uses the rule:
  - Requirements are what the acquirer cares enough about to make conditions for acceptance (may be "what" or "how")
  - Design is the set of decisions made by the developer in response to requirements (may be "what" or "how")
    - Requirement, design, and testing requirements reflect these meanings

Inclusion of Software Quality Assurance

- 2167A:
  - Requires the developer to perform software product evaluations
  - Relies on DOD-STD-2168 for software quality assurance
  - Many questions are raised about the difference between the two
- 498:
  - Requires the developer to perform software product evaluations
  - Incorporates software quality assurance, using key points from 2168
  - Clarifies the scope of SQA in such a way that the overlap with software product evaluations is removed

Clarification of CM Requirements

- 2167A:
  - Uses the concept of "developmental configuration," which causes confusion
  - Does not acknowledge that computer files are often the entities placed under CM, rather than CSUs, which may be conceptual vs physical
  - Limits configuration control to deliverables, and to just before delivery

- 498:
  - Eliminates the concept of "developmental configuration"
  - Requires identification of entities at the level at which they will actually be controlled (such as computer files)
  - Requires control of in-process and final work products, acknowledging a range of levels -- author control, project control, acquirer control, etc.

Applicability to More Types of Projects

- 2167A:
  - Is written in terms of "Government" vs "contractor"
  - Can be confusing for Government in-house development projects
  - Can be confusing for prime contractor - subcontractor relationships

- 498:
  - Is written in terms of "acquirer" and "developer"
  - Defines contractual terms in ways usable in the absence of a contract
  - Generalizes usability of the standard

Clearer Requirements on Preparing for Use/Support

- 2167A:
  - Is written as though testing is the final activity
  - Does not make clear the considerable tasks of preparing the completed software for delivery to users and to the support agency
  - Does not clearly distinguish between preparing for software use and preparing for software transition

- 498:
  - Includes separate activities for preparing for software use and preparing for software transition
  - Distinguishes the tasks that constitute each of these activities

Improved Treatment of the Software Itself

- 2167A:
  - Offers no means of ordering the executable software via CDRL
  - Offers no means of ordering the source code and data files via CDRL
  - Incorrectly mimics hardware development by treating the final design as the end product of software development
- 498:
  - Offers the SW Product Specification (SPS) DID as a means for ordering the executable software and the source code and data files via CDRL
  - Treats the software itself as the final product of software development

Amended Set of DIDs

- 2167A:
  - Has 17 associated DIDs
- 498 has 5 additional DIDs:
  - Two were in 2167, deleted in 2167A, restored based on user request
    - Operational Concept Description (OCD)
    - Database Design Description (DBDD)
  - Three were added as part of the merger with DOD-STD-7935A
    - Software Installation Plan (SIP) -- for cases where developer installs SW
    - Software Center Operator Manual (SCOM) -- for computer center staff
    - Software Input/Output Manual (SIOM) -- for users of computer center

Name Changes to Selected DIDs

Title in 2167A	Title in 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 **Operation** Manual
Software Programmer's Manual	**Computer Programming** Manual
Software **User's** Manual	Software **User** Manual
Computer Resources Integrated Support Doc	**Software Transition Plan**
Version Description **Document**	**Software** Version Description

- Rationale:
  - "Subsystem" is clearer than "segment"
  - "Description" decreases implication of traditional documentation
  - Others: clarification and consistency with other titles

Improved Consistency Among DIDs

- The 2167A DIDs are inconsistent in their treatment of:
  - Interfaces
  - Data descriptions
  - System vs software requirements
  - Traceability
- The 498 DIDs:
  - Provide consistent treatment of interfaces, regardless of level or type
  - Provide consistent treatment of data -- inputs, outputs, stored data, interface data, messages, etc.
  - Makes system and software specifications parallel and consistent
  - Provide consistent treatment of traceability

Conclusions

- MIL-STD-498:
  - Corrects problems reported in the use of DOD-STD-2167A
  - Reflects advances in the state-of-the-art in software development
  - Is applicable to more types of systems than DOD-STD-2167A
  - Reflects current DoD initiatives such as reuse and reengineering
- Based on these advances:
  - MIL-STD-498 is recognized as a clear improvement over 2167A
  - Most 2167A users are anxious to switch to 498