Briefing Report: Software Acquisition,
A Series of Lectures Prepared for
Acquisition/Logistics Excellence Week 2001

31 July 2002

Prepared by
L. A. ABEelson, R. J. ADAMS, S. ESLINGER,
J. S. KERNER, K. L. OWENS, B. E. SHAW
Software Engineering Subdivision
Computer Systems Division

B. R. TROUP
Software Engineering Department
Reconnaissance Systems Division

Prepared for
SPACE AND MISSILE SYSTEMS CENTER
AIR FORCE SPACE COMMAND
2430 E. El Segundo Boulevard
Los Angeles Air Force Base, CA 90245

Engineering and Technology Group

APPROVED FOR PUBLIC RELEASE;
DISTRIBUTION UNLIMITED
This report was submitted by The Aerospace Corporation, El Segundo, CA 90245-4691, under Contract No. F04701-00-C-0009 with the Space and Missile Systems Center, 2430 E. El Segundo Blvd., Los Angeles Air Force Base, CA 90245. It was reviewed and approved for The Aerospace Corporation by M. A. Rich, Principal Director, Software Engineering Subdivision. Michael Zambrana was the project officer for the Mission-Oriented Investigation and Experimentation (MOIE) program.

This report has been reviewed by the Public Affairs Office (PAS) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nationals.

This technical report has been reviewed and is approved for publication. Publication of this report does not constitute Air Force approval of the report's findings or conclusions. It is published only for the exchange and stimulation of ideas.

Michael Zambrana
SMC/AXE
**REPORT DOCUMENTATION PAGE**

<table>
<thead>
<tr>
<th>1. REPORT DATE (DD-MM-YYYY)</th>
<th>2. REPORT TYPE</th>
<th>3. DATES COVERED (From - To)</th>
</tr>
</thead>
<tbody>
<tr>
<td>31/07/2002</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4. TITLE AND SUBTITLE

Briefing Report: Software Acquisition, A Series of Lectures Prepared for Acquisition/Logistics Excellence Week 2001

5a. CONTRACT NUMBER

F04701-00-C-0009

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

6. AUTHOR(S)

L. A. Abelson, R. J. Adams, S. Eslinger,

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

The Aerospace Corporation
Laboratory Operations
El Segundo, CA 90245-4691

8. PERFORMING ORGANIZATION REPORT NUMBER

TR-2002(8550)-4

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

Space and Missile Systems Center
Air Force Space Command
2450 E. El Segundo Blvd.
Los Angeles Air Force Base, CA 90245

10. SPONSOR/MONITOR’S ACRONYM(S)

SMC

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

SMC-TR-02-33

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This report contains five lectures on topics in software acquisition that are critically important to SMC/NRO programs in today’s acquisition environment. Software acquisition is always a high risk endeavor for SMC’s/NRO’s software-intensive systems. This series of lectures provides specific, practical suggestions that can be applied immediately to SMC/NRO programs to mitigate software risks. Topics covered include COTS software lessons learned, object-oriented metrics, the Joint Technical Architecture (JTA) and Common Operating Environment (COE), human systems integration, and selecting a capable software contractor team.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:

<table>
<thead>
<tr>
<th>17. LIMITATION OF ABSTRACT</th>
<th>18. NUMBER OF PAGES</th>
<th>19a. NAME OF RESPONSIBLE PERSON</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. REPORT</td>
<td>b. ABSTRACT</td>
<td>c. THIS PAGE</td>
</tr>
<tr>
<td>UNCLASSIFIED</td>
<td>UNCLASSIFIED</td>
<td>UNCLASSIFIED</td>
</tr>
</tbody>
</table>

19b. TELEPHONE NUMBER (include area code)

(310)336-2906

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. 239.18
Preface

This briefing report contains a series of five lectures on topics in software acquisition prepared for SMC’s Acquisition/Logistics Excellence Week 2001. The lectures were scheduled to be presented on September 11, 2001 but were not delivered due to the national emergency. To disseminate this information to as wide an audience as possible, two actions were taken. The first action was to present the briefings in the Computer Systems Division (CSD) Technical Forum series during FY02. The second action is the publication and distribution of this briefing report.

The topics in software acquisition covered by these lectures are critically important to SMC and NRO programs in today’s acquisition environment. Software acquisition is always a high risk endeavor for SMC’s and NRO’s software-intensive systems. This series of lectures provides specific, practical suggestions that can be applied immediately to SMC and NRO programs to mitigate software risks. Each lecture also contains a set of resources for use by the reader in obtaining additional information on the topics.

The five lectures contained in this briefing report are as follows:

- COTS-Based Systems: Lessons Learned from Experiences with Commercial Off-the-Shelf (COTS) Software Use on Space Systems
- Recommended Metrics for Use with Object-Oriented (OO) Techniques
- Joint Technical Architecture (JTA) and Common Operating Environment (COE): Interoperability and Affordability
- Selecting a Capable Software Contractor Team: Evaluation Techniques to Support Acquisition and Logistics Excellence
- Human Systems Integration Acquisition Policies and Concerns
Contents

Abstract ......................................................................................................................... 1
COTS-Based Systems: Lessons Learned from Experiences with COTS Software
   Use on Space Systems ......................................................................................... 2
Abstract ....................................................................................................................... 56
Recommended Metrics for Use with Object-Oriented Techniques ...................... 57
Abstract ....................................................................................................................... 109
JTA and COE: Interoperability and Affordability ...................................................... 110
Abstract ....................................................................................................................... 198
Selecting a Capable Software Contractor Team: Evaluation Techniques to Support
Acquisition and Logistics Excellence ......................................................................... 199
Abstract ....................................................................................................................... 288
Human Systems Integration: Acquisition Policies and Concerns ......................... 289
Abstract: Lessons Learned from Experiences with COTS Software Use on Space Systems

DoD 5000.2-R requires the exploitation of Commercial Off-the-Shelf (COTS) computer system products in DoD acquisitions. The incorporation of software into software-intensive systems brings promises of reduced cost and schedule and higher reliability and maintainability by using “proven” software. However, the reality of using COTS software can be very different! Last year, Aerospace software acquisition personnel performed a survey of SMC and NRO programs on their experiences with incorporating COTS software into their systems. Six major lessons learned were derived from the survey. This lecture presents the six lessons learned and provides recommendations for acquirers and developers for improving the acquisition and development of COTS-based systems.

Acquisition and Logistics Excellence Week 2001

COTS-Based Systems:
Lessons Learned from Experiences with COTS Software Use on Space Systems

Richard J. Adams
Suellen Eslinger
Software Engineering Subdivision

Contact: Suellen.Eslinger@aero.org
Acknowledgements

The COTS software study described in this briefing was sponsored by the Chief Engineer's Office (SMC/AXE). The remainder of the technical content, as well as the preparation of the briefing, was developed under the Mission-Oriented Investigation and Experimentation (MOIE) program (Software Acquisition Task).

Outline

• Definitions and Background
• The Realities of COTS Software
  ➢ Results of Aerospace COTS Software Study
• Recommendations
  ➢ For the Development Organizations
  ➢ For the Acquisition Organizations
• Conclusion
• Resources
• Acronyms
• Back-up Charts
  ➢ Evaluation Criteria for COTS and Reuse Software Products
FAR Definition of Commercial Item

"(a) Any item, other than real property, that is of a type customarily used for nongovernmental purposes and that --
   (1) Has been sold, leased, or licensed to the general public; or,
   (2) Has been offered for sale, lease, or license to the general public;

(b) Any item that evolved from an item described in paragraph (a) of this definition through advances in technology or performance and that is not yet available in the commercial marketplace, but will be available in the commercial marketplace in time to satisfy the delivery requirements under a Government solicitation;

(c) Any item that would satisfy a criterion expressed in paragraphs (a) or (b) of this definition, but for --
   (1) Modifications of a type customarily available in the commercial marketplace; or

(2) Minor modifications of a type not customarily available in the commercial marketplace made to meet Federal Government requirements. Minor modifications means modifications that do not significantly alter the nongovernmental function or essential physical characteristics of an item or component, or change the purpose of a process...

(d) Any combination of items meeting the requirements of paragraphs (a), (b), (c), or (e) of this definition that are of a type customarily combined and sold in combination to the general public;

..."

(h) A nondevelopmental item, if the procuring agency determines the item was developed exclusively at private expense and sold in substantial quantities, on a competitive basis, to multiple State and local governments."

FAR, Section 2.101
Commercial Off-the-Shelf Software Products

- According to the FAR, the following are allowed to be considered COTS software products:
  - Software packages without a single customer/user
  - Pre-release software packages
  - Alpha or beta test versions of software packages
  - Modified software packages
  - Software products developed with contractor funds and sold to State or local government agencies under competitive contracts but never offered for sale to the public
- The FAR definition cannot be changed in a contract.
- Each COTS software product must be managed according to its degree of risk.

COTS software products are not all created equal!

Outline

- Definitions and Background
- The Realities of COTS Software
  - Results of Aerospace COTS Software Study
- Recommendations
  - For the Development Organizations
  - For the Acquisition Organizations
- Conclusion
- Resources
- Acronyms
- Back-up Charts
  - Evaluation Criteria for COTS and Reuse Software Products
COTS-Based System Study

- **Definition:** A COTS-based system (CBS) is a system that contains commercial off-the-shelf software components as elements of the system.

- **Study Purpose:**
  - Synthesize and share lessons learned from actual CBS development and sustainment experiences on USAF Space and Missile Systems Center (SMC) and National Reconnaissance Office (NRO) programs
  - Provide recommendations to help mitigate the risks inherent in CBS development and sustainment

CBS Study Approach

- **Step 1**
  - Developed a COTS software experience questionnaire
  - Interviewed SMC/NRO program representatives using the questionnaire as a discussion framework
  - Obtained any available material on COTS software experiences and lessons learned

- **Step 2**
  - Consolidated the results into one experience list
  - Performed an iterative series of analysis and synthesis to identify significant lessons learned

- **Step 3**
  - Developed recommendations to help mitigate the risks inherent in CBS development and sustainment
COTS Software Capabilities are Essential!

- Today's complex military systems require the leverage provided by COTS software.
- COTS software enables focusing on meeting the military's unique needs.

Lesson Learned - 1

Critical aspects of CBS development and sustainment are out of the control of the customer, developer and user.

- Vendors are market driven.
  - The military is not the market.
  - The market may diverge from military needs.
- Vendors' strategies and market position may change.
  - Go out of business
  - Drop or de-emphasize products or platforms
  - Change the type and quality of customer support
  - Change or drop promised features, performance and updates
Lesson Learned - 1 (Cont.)

- **Product release quality and content are unpredictable.**
  - Focus on features to attract new customers
  - Focus on fixes for primary customers
  - Limited testing, including regression testing
  - Features added but performance degraded
  - Computer resource requirements increased
  - Incompatibility with other products introduced
  - Backward compatibility with earlier versions eliminated

- **Product release schedule may:**
  - Be time-to-market driven (frequency and release dates).
  - Slip features and fixes.
  - Depend on upgrades to other COTS software.

- **Product and service costs are market driven.**
  - Fees and fee structures may change (licenses and services).

---

Lesson Learned - 2

*Full application of system and software engineering is required throughout the CBS life cycle.*

- Systems and software engineering are still required.
  - Using COTS software only shortens part of the software life cycle.
  - Incorporating COTS releases requires a full development life cycle.
  - Conflicts among multiple COTS products add complexity.

- **The CBS architecture must support COTS software evolution/replacement.**
  - True “plug and play” capability does not exist.
  - Computer resource margin and growth path must be sufficient.

- **Hands-on prototyping in a system context is essential.**
  - Integration of COTS software can cause unexpected impacts.
  - e.g., performance degradation, product incompatibilities
Lesson Learned - 2 (Cont.)

- Safety, security and supportability must be designed into the CBS.
  - COTS software focuses on the commercial marketplace.
  - COTS software is designed independently, not as part of a system.
  - CBS vulnerabilities can be determined by the products used.

- Periodic evaluation of COTS software is required.
  - Perform initial evaluation for product selection and subsequent re-evaluation for product evolution.
  - Use multi-dimensional evaluation criteria, not just functionality.
    - e.g., unique military needs, legacy interfaces, vendor characteristics, operations concept, cost
  - Evaluate computer hardware and COTS software together.
  - Prepare backup strategies and contingency plans.

Lesson Learned - 3

CBS development and sustainment require a close, continuous and active partnership among the customer, developer and user.

- The customer, developer and user must be prepared to trade cost, schedule, performance and O&M concepts.
  - Must prioritize requirements initially and reassess as necessary
  - Must understand which requirements can be relaxed to achieve a COTS-based solution
  - May need to re-engineer O&M procedures to accommodate a COTS-based solution
  - May discover COTS limitations and incompatibilities at any point in the life cycle
Lesson Learned - 3 (Cont.)

- Customer, developer and user must be active partners to ensure:
  - Adequacy of major trade decisions.
  - Full understanding of the evolving CBS capabilities.
  - Acceptability of delivered CBS.

Lesson Learned - 4

*Every CBS requires continuous evolution throughout development and sustainment.*

- Currency with COTS software upgrades is essential.
  - Delaying upgrades exacerbates system impacts.
  - Vendors support only a limited number of past releases.
  - Technology turnover occurs every 12 to 18 months.
  - Hardware upgrades may require COTS software upgrades.

- External organizations or systems can drive COTS software upgrades, replacements or additions.
  - DII/COE changes
  - GOTS changes
  - Legacy military system interfaces
Lesson Learned - 4 (Cont.)

- **COTS software may need to be replaced or added at any time.**
  - Elimination of product support by vendor
  - Divergence of product from system needs
  - Increased costs for licenses or support services
  - Identification of unacceptable limitations or vulnerabilities
  - Changes in functionality or performance
  - New or modified user needs
- **Modifying COTS software should be a last resort.**
  - Constrains the CBS evolution path
  - Requires a long-term relationship with COTS vendor
  - Increases life cycle costs

Lesson Learned - 5

*Current processes must be adapted for CBS acquisition, development and sustainment.*

- **System and software engineering processes must:**
  - Provide robust COTS software evaluation and selection criteria.
  - Require iterative life cycle models with extensive prototyping.
  - Integrate top-down and bottom-up development methodologies.
  - Require incorporation of COTS upgrades during development.
  - Account for COTS software in safety, security and supportability.
  - Enhance configuration management for COTS software complexity.
- **Time and effort need to be reallocated.**
  - More to evaluation, prototyping and analysis
  - Less to software implementation
  - More to integration
Lesson Learned - 5 (Cont.)

- Customer and user processes must:
  - Mandate prioritization of user requirements.
  - Result in contracts compatible with contractor CBS processes.
  - Result in milestones compatible with CBS development schedules.
  - Allow flexible and efficient response to unexpected impacts.
  - Support the schedule variability of COTS software upgrades.
  - Provide standardized user safety certification and security accreditation.

- Standardized licensing processes are needed:
  - To support maintaining COTS software license currency.
  - To ensure suitability for military needs.
    - e.g., no expiring keys, no export restrictions

Lesson Learned - 6

Actual cost and schedule savings with CBS development and sustainment are overstated.

- Overlooked or underestimated tasks
  - Systems and software engineering
  - Hands-on prototyping
  - Integrated system training and documentation
  - Acquisition of COTS software in-depth knowledge
    - e.g., mentors, toolsmiths, vendor support
  - Component and system testing, including performance testing
  - Component and system regression tests with each upgrade
  - Related software changes to support COTS software upgrades
    - Glue code, database changes, configuration files
  - Developer/operator training needed for COTS upgrades
Lesson Learned - 6 (Cont.)

- Unexpected impacts
  - Changes to license or service fees
  - Conflicts with the vendor's market
    - e.g., vendor charges to fix problems, refuses to support upgrade/platform, charges for escrowing source code.
  - Identification of COTS software limitations or problems (possibly with each upgrade)
    - e.g., performance degradation, interface changes, version incompatibility, new bugs, increased computer resource usage, insufficient documentation, amount and complexity of glue code, need for additional newly developed software
  - Externally caused COTS software upgrades/replacements
  - Problems or incompatibilities discovered during integration
  - Interdependencies of COTS software upgrades

Outline

- Definitions and Background
- The Realities of COTS Software
  - Results of Aerospace COTS Software Study
- Recommendations
  - For Development Organizations
  - For Acquisition Organizations
- Conclusion
- Resources
- Acronyms
- Back-up Charts
  - Evaluation Criteria for COTS and Reuse Software Products
Recommendations for Developers - 1

• Thoroughly evaluate all COTS software products before committing to their use.
  ◆ Use a robust set of evaluation criteria (see back-up charts).
  ◆ Prioritize criteria according to program needs.
  ◆ Use hands-on prototyping in the evaluation.
  ◆ Consider availability of COTS software in the selection of COTS hardware.

• Prototype potential and selected COTS software products within the system context.
  ◆ Determine suitability for meeting requirements.
  ◆ Determine suitability for integrating into new system architecture.
  ◆ Determine the amount of software needed for “glue code” or wraps.
  ◆ Determine the compatibility among multiple COTS software products.

• Establish agreements with vendors/developers.
  ◆ Ensure licenses are suitable for the military application.
  ◆ Obtain needed vendor support.

Recommendations for Developers - 2

• Adapt software and systems engineering processes to include COTS software.
  ◆ Requirements analysis, design, implementation, integration and verification processes
  ◆ Specialty engineering processes (especially security, safety and supportability)
  ◆ Integral processes (e.g., configuration management; metrics; peer reviews; project planning, tracking and oversight)

• Develop the system and software architecture to:
  ◆ Support continuous evolution of COTS software products.
  ◆ Ensure the satisfaction of system safety, security, and supportability requirements.

• Work closely with the customer and user to
  ◆ Trade requirements against COTS software capabilities (especially derived requirements).
  ◆ Ensure the adequacy of O&M strategies.
Recommendations for Developers - 3

- Use iterative system/software life cycle models.
  - Define how COTS software products fit into the builds.

- Plan for maintaining currency with COTS software releases during both development and sustainment.
  - Define how COTS software upgrades fit into the builds.

- Thoroughly test COTS software (development testing).
  - Full functional and integration testing for all COTS software products
    - Include testing of additional functionality of the COTS software products that is not required for the new system but is not removed or disabled.
  - Full regression testing of every COTS software upgrade within the system context

- Verify all requirements allocated to COTS software (qualification testing).
  - Full qualification testing (including all test methods I, A, D, T and all test levels) for all requirements, whether satisfied by COTS, unmodified reuse, modified reuse, or newly developed software
    - Each applicable test method at each applicable test level

Recommendations for Developers - 4

- Incorporate COTS software considerations into the cost and schedule estimates.
  - All necessary COTS software-related activities
    - Don't shortchange software and systems engineering!
  - All COTS software-related costs

- Ensure sufficient cost and schedule management reserves exist to cover unexpected COTS software problems.

- Evaluate technical, cost and schedule risk associated with the selected COTS software products.
  - Develop risk mitigation and contingency plans.

- Don't modify COTS software!
Recommendations for Acquirers - 1

- Consider COTS software when defining the acquisition strategy.
  - Determine whether the use of COTS software will be actively encouraged.
  - Determine the software maintenance strategy (near-term and long-term).
  - Determine how to achieve the necessary cost and schedule management reserve.

- Adapt acquisition team processes to support the use of COTS software.
  - Requirements prioritization in partnership with the user
    - Perform market analysis to understand COTS software capabilities.
    - Determine which requirements are military essential versus which requirements can be traded against COTS software capabilities.
  - Be prepared to make flexible and efficient responses to impacts caused by unexpected problems with COTS software.
    - Cost, schedule and performance trades may be needed at any point in time.
  - Establish a close, continuous and active partnership with the developer and user.

Recommendations for Acquirers - 2

- Contract for the following:
  - A COTS software upgrade strategy during development as well as sustainment
  - A balanced solution among COTS, reuse and new development to meet cost, schedule and performance objectives
    - Don't say “maximize the use of COTS.”
  - Integrated Product and Process Development (IPPD) to encourage a close, continuous and active partnership among the customer, developer and user
  - Full life cycle systems and software engineering
  - Additional emphasis on safety, security and supportability

- Don’t use FAR 12 procurements for software-intensive ground systems.
  - Ground systems for space applications are almost never commercial items in themselves.
  - They are large, complex software-intensive systems some of whose components contain COTS software.
Recommendations for Acquirers - 3

- **Develop the RFP to support the COTS software strategy.**
  - Include the goals of the acquisition strategy with respect to COTS software in the statement of objectives.
  - Include evaluation criteria concerning COTS software issues in the Mission Capability evaluation factor (Section M) and corresponding proposal preparation instructions in Section L.
  - Require O&M documentation at the system level (not just COTS software documentation).
  - Ensure the RFP does not constrain the contractor to processes and schedule unsuitable for the use of COTS software (e.g., waterfall life cycle models, milestone schedules with inflexible accomplishment criteria, fixed milestone schedules incompatible with iterative life cycle models).

- **Perform an SDCE during source selection including questions and criteria to evaluate the bidders’ COTS software capabilities.**
  - A set of questions and criteria has recently been developed for this purpose.

Recommendations for Acquirers - 4

- **Determine whether the bidders have included all costs associated with COTS software in their cost proposals.**
  - Effort for COTS software prototyping
  - COTS software upgrades during development
  - Enough software and systems engineering effort
  - Enough effort for testing (integration, verification, regression)
  - Costs for training, licenses, vendor support

- **Identify the cost risk associated with the COTS software.**
  - Overly optimistic estimates of amount of COTS software that can be used
  - Underestimation of amount of “glue” code needed
  - Overly optimistic software productivity numbers that do not take into account:
    - Learning curves for new COTS software products
    - Effort to handle the uncertainties and unexpected problems that occur with COTS software
Recommendations for Acquirers - 5

- Participate in the developer's evaluation of COTS software.
  - Use Aerospace test beds to assist with COTS software product evaluation within the system context.
  - Characterization/stress testing, prototyping, reliability evaluation
  - Testability and maintainability evaluation
  - Vulnerability analysis (security)
  - Provide lessons learned from experience with specific COTS software products on other programs.

Prepare for the continual inherent uncertainties in CBS development and sustainment!

Recommended Government Initiatives

- Repository for actual experiences with COTS software products
  - Not based on vendor literature!
- Guidance for CBS life cycle cost and schedule estimation
  - Current models are not adequate.
- Specific acquisition guidance for CBS procurements
  - Recommended language for RFPs and technical requirements documents
- Standardized processes for safety certification and security accreditation that include COTS software
- Standardized Government licenses for COTS software that address military needs
Outline

- Definitions and Background
- The Realities of COTS Software
  - Results of Aerospace COTS Software Study
- Recommendations
  - For the Development Organizations
  - For the Acquisition Organizations
- Conclusion
- Resources
- Acronyms
- Back-up Charts
  - Evaluation Criteria for COTS and Reuse Software Products

CBS benefits are achieved only by careful preparation and execution!
Conclusion

• Affordability constraints drive the increasing reliance on COTS software.
  ❖ Systems are now too large and costly to build completely from scratch.
• However, use of COTS software needs to be considered a risk area that must be managed throughout the system life cycle.
• Risk mitigation efforts appropriate for COTS software need to be put into place during both the pre- and post-contract award periods.

The Government and contractors need to become an intelligent buyers and developers of COTS-Based Systems.

Outline

• Definitions and Background
• The Realities of COTS Software
  ❖ Results of Aerospace COTS Software Study
• Recommendations
  ❖ For the Development Organizations
  ❖ For the Acquisition Organizations
• Conclusion
• Resources
• Acronyms
• Back-up Charts
  ❖ Evaluation Criteria for COTS and Reuse Software Products
The Software Engineering Institute’s COTS-Based Systems Initiative

- The SEI has developed extensive guidance on COTS-Based Systems.
  - Information can be found on-line at http://www.sei.cmu.edu/cbs/cbs_description.html
- The COTS-Based Systems Initiative has developed a series of monographs giving guidance on COTS for DoD programs.
  - Information can be found on-line at http://www.sei.cmu.edu/cbs/monographs.html
- Another interesting publication available from the COTS-Based Systems Initiative website:

Other Recent DoD References

- The Air Force Scientific Advisory Board (SAB) has published the results of their study on COTS in a report and a briefing.
  - The report and briefing are available on-line at http://www.sab.hq.af.mil/Archives/index.htm
- OSD Guide
  - Developed by the SEI for OSD
  - Commercial Item Acquisition: Considerations and Lessons Learned, June 26, 2000, Report # TBD.
Aerospace Publications


  ❖ Includes paper and briefing charts

Aerospace Publications (Continued)


  ❖ To be available on-line at http://www.aero.org/publications/papers/tech-reports.html
Aerospace Resources

- Computer Systems Division (CSD)
  - All three subdivisions have extensive experience with various COTS software packages.
  - All three subdivisions have test bed resources that can be used to evaluate COTS and reuse software.
- Rami Razouk, General Manager of CSD, x66644
- Mike Campbell, Principal Director, Computer Science and Technology Subdivision, x61850
- Fred Pollack, Principal Director, Computer Engineering Subdivision, x65938
- Mary Rich, Principal Director, Software Engineering Subdivision, x65313
- Suellen Eslinger, Distinguished Engineer, Software Engineering Subdivision, x62906
- Richard J. Adams, Senior Engineering Specialist, Software Engineering Subdivision, x62907

Outline

- Definitions and Background
- The Realities of COTS Software
  - Results of Aerospace COTS Software Study
- Recommendations
  - For the Development Organizations
  - For the Acquisition Organizations
- Conclusion
- Resources
- Acronyms
- Back-up Charts
  - Evaluation Criteria for COTS and Reuse Software Products
### Acronyms - 1

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBS</td>
<td>COTS-Based System</td>
</tr>
<tr>
<td>COE</td>
<td>Common Operating Environment</td>
</tr>
<tr>
<td>COTS</td>
<td>Commercial Off-the-Shelf</td>
</tr>
<tr>
<td>CSD</td>
<td>Computer Systems Division</td>
</tr>
<tr>
<td>DII</td>
<td>Defense Information Infrastructure</td>
</tr>
<tr>
<td>DoD</td>
<td>Department of Defense</td>
</tr>
<tr>
<td>FAR</td>
<td>Federal Acquisition Regulations</td>
</tr>
<tr>
<td>GOTS</td>
<td>Government Off-the-Shelf</td>
</tr>
<tr>
<td>I, A, D, T</td>
<td>Inspection, Analysis, Demonstration, Test</td>
</tr>
<tr>
<td>IPPD</td>
<td>Integrated Product and Process Development</td>
</tr>
<tr>
<td>MOIE</td>
<td>Mission-Oriented Investigation and Experimentation</td>
</tr>
<tr>
<td>NRO</td>
<td>National Reconnaissance Office</td>
</tr>
<tr>
<td>O&amp;M</td>
<td>Operations and Maintenance</td>
</tr>
<tr>
<td>OSD</td>
<td>Office of the Secretary of Defense</td>
</tr>
<tr>
<td>RFP</td>
<td>Request for Proposal</td>
</tr>
</tbody>
</table>

### Acronyms - 2

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S/W</td>
<td>Software</td>
</tr>
<tr>
<td>SAB</td>
<td>Scientific Advisory Board</td>
</tr>
<tr>
<td>SDCE</td>
<td>Software Development Capability Evaluation</td>
</tr>
<tr>
<td>SEI</td>
<td>Software Engineering Institute</td>
</tr>
<tr>
<td>SMC</td>
<td>Space and Missile Systems Center</td>
</tr>
<tr>
<td>STC</td>
<td>Software Technology Conference</td>
</tr>
<tr>
<td>TBD</td>
<td>To Be Determined</td>
</tr>
<tr>
<td>TR</td>
<td>Technical Report</td>
</tr>
<tr>
<td>USAF</td>
<td>United States Air Force</td>
</tr>
</tbody>
</table>
Outline

- Definitions and Background
- The Realities of COTS Software
  - Results of Aerospace COTS Software Study
- Recommendations
  - For the Development Organizations
  - For the Acquisition Organizations
- Conclusion
- Resources
- Acronyms
- Back-up Charts
  - Evaluation Criteria for COTS and Reuse Software Products

Key Criteria for Evaluating COTS and Reuse Software Products - 1

- Ability to provide required capabilities and meet required constraints
  - Ability to satisfy requirements
  - Ability to achieve necessary performance, especially with realistic operational workloads
  - Appropriateness of algorithms in the COTS/reuse software for use in the new system
  - As evidenced by characterization/stress testing within the system context to determine capabilities and performance

- Ability to provide required protection (safety, security, and privacy)
  - Provided inherently in the COTS or reuse software product, or
  - Able to be provided around the COTS/reuse software by system design and implementation
Key Criteria for Evaluating COTS and Reuse Software Products - 2

- **Reliability/maturity**
  - As evidenced by an established track record
  - As evidenced by prototype evaluation within the system context

- **Testability**
  - As evidenced by the ability to identify and isolate faults

- **COTS or reuse software supplier viability**
  - Compatibility of COTS or reuse supplier’s future direction with program needs
  - Supplier long-term commitment to software COTS or reuse product
  - Supplier long-term business prospects
  - Type of supplier support available
  - Quality of supplier support available

Key Criteria for Evaluating COTS and Reuse Software Products - 3

- **Suitability for incorporation into the new system architecture**
  - Compatible software architecture and design features
  - Absence of obsolete technologies
  - Need for re-engineering and/or additional code development (e.g., wraps, “glue” code)
  - Compatibility among the set of COTS software packages
  - As evidenced by prototyping within the system context (e.g., to determine compatibility, wraps, “glue” code)

- **Ability to remove or disable features/capabilities not required in the new system**
  - Impact if those features cannot be removed/disabled or are not removed/disabled
  - As evidenced by prototyping within the system context
Key Criteria for Evaluating COTS and Reuse Software Products - 4

- **Availability of personnel knowledgeable about the COTS/reuse product**
  - Training required
  - Hiring required
  - Vendor or third party support required

- **Acceptability of software product licensing and data rights**
  - Restrictions on copying/distributing the software or documentation
  - License or other fees applicable to each copy
  - Acquirer's usage and ownership rights, especially to the source code
    - Ability to place source code in escrow against possibility of vendor/developer going out of business
  - Warranties available
  - Absence of unacceptable restrictions in standard license

---

Key Criteria for Evaluating COTS and Reuse Software Products - 5

- **Maintainability, including:**
  - Likelihood the software product will need to be changed to meet requirements
  - Feasibility/difficulty of accomplishing that change, if changes are to be made by the program reusing the software product
    - Quality of design and code
    - Need for reengineering and/or restructuring
  - Feasibility/difficulty of accomplishing that change, if changes are to be made by the vendor or product developer (e.g., for COTS or proprietary software)
    - Priority of changes required by this program versus other changes being made
    - Likelihood that the changed version will continue to be maintained by the vendor/developer
    - Likelihood of being able to modify future versions to include changes
    - Impact on life cycle costs
    - Impact if the current version is not maintained by the vendor/developer or changes are not able to be incorporated into future versions
Key Criteria for Evaluating COTS and Reuse Software Products - 6

- **Impacts of upgrades of the COTS/reuse software**
  - Frequency of COTS/reuse upgrades/modifications being made by the vendor/developer (i.e., of a new version being released) after a particular version has been incorporated into new system
  - Feasibility/difficulty of incorporating the new version of the COTS/reuse product into the new system
  - Impact if the new version is not incorporated
  - Ability of the new architecture to support the evolution of COTS/reuse software products

- **Interoperability with other system and system-external elements**
  - Compatibility with system interfaces
  - Adherence to standards (e.g., open systems interface standards)

Key Criteria for Evaluating COTS and Reuse Software Products - 7

- **Compatibility of the planned upgrades of COTS or reuse software with software development plans and schedules**
  - Compatibility of planned upgrades with build content and schedules
  - Impact on development cost and schedule to incorporate upgrades
  - Dependencies among COTS and reuse software packages
    - Potential for an incompatible set of COTS and reuse software packages
    - Potential for schedule delays until all dependent COTS and reuse software products are upgraded

- **Availability and quality of documentation and source files**
  - Completeness
  - Accuracy
Key Criteria for Evaluating COTS and Reuse Software Products - 8

- Criticality of the functionality provided by the COTS or reuse software
  - Availability of alternate source(s) for the functionality

- Short- and long-term cost impacts of using the COTS/reuse software product
  - Amount of management reserve needed to handle uncertainties
    - e.g., less COTS/reuse usable; more newly developed software required; COTS/reuse limitations identified

- Technical, cost, and schedule risks and tradeoffs in using the COTS or reuse software product
  - Ability to tolerate COTS or reuse software problems beyond the program's control at any point in the system life cycle
  - Ability to incorporate continuous evolution of COTS or reuse products during development and sustainment

Abstract: Recommended Metrics for Use with Object-Oriented Techniques

Object-oriented methodology is the latest in the software design philosophies to transition into the DoD software development community. In addition, a lifecycle management approach has been introduced that is based on the evolutionary development philosophy where the software lifecycle is iterated a number of times to develop multiple object-oriented products of increasing capability. The application of object-oriented methodology introduces some unique acquisition management challenges. This lecture defines 29 specific metrics that may be used for monitoring object-oriented design and development. In addition, guidance on which metrics would be useful to a program during the process of transitioning to object-oriented technology is provided. This lecture directly addresses the new DoD 5000.2-R requirements to use a software measurement process and an iterative software development process.
Recommended Metrics for Use with Object-Oriented Techniques

Linda Abelson
Software Systems Acquisition Department
Software Engineering Subdivision

Acknowledgements

The technical content, as well as the preparation, of this briefing was developed under the Mission-Oriented Investigation and Experimentation (MOIE) program (Software Acquisition Task).
Outline

⇒ Introduction to Metrics
  • What Makes Object-Oriented Metrics Different?
  • Object-Oriented Metrics Defined
  • Rules of Thumb for Metrics Selection for Object-Oriented Programs
  • Summary

What Metrics Provide

• High level visibility into health and status of the evolving software system

• Lower level data for timely problem detection, isolation, and impact assessment

• Summarization of all planning assumptions into quantitative entities

• Data to share with other disciplines
  Development cost and schedule estimation and tracking
  Reliability prediction
  Maintenance effort estimation
  Test planning efforts
The Metrics Process

- The metrics process supports the information needs of managers
- Management structures are not flat
- The infrastructure is planned to support measurement
- Metrics give you insight into projects and business functions
- Metrics must address the known and unknown information needs

Outline

- Introduction to Metrics
  ⇒ What Makes Object-Oriented Metrics Different?
- Object-Oriented Metrics Defined
- Rules of Thumb for Metrics Selection for Object-Oriented Programs
- Summary
The Object-Oriented (OO) Paradigm

- Our systems are getting more complex
- Object-Oriented Technology was developed to respond to this complexity
- Measured productivity indicates that the object technologies are improving our ability to field these systems

Software Complexity Continues to Increase

Two Distinct Aspects in Applying Object-Oriented Technology

- As a method of defining the requirements and design
- As a philosophy for managing the life cycle of the project

Each has Impacts on the Metrics Definitions.
Comparison Between Object-Oriented and Structured Methods

Structured Analysis Approach

"machine driven"

Object-Oriented Approach

"domain driven"

Comparison Between Waterfall "Once Through" and Evolutionary Life Cycles

- With the new focus on "real world" design representations
  - traditional step-wise refinement became obsolete
- A means of describing and managing in an incremental manner was required
Waterfall “Once Through” Life Cycle

- Waterfall Characteristics:
  - A sequence of product oriented activities: requirements definition through system testing
  - One activity is completed before proceeding to the next
  - The activities do not overlap
  - Needed modifications to an earlier activity cause rework in subsequent activities
  - Hardware and software are completed at the same time

OO Specific Iterative Development Life Cycle

Outline

- Introduction to Metrics
- What Makes Object-Oriented Metrics Different?
  ⇒ Object-Oriented Metrics Defined
- Rules of Thumb for Metrics Selection for Object-Oriented Programs
- Summary

Metrics for Use With Object-Oriented Techniques

- Need to manage multiple competing characteristics of a good object model (competing quality factors)
- Need to use application specific terminology to adequately reflect the state of the project
- Need to manage multiple builds and functionality

Object-Oriented Metrics are Different.
Overview of the Metrics Framework

*Growth and Stability* - measures the stability of the functionality required of the software

*Development Performance* - measures the capability of the developer to accomplish the job

*Product quality* - measures the ability of the delivered software product to support the users’ needs without failure

*Project Resource* - measures the balance between the work to be performed and the personnel resources assigned to the project

*Schedule and Progress* - measures the completion of milestones

*Technical Adequacy* - measures the viability of the technical approach

---

Object-Oriented Unique Issues

*Growth and Stability* - The three categories selected for modification include the Size, Requirements Volatility and the Design Volatility measurement categories. Each of these rely on measurement values that are uniquely described in the OO paradigm.

*Product quality* - The three categories selected for inclusion include: Inheritance, Object Structure and Coupling. These three measurement categories are new to the metrics framework.

*Schedule and Progress* - The five categories selected for modification and inclusion include: Milestone, Class Status, Use-Case Status, Build Content. The modified measurement category is the Milestone metric. This measurement category has been modified to reflect an evolutionary terminology in the milestone titles. The remaining four metrics are new to the metrics framework.
# Catalog of OO Unique Metrics

<table>
<thead>
<tr>
<th>ISSUE</th>
<th>CATEGORY</th>
<th>RECOMMENDED METRICS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Growth and</td>
<td><strong>1 Size</strong></td>
<td>Planned vs Actual Use Cases Completed</td>
</tr>
<tr>
<td>Stability</td>
<td></td>
<td>Planned vs Actual Classes Completed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Number of Attributes in a Class</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Number of Methods in a Class</td>
</tr>
<tr>
<td></td>
<td><strong>2 Requirements Volatility</strong></td>
<td>Added, Deleted and Modified Use Cases</td>
</tr>
<tr>
<td></td>
<td><strong>3 Design Volatility</strong></td>
<td>Added, Deleted and Modified Classes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added, Deleted and Modified Methods</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added, Deleted and Modified Attributes</td>
</tr>
<tr>
<td>Product Quality</td>
<td><strong>4 Inheritance</strong></td>
<td>Number of Children per Class</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Depth of Inheritance Tree per Class</td>
</tr>
<tr>
<td></td>
<td><strong>5 Object Structure</strong></td>
<td>Weighted Methods per Class</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Type of Methods in Class</td>
</tr>
<tr>
<td></td>
<td><strong>6 Coupling</strong></td>
<td>Coupling Between Object Classes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Response for a Class</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ISSUE</th>
<th>CATEGORY</th>
<th>RECOMMENDED METRICS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schedule and</td>
<td><strong>7 Milestone</strong></td>
<td>Planned vs Actual Milestone Days</td>
</tr>
<tr>
<td>Progress</td>
<td></td>
<td>Milestone Slip Ratio</td>
</tr>
<tr>
<td></td>
<td><strong>8 Class Status</strong></td>
<td>Planned vs Actual Classes Completed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Planned vs Actual Methods Completed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Planned vs Actual Attributes Completed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Class Traceability Status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Integration Test Traceability Status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Planned vs Actual Classes that have Successfully Passed Integration Test</td>
</tr>
<tr>
<td></td>
<td><strong>9 Use Case Status</strong></td>
<td>Planned vs Actual Use Cases Completed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Use Case Traceability Status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Functional Test Traceability Status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Planned vs Actual Use Cases that have Successfully Passed Functional Test</td>
</tr>
<tr>
<td></td>
<td><strong>10 Build Content - Classes</strong></td>
<td>Planned vs Actual Classes in Build</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ratio of Classes in Build</td>
</tr>
<tr>
<td></td>
<td><strong>11 Build Content - Use Cases</strong></td>
<td>Planned vs Actual Use Cases in Build</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ratio of Use Cases in Build</td>
</tr>
</tbody>
</table>
Outline

• Introduction to Metrics
• What Makes Object-Oriented Metrics Different?
• Object-Oriented Metrics Defined
⇒ Rules of Thumb for Metrics Selection for Object-Oriented Programs
• Summary

Selecting Object-Oriented Metrics

• Object-Oriented technology is currently in an industry-wide transition process
  ❖ Many contractors are challenged by Object-Oriented techniques
  ❖ Some contractors are successful; some are not
• How does a new technology become normal practice?
  ❖ Understanding the Technology Transition Process
• What differentiates a successful organization?
  ❖ Understanding the Technology Adoption Curve
Technology Transition Lifecycle

Adapted from: Rogers, Everett M., Diffusion of Innovation.

Metrics Useful During Awareness Phase

Problem: Organization has no experience applying this technology

<table>
<thead>
<tr>
<th>Goal</th>
<th>Question</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Define and stabilize technology process</td>
<td>Is there organizational experience in applying new technology?</td>
<td>• CMM Level</td>
</tr>
<tr>
<td>Stabilize product scope</td>
<td>Are requirements activities on track? Have the requirements stabilized?</td>
<td>• Use-Case Status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Requirements Volatility</td>
</tr>
<tr>
<td>&quot;Expert&quot; peer reviews</td>
<td>Who's on the project? How frequently do personnel change? Are the review activities on track?</td>
<td>• Staff Experience</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Staff Turnover</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Review Status</td>
</tr>
</tbody>
</table>

Basil and Weiss, A methodology for collecting valid software engineering data, IEEE Trans. on SW Eng., 1984
Plan vs Actual Use Cases Completed

This indicator provides an estimate of software size. Unplanned additions and changes to the number and magnitude of use cases can adversely influence schedules and costs.

Use Case Traceability Status

The traceability metric measures the degree to which software products have implemented requirements allocated from higher level specifications.
Functional Test Traceability Status

The traceability metric measures the degree to which the functional test cases cover the software use cases.

Plan vs Actual Use Cases Successfully Passed Functional Test

The Test Passed indicator monitors test progress during qualification phase of a software project. The criteria for determining whether a use case has been successfully tested must be well defined for this metric to be meaningful.
Added, Deleted and Modified Use Cases

This indicator can be used to monitor changes to requirements throughout a project, which can serve as a leading indicator of delays, cost increases and rework. The churn ratio (ratio of modified use cases to actual) is provided as an indicator of the amount of rework being accomplished.

Metrics Useful During Exploration Phase

Problem: Organization has limited experience applying the technology, but has studied the problem extensively.

<table>
<thead>
<tr>
<th>Goal</th>
<th>Question</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stabilize design and code</td>
<td>Are design activities on track?</td>
<td>• Class Status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Design Volatility</td>
</tr>
<tr>
<td>Define and monitor code quality criteria</td>
<td>Have quality criteria been defined?</td>
<td>• Inheritance</td>
</tr>
<tr>
<td></td>
<td>Are quality criteria being monitored?</td>
<td>• Object Structure</td>
</tr>
<tr>
<td></td>
<td>Is code quality within normal range?</td>
<td>• Coupling</td>
</tr>
<tr>
<td>“Expert” peer reviews</td>
<td>Who’s on the project?</td>
<td>• Staff Experience</td>
</tr>
<tr>
<td></td>
<td>How frequently do personnel change?</td>
<td>• Staff Turnover</td>
</tr>
<tr>
<td></td>
<td>Are the review activities on track?</td>
<td>• Review Status</td>
</tr>
</tbody>
</table>
Plan vs Actual Classes Completed

This indicator provides an estimate of software size. Unplanned additions and changes to the number and magnitude of classes can adversely influence schedules and costs.

Plan vs Actual Methods Completed

This indicator provides an estimate of class status. Unplanned additions and changes to the number and magnitude of methods can adversely influence schedules and costs.
Plan vs Actual Attributes Completed

This indicator provides an estimate of class status. Unplanned additions and changes to the number and magnitude of attributes can adversely influence schedules and costs.

Class Traceability Status

The class traceability status metric measures the degree to which software design products have implemented the software requirements.
Integration Test Traceability Status

The integration test traceability status metric measures the degree to which the integration test cases cover the software design.

Plan vs. Actual Classes that Have Successfully Passed Integration Tests

The Test Passed indicator monitors test progress during the integration and test phase of a software project. The criteria for determining whether a class has been successfully tested must be well defined for this metric to be meaningful.
Added, Deleted and Modified Classes

This indicator can be used to monitor changes to design throughout a project, which can serve as a leading indicator of delays, cost increases and rework. The churn ratio (ratio of modified classes to actual) is provided as an indicator of the amount of rework being accomplished.

Metrics Useful During Adoption Phase

**Problem:** Organization has limited (usually on a small scale) experience applying technology.

<table>
<thead>
<tr>
<th>Goal</th>
<th>Question</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stabilize product scope</td>
<td>Are requirements activities on track?</td>
<td>• Use Case Status</td>
</tr>
<tr>
<td></td>
<td>Have the requirements stabilized?</td>
<td>• Requirements Volatility</td>
</tr>
<tr>
<td>Stabilize product design</td>
<td>Are design activities on track?</td>
<td>• Class Status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Design Volatility</td>
</tr>
<tr>
<td>Define and monitor code</td>
<td>Have quality criteria been defined?</td>
<td>• Inheritance</td>
</tr>
<tr>
<td>quality criteria</td>
<td>Are quality criteria being monitored?</td>
<td>• Object Structure</td>
</tr>
<tr>
<td></td>
<td>Is code quality within normal range?</td>
<td>• Coupling</td>
</tr>
<tr>
<td>“Expert” peer reviews</td>
<td>Are the review activities on track?</td>
<td>• Review Status</td>
</tr>
</tbody>
</table>
A Note on Thresholds...

- All thresholds are provided as “rules of thumb”

- No threshold is absolute
  - Care must be taken to match thresholds to the needs of the program
  - A best practice is only “best” when it achieves results

- For the metrics where thresholds are shown:
  - Thresholds should be defined and adhered to for these metrics
  - Lack of any threshold for these metrics will make the time spent analyzing and collecting the metric meaningless

Number of Children per Class

![Number of Children per Class](chart)

This metric indicates the number of immediate subclasses (i.e., child classes) subordinated to a class in the class hierarchy. This metric measures the horizontal breadth of the class structure.
**Depth of Inheritance Tree per Class**

This metric measures the depth of inheritance (DIT) of the class. In cases involving multiple inheritance, the DIT will be the maximum length from the node to the root of the tree. This metric measures the vertical depth of the class structure.

**Weighted Methods per Class**

This metric describes the complexity of a class through the complexity of its methods. It is calculated by summing the McCabe complexity for each method in the class.

Type of Methods in Class

The number of public methods in a class is a measure of the amount of system functionality being provided by the class. In addition, the number of public methods is a reflection of the total number of methods provided by the class, because each public method is supported by some number of private methods.

Coupling Between Object Classes

This metric counts the number of other classes with which a particular class communicates via calls to methods in the other classes.
The response set of a class is the set of methods that can be invoked in response to a message sent to the class.

Technology Adoption Curve

Adapted from: Moore, Geoffrey A., Crossing the Chasm.
Using Metrics to Compensate for Technology Adoption Mismatches

<table>
<thead>
<tr>
<th>CATEGORY</th>
<th>Technology Phase/Staffed with</th>
<th>Awareness</th>
<th>Exploration</th>
<th>Adoption</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Innovators</td>
<td>Match</td>
<td>Match</td>
<td>Mismatch</td>
<td></td>
</tr>
<tr>
<td>2. Early Adopters</td>
<td>Match</td>
<td>Match</td>
<td>Match</td>
<td></td>
</tr>
<tr>
<td>3. Majority</td>
<td>Mismatch</td>
<td>Match</td>
<td>Match</td>
<td></td>
</tr>
<tr>
<td>4. Skeptics</td>
<td>Mismatch</td>
<td>Mismatch</td>
<td>Match</td>
<td></td>
</tr>
<tr>
<td>5. Laggards</td>
<td>Mismatch</td>
<td>Mismatch</td>
<td>Mismatch</td>
<td></td>
</tr>
</tbody>
</table>

Awareness & Exploration Phase being managed by Category 3, 4 and 5
- STRENGTH: Usually well versed in management
- WEAKNESS: Not technology savvy; may set up roadblocks.
- COMPENSATION STRATEGY: stress quality criteria

Adoption Phase being managed by Category 1
- STRENGTH: Technology advocates
- WEAKNESS: Not well versed in management; usually schedule slippages are common
- COMPENSATION Strategy: Stress progress metrics; ensure quality criteria adhered to

Outline

- Introduction to Metrics
- What Makes Object-Oriented Metrics Different?
- Object-Oriented Metrics Defined
- Rules of Thumb for Metrics Selection for Object-Oriented Programs

⇒ Summary
Summary

- Metrics are useful for
  - Development status
  - Development risks and issues
  - Evaluation of new technology applications
    - Choosing the right metrics is crucial
- Metrics selection for the OO technology
  - Identification of the technology transition lifecycle phase
  - Use of the technology adoption curve
- Metrics are a key management tool
  - OO metrics are an extension of traditional metrics

Resources: External Organizations

- Joint Logistics Commanders’ Joint Group on Systems Engineering—Practical Software Measurement (PSM)
  http://www.psmsc.com/
- Software Engineering Institute (SEI)
  http://www.sei.cmu.edu/
- Software Productivity Consortium (SPC)
  http://www.software.org/
- Software Program Managers Network (SPMN)
  http://www.spmn.com/
- Software Technology Support Center (STSC)
  http://www.stsc.hill.af.mil/
Resources: Aerospace

- Computer Systems Division
  - Software Engineering Subdivision
    - Suellen Eslinger, x62906
  - Software Systems Acquisition Department
    - Linda Abeloson, x67350
  - Computer Engineering Subdivision
    - Colleen Ellis, x61324

- Reconnaissance Systems Division
  - Software Engineering Department
    - Rita Creel (703) 633-5634

References

- Booch, Grady. *Object-Oriented Analysis and Design with Applications*.


Abstract: JTA and DII COE: Interoperability and Affordability

The Department of Defense has many ongoing initiatives to facilitate interoperability and affordability of DoD systems. The ones that most impact SMC include the Joint Technical Architecture (JTA), a set of information technology standards, defined interfaces, and rules; and the Common Operating Environment (COE), which is implemented via a set of modular software that provides generic functions or services, such as common support applications and infrastructure services, and a set of guidelines for their use. DoD 5000.2-R and other policy statements require compliance with all applicable JTA standards; JTA in turn includes compliance with COE among its mandates. This talk describes the JTA as well as the COE architecture and levels of runtime compliance. It also explains the process for implementing JTA on a program and the requirements for achieving COE compliance at Level 5 or above.

Acquisition and Logistics Excellence Week 2001

JTA and COE: Interoperability and Affordability

Judy Kerner
Software Engineering Subdivision

Charts updated as of May 2002
Acknowledgements

This briefing has evolved from a series of briefings that have been funded by many sources. The three most recent updates, including this ALEW 2001 briefing, have been developed under the Mission-Oriented Investigation and Experimentation (MOIE) program (Software Acquisition Task).

Outline

• Introduction
  • Motivation for Open Systems
  • JTA and COE Compared
• Joint Technical Architecture (JTA)
• Common Operating Environment (COE)
• Program Actions
• Resources
Anticipated and Unanticipated Interfaces

- Systems must evolve to meet changing requirements and environments
- If System A interfaces only with System B, one jointly created ICD should be sufficient
- But what if System A may be deployed with unanticipated systems - the current situation!
  - If a system is implemented with a standard interface, then it should be able to interface at least with other systems built to use the same standard interface
  - Implementing to common interface standards reduces the n-way problem to many one-interface mappings
Achieving Interoperability

By Interface Ctrl Docs (ICDs)  By common interface stds

- Each interface separately defined
- Evolutionary path prohibitive
  - Never gets cheaper
  - Each new system needs all I/Fs
- Minimal chance that additional system happens to be interoperable

- Std interface defined once, used by all
- Evolutionary path identified
  - Each system built to the same stds
  - Can evolve in sync if stds evolve
- Better chance that additional system will be interoperable

Can’t know in advance all the systems that will have to interoperate

Open Systems Are Based on Common Interface Standards

- An open system is a system that implements sufficient open standards for interfaces, services, and supporting formats to enable properly engineered components
  - to be utilized across a wide range of systems with minimal changes,
  - to interoperate with other components on local and remote systems, and
  - to interact with users in a style that facilitates portability.

[Adapted from OS-JTF Terms of Reference 1998 - emphasis added]

- Key characteristics: facilitates
  - Component portability
  - Component interoperability
  - User portability
JTA and COE: 
DoD Open Systems Initiatives

- **JTA: standards specifications for IT**
  - Mandates open standards and protocols
  - Tracks emerging improvements

- **COE: guidelines and common products**
  - Layered, modular infrastructure maximizes reuse
  - Based on industry standard hardware, government and commercial software

Outline

- **Introduction**
  - Motivation for Open Systems
  - JTA and COE Compared
- **Joint Technical Architecture (JTA)**
- **Common Operating Environment (COE)**
- Program Actions
- Resources
## JTA vs. COE

![Image from DII COE I&RTS V4.1](image)

Standards specifications

Common products

## Comparison of JTA and COE

<table>
<thead>
<tr>
<th></th>
<th>JTA</th>
<th>COE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Contents</strong></td>
<td>Industry and some military specifications and standards</td>
<td>Middleware and infrastructure software and utilities</td>
</tr>
<tr>
<td><strong>Features</strong></td>
<td>Interface specifications</td>
<td>Mostly open system products</td>
</tr>
<tr>
<td><strong>Software</strong></td>
<td>No software is identified except COE</td>
<td>Implemented using DISA-approved COTS and GOTS software</td>
</tr>
<tr>
<td><strong>Implementation Context</strong></td>
<td>Compliance with any standard required only if corresponding service is in system; JTA has additional applicability guidance for each standard</td>
<td>I&amp;RTS defines COE compliance levels and segmentation, provides rules for interaction among software components</td>
</tr>
<tr>
<td><strong>Mandate</strong></td>
<td>Mandated in DoD and DoD Component policies</td>
<td>Mandated in JTA, for C2, combat support, and intelligence systems</td>
</tr>
</tbody>
</table>
Outline

- Introduction
- Joint Technical Architecture (JTA)
  - Mandates
  - JTA Explained
- Common Operating Environment (COE)
- Program Actions
- Resources

JTA Scope and Evolution

- **JTA Version 1.0, August 1996**
  - C4I systems only
- **JTA Version 2.0, May 1998**
  - "...mandates the minimum set of standards and guidelines for the acquisition of all DoD systems that produce, use, or exchange information." [From Implementation Memo for DoD JTA Ver. 2.0]
- **JTA Version 3.0, November 1999**
  - Implementation memo incorporated JTA V2.0 memo
  - JTA V3.1 published March 2000
    - Only significant change was to move Gigabit Ethernet from Emerging status to Mandated status
- **Final JTA Version 4.0 put out on DISA Website April 2001**
  - Promulgation memo not yet released
- **JTA Version 5.0 development in progress**
  - Now in second of three phases, focusing on content
JTA Compliance in V2.0/V3.0 OSD Implementation Memo

- JTA is required for all emerging and changes to DoD systems that
  - Produce, use, or exchange information electronically;
  - Cross a functional or DoD Component boundary; and
  - Give the warfighter or DoD decision maker an operational capability
- Waivers possible for cost, schedule, or performance
  - Mandatory standards in JTA must be implemented by systems that have a need for the corresponding services
  - Specification of standards outside of JTA must be additive, complementary, and not conflict with JTA mandated standards
- Each DoD Component is responsible for implementation, to include
  - Compliance assurance
  - Programming and budgeting of resources
  - Scheduling

Policy for JTA Compliance

- DoD 5000.2-R (5 Apr 2002)
  - JTA is required for all new, or changes to existing, IT, including NSS
  - DoD CAE or cognizant OSD Principal Staff Assistant may grant a waiver if the use of a JTA mandated standard will negatively impact cost, schedule, or performance - waiver requests must detail impact
- CJCSI 6212.01B (8 May 2000)
  - National Security Systems and Information Technology Systems must comply with applicable IT standards contained in the current DoD JTA
- Air Force Implementation Plan for the DoD JTA (1 Dec 1998)
  - The DoD JTA is the technical architecture view for the DoD and is mandatory for use in the Air Force
  - The Joint Technical Architecture-Air Force (JTA-AF) is an extension of the DoD JTA and is a comprehensive source for Air Force standards guidance [Note - the JTA-AF is being replaced by the Infostructure Technology Reference Model (i-TRM)]
Outline

• Introduction
• Joint Technical Architecture (JTA)
  ❖ Mandates
  ❖ JTA Explained
• Common Operating Environment (COE)
• Program Actions
• Resources

Joint Technical Architecture (JTA)

• DoD mandate for interoperability standards and guidelines at system/component interfaces
  ❖ Facilitates joint and combined force operations
  ❖ Communicates DoD’s preference for open system, standards-based products and implementations
• JTA requires all mandated standards to meet selection criteria:
  ❖ Interoperability
  ❖ Technical maturity
  ❖ Implementability
  ❖ Public availability
    ❖ But a very few standards are classified
• Most standards in JTA are commercial standards
What is JTA?

- Technical Architecture: The minimal set of rules governing the arrangement, interaction and interdependence of system parts or elements, whose purpose is to ensure that a conformant system satisfies a specified set of requirements [Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance [C4ISR] Architecture Framework V2.0]
- DoD Joint Technical Architecture (JTA) defines the common interface standards for Information Technology (IT)
- In addition to the technical architecture (TA), the C4ISR Architecture Framework also defines operational architecture (OA) and system architecture (SA)
  - Joint OA (JOA) and Joint SA (JSA) are not available yet
  - In the absence of a JSA, the Defense Information Infrastructure (DII) COE is mandated

JTA Version 4.0 Hierarchy Model

[From JTA V4.0]
Structure of JTA Core

Sect. 1. Overview (DII COE I&RTS)
Sect. 2. Information Processing Stds (e.g. POSIX, SQL)
Intro Mandated
Emerging
Sect. 3. Information Transfer Stds (e.g. TCP/IP, HTTP)
Intro Mandated
Emerging
Sect. 4. Info Modeling, Metadata, Info Exchange Stds (e.g. IDEF0, UML, USMTF)
Intro Mandated
Emerging
Sect. 5. Human-Computer Interface Stds (e.g. DOD HCI Style Guide)
Intro Mandated
Emerging
Sect. 6. Info Security Stds (e.g. Common Criteria, SSL)
Intro Mandated
Emerging

JTA Ver. 4.0 Sect. 1
Overview and Policy Statements

- Section 1 contains the only policy in JTA:
  - DII COE Integration and Runtime Specification is mandated for Command and Control, Combat Support, and Intelligence Systems
  - Combined and Coalition Standardization and/or Interoperability documents are referenced
    ✦ DoD Directive 2010.6 and CJCSI 2700.0 (governing)
    ✦ DoDD 2010.6: Standardization and Interoperability of Weapon Systems and Equipment within NATO, 5 Mar 1980
    ✦ CJCSI 2700.01: International Military Rationalization, Standardization, and Interoperability between the US and Its Allies and Other Friendly Nations, 30 Jan 1995
    ✦ NATO Consultation, Command and Control Technical Architecture, 15 Dec 2000
    ✦ Allied Communications Publication 140, Combined Interoperability Technical Architecture, 3 May 1999
- JTA Sections 2 - 6 and the domain and subdomain annexes contain IT standards cited as either mandated or emerging
  - Selected JTA standards are in the following charts
Selected JTA Ver. 4.0 Sect. 2
Information Processing Standards

- User Interface: Motif, XCDE, X11R6
- Data Management: SQL
- Operating Systems: POSIX, Win32
- Remote/Distributed Computing: DCE, CORBA, IIOP
- Graphics: CGI, OpenGL
- Data/Video/Audio/Imagery Interchange: SGML, HTML, XML, GIF, RPF, VPF, WGS-84, CGM, JPEG, JFIF, NITF, MPEG-2, GRIB, BUFR

- Emerging Standards: SQL3, ODMG, XHTML™, XSL, VRML, PNG, RT POSIX, RT CORBA

Selected JTA Ver. 4.0 Sect. 3
Information Transfer Standards

- E-mail: ACP123, SMTP, MIME
- Directory Services: X.500, LDAP, DNS
- Application Spt: FTP, TELNET, DHCP, HTTP, URL
- Protocols: TCP, IP, UDP, BOOTP, SNMP, OSPF, CSMA/CD, PPP
- Networking: ISDN, ATM, GbE, SONET
- Multimedia/Video/Audio Stds, Facsimile Stds
- Satellite Comm: GPS, MILSATCOM, CCSDS Standards

- Emerging Standards: IPv6, SCPS, SAASM, RSVP
Selected JTA Ver. 4.0 Sect. 4
Information Modeling, Metadata, and Information Exchange Standards

- Modeling: IDEF0, IDEF1X, UML
- Data Definitions: DoD 8320.1-1M, DDDS
- Bit-Formatted Msgs: TADIL-J, LINK16, VMF
- Character-Formatted Messages: USMTF

- Emerging Standards: XMI, MIDS, LINK 22

Selected JTA Ver. 4.0 Sect. 5
Human Computer Interface Standards

- Style Guides:
  - DOD HCI Style Guide
  - DII User Interface Specification
  - Windows Style Guide
  - CDE 2.1/Motif Style Guides
- Human-Centred Design Processes: ISO 13407
- Symbology: MIL-STD-2525B

- Emerging Standards: GeoSym™
Selected JTA Ver. 4.0 Sect. 6
Information Security Standards

• Security: Common Criteria, Fortezza, FIPS PUB 140
• Authentication: Kerberos, FIPS PUB 112, X.509
• Algorithms: DES, DSA, KEA, SKIPJACK
• Protocols: S/MIME, KMP, SSL
• Network: SDN, Common Security Label

• Emerging Standards: TLS, RADIUS, S/MIME V3, 3DES, AES, SSH, PKCS, Protection Profiles for VPN, Firewall, IDS

Selected JTA Ver. 4.0 Domain and Subdomain Mandated Standards

• C4ISR Domain: NITF Extensions, NTSDS
• Modeling & Simulation Domain: HLA
• Combat Support Domain: CALS, IGES
• Weapon Systems Domain: IFF Standards
• Some subdomain standards are unique:
  ❖ DICOM
• Some standards are mandated in multiple subdomains:
  ❖ PCMCIA
  ❖ SCSI-2
Outline

- Introduction
- Joint Technical Architecture (JTA)
- Common Operating Environment (COE)
  - Mandates for COE
  - What is COE?
  - Segmentation
  - Status of COE
- Program Actions
- Resources

Motivation for COE

COE was originally a response to observations that:

- Certain functions are required for nearly all C2 systems
  - Mapping, track mgmt, comm interfaces, etc.
- These functions are built repeatedly but incompatibly
  - Even when the requirements are the same, or vary only slightly, between systems
- Extracting these common functions and implementing them as a set of extensible low-level building blocks could
  - Accelerate development schedules
  - Achieve substantial savings through software reuse
  - Significantly improve interoperability
    - Common software is used across systems for common functions

[Based on I&RTS V4.1]
The Paige Memo:
Original DII COE Mandate

... All UNIX-based C4I legacy systems, other than mainframe base systems, shall be Level 5 DII COE compliant. All new C4I emerging systems and upgrades shall be level 6 DII COE compliant with the goal of achieving level 7...

Signed by Emmett Paige, Jr. (ASD/C3I)
May 23, 1997

Mandated by the DoD JTA

The DII COE, as defined in the DII COE I&RTS, is fundamental to a Joint System Architecture (JSA). In the absence of a JSA, the JTA mandates that at a minimum, all Command and Control (C2), Combat Support, and Intelligence Systems supporting the Joint Task Forces (JTFs) and Combatant Commands will use the DII COE. All applications of a system that must be integrated into a DII platform shall be at least DII COE I&RTS Level 5-compliant ... with a goal of achieving Level 8.

JTA Version 4.0
2 April 2001
Reiterated in the Air Force Implementation Plan for DoD JTA

The DoD JTA mandates the use of the DII COE ... The DoD JTA further mandates that all applications of a system which must be integrated into the DII shall be at least DII COE I&RTS level 5 compliant ... with a goal of achieving level 8 (full DII COE compliance level).

AF Implementation Plan for DoD JTA
Version 2.0
1 December 1998

Outline

- Introduction
- Joint Technical Architecture (JTA)
- Common Operating Environment (COE)
  - Mandates for COE
  - What is COE?
  - Segmentation
  - Status of COE
- Program Actions
- Resources
What is COE?

- COE is implemented as an integrated collection of Commercial off the Shelf (COTS) and Government off the Shelf (GOTS) software components intended to support the end-user's "Mission Segment"
- The COE is not a system; it is a foundation for building systems
- The COE is a network-centric "plug and play" open architecture
- The COE is also an evolutionary acquisition and implementation strategy

COE Concept

- An architecture that is fully compliant with applicable guidelines
- An approach for building interoperable systems
- A reference implementation containing a collection of reusable software components
- A software infrastructure for supporting mission-area applications
- A set of guidelines, standards, and specifications that describe
  - How to reuse existing software
  - How to properly build new software so that integration is seamless and, to a large extent, automated

[I&RTS V4.1]
Component Sources and Ownership

- **Mission Applications**: Mission-unique functionality
  - Developed by: Services/Agencies, some commercial industry
  - Controlled and deployed by: Services and Agencies

- **Common Support Applications**: Functionality common within domains
  - Developed by: Services/Agencies, commercial industry
  - Controlled and deployed by: DISA

- **Infrastructure Services**: Functionality common across DoD
  - Developed by: Services/Agencies, DISA, commercial industry
  - Controlled and deployed by: DISA

- **COE Kernel**: Functionality needed for all applications
  - Developed by: commercial industry and DISA, but would like to evolve to all commercial implementation
  - Controlled and deployed by: DISA
Segments

- Segments are the most basic building blocks from which a COE-based system is built
- A segment is defined as a collection of related functions as seen from the perspective of the end user
  - Segments are defined in terms of the functionality they provide, not in terms of modules.
  - A segment may in fact consist of one or more modules.
- Segments consist of executable software, documentation and data, packaged in coherent self-contained units
  - Documentation and some of the data is for use in the development and/or integration environment.
  - Executable software and other data is for use in the runtime environment.
- All segments must be registered with DISA prior to deployment on COE
  - Segments that will be part of the COE have additional requirements
Abbreviated Segments

- Segments may be full or abbreviated
- Abbreviated segments (new with I&RTS Version 4.0)
  - Enable the use of COTS products that are difficult to repackage
  - Allow for the use of COTS original distribution media and installation instructions while preserving the benefits of COE segmentation
  - Do NOT result in COTS software modification
- Abbreviated segmentation is especially important in the NT environment due to the extensive software base of COTS applications
- Segmentation requirements are defined in the DII COE Integration and Runtime Specification

Outline

- Introduction
- Joint Technical Architecture (JTA)
- Common Operating Environment (COE)
  - Mandates for COE
  - What is COE?
  - Segmentation
  - Status of COE
- Program Actions
- Resources
Current COE Status

- Version 4.6 kernel was released for distribution Apr 02
  - Sun/Solaris Solaris 7 and 8
  - HPUX 11.0
  - Windows NT 4.0, Windows 2000
- Version 4.7 has been released for Beta testing
- Real-time kernel services were released May 01
  - Lynx OS/Power PC reference implementation
  - Real-time integration toolkit
  - Many real-time tasks were deferred due to lack of funding
- DISA is currently planning only “soft” Real-Time COE
  - Version 5.0 is planned to be a merged release (COE and Real-Time)

Future COE Products

- Future releases expected to include new capabilities, and platforms
  - RT COE planned for Version 5.0
    - Lynx OS/Power PC reference implementation
    - Kernel certification program (additional platforms) had 13 vendors express interest in validating their own COE-compatible kernel
    - Compaq and SGI have already validated for specific configurations
- Planned a major release every 24-30 months (e.g., 4.0 to 5.0)
  - “Patch” releases as required, about every 6 months

<table>
<thead>
<tr>
<th>APR 01</th>
<th>OCT 01</th>
<th>APR 02</th>
<th>OCT 02</th>
<th>APR 03</th>
<th>OCT 03</th>
</tr>
</thead>
<tbody>
<tr>
<td>V4.4</td>
<td>V4.5</td>
<td>V4.6</td>
<td>V4.7</td>
<td>V4.8</td>
<td>V4.9</td>
</tr>
</tbody>
</table>

- “Build List” available on DISA’s web page
Outline

- Introduction
- Joint Technical Architecture (JTA)
- Common Operating Environment (COE)
- Program Actions
  - Complying with JTA
  - COE Compliance
  - Considerations
- Resources

Complying with JTA

- Implementation of the JTA means use of applicable standards cited as mandated in the JTA
  - Required for new programs and major upgrades
- Compliance with all applicable JTA standards must be evaluated
  - Migration plans or waiver requests may be required
- JTA contains many industry standards that will be implemented regardless of the mandate
- COE compliance at any level is not sufficient to ensure JTA compliance
  - But COE is also required for many programs
Implementing JTA on a Program

- Develop a JTA profile for the system
  - To assess JTA compliance of an existing program
  - To plan for JTA compliance in a developing program
- To do this:
  - Create a table using the List of Mandated and Emerging Standards (LMES), formerly Appendix B
  - Use guidance provided in the JTA text to determine applicability:
    - For each service area, whether it is applicable to the program
    - For each applicable service area, which standard(s) within the service area are applicable
  - For each applicable standard, determine whether the system is/will be compliant with the standard
  - In the Comments column, note either the component/COTS product that implements the JTA standard, or the rationale for non-applicability/non-compliance

### Example JTA Standards Profile Entries

<table>
<thead>
<tr>
<th>JTA Section</th>
<th>Currently Mandated Standard</th>
<th>Applicable</th>
<th>Comply</th>
<th>Alternate Standard</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2.2.1.4.1 Document Interchange</td>
<td>ISO 8879-1986, SGML (with Amendment 1 and Technical Corrigenda 1 and 2)</td>
<td>N</td>
<td></td>
<td></td>
<td>No long-term document storage</td>
</tr>
<tr>
<td></td>
<td>HTML 4.01 Specification</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td>Help files</td>
</tr>
<tr>
<td></td>
<td>XML 1.0</td>
<td>Y</td>
<td>N</td>
<td>Proprietary format</td>
<td>Will transition to XML in upgrade</td>
</tr>
<tr>
<td>2.2.2.1.4.2 Graphics Data Interchange</td>
<td>JPEG File Interchange Format (JFIF), Version 1.02</td>
<td>N</td>
<td></td>
<td></td>
<td>Not exchanging imagery</td>
</tr>
</tbody>
</table>
Next Steps in Implementing JTA

- The JTA standards profile is a starting point for:
  - Familiarizing designers with relevant standards
  - Providing references for implementors
  - Developing compliance criteria for testing
  - Establishing customers' acceptance criteria
  - Creating waiver requests or migration plans if needed
- Compliance with JTA, and COE if applicable, must be in RFPs and in all relevant contractual documents

Recommendations for Adopting JTA

- Even when JTA is mandated, additional standards can be required for any program if they don't conflict with JTA standards
- Conflicts can arise when multiple standards and/or non-standard interfaces exist for the same services
- Customers and developers need continued awareness of relevant existing and proposed JTA standards
Next Steps for Ongoing Programs

- Identify and consolidate current standards used in requirements documents (e.g., ORD, TRD, ICDs)
- Compare with JTA list of standards and identify differences/exceptions
- Formulate a suggested tailored JTA list for the program architecture (system, segments, external interfaces)
- Work with contractors to incorporate key JTA standards into development effort whenever feasible
- Work with JTA Development Group to propose additions or changes for future versions of JTA
  - As appropriate, propose key standards used in program

Outline

- Introduction
- Joint Technical Architecture (JTA)
- Common Operating Environment (COE)
- Program Actions
  - Complying with JTA
  - COE Compliance
  - Considerations
- Resources
COE Compliance Categories

- **Runtime Environment**
  - How well software fits within COE executing environment
  - Degree to which software reuses COE components
- **Style Guide**
  - How well software operates from "look and feel" perspective
  - Documented in the User Interface Specifications (UIS), Oct 1999
- **Architectural Compatibility**
  - How well software fits within COE architecture
- **Software Quality**
  - Measures traditional software metrics

Compliance may be assessed in each category
  - I&RTS Appendix B contains detailed runtime compliance checklist

Levels of Runtime Compliance

- L8: Segment uses only public APIs, does not duplicate any functionality contained elsewhere in the system, fully UIS-compliant
- L7: Interoperable Compliance
  - L7: Access through published APIs, few if any private APIs, segments do not duplicate functionality in COE component segments
  - L6: Intermediate Compliance
  - L5: Minimal Compliance
  - L4: Bootstrap Compliance
  - L3: Platform Compliance
  - L2: Network Compliance
  - L1: Standards Compliance
  - L1: Capabilities share only common set of COTS standards
- L3: Two applications co-exist on the same platform as COE-based software
- L5: All segments share same Kernel COE; segments are registered, functionality is available via Executive Manager, native "look&feel"
- L4: All applications are in segment format, share COE kernel
- L6: Segments reuse one or more COE-component segments, UIS-compliant w/documentedit minor diffs
- L8: Full Dll Compliance

COE Compliance Testing

- For COE components
  - DISA does the testing and certification
- For mission application components
  - Services/Agencies do the testing and certification
  - Each Service/Agency has a different policy
  - AF process is very similar to the DISA process
  - AF COE lab (ESC/DI)
    - Provides information to the contractors about test procedures that should be used when testing for compliance
    - Works with the SPO and contractor on testing and certifying segments before they are deployed

Outline

- Introduction
- Joint Technical Architecture (JTA)
- Common Operating Environment (COE)
- Program Actions
  - Complying with JTA
  - COE Compliance
  - Considerations
- Resources
Policy Considerations

• JTA and COE compliance are not the same
  ❖ JTA mandates COE compliance for C2, CS, intel systems
  ❖ Scope and application are broader for JTA
    ✤ 5000.2-R: “all new, or changes to existing, IT”
  ❖ Impact on program architecture may be greater for COE
  ❖ Programs must understand difference between requirements for
    JTA compliance and for COE compliance

• Programmatic implications of compliance
  ❖ No additional funding for compliance
  ❖ Waiver Request and Migration Plan processes identified in Air
    Force Implementation Plan (AFIP)
  ❖ But AFIP leaves many ambiguities that need resolution for
    individual SPOs

Implementation Considerations

• Lag in incorporating new technology into JTA and COE
• Impact of JTA and COE evolution on program development
  ❖ The issue of when to switch versions must be addressed for both,
    but with very different considerations for each
    ✤ JTA standards go into contractual documents
    ✤ COE components are part of the delivered system

• Questions about maturity and long-term maintenance of
  COE components
  ❖ These are similar to considerations for adopting COTS

• COE compliance certification process may be difficult
  ❖ Aerospace/SMC working to get approval for local COE certification
    facility
Outline

- Introduction
- Joint Technical Architecture (JTA)
- Common Operating Environment (COE)
- Program Actions
- Resources
  - Aerospace Capabilities
  - Reference Information

Aerospace Participation in JTA and COE Evolution

- COE-related participation
  - Technical Working Groups (TWGs)
    - Real-Time Advisory Group (formerly Real-Time Working Group)
    - Security Services TWG
    - Common Operational Picture (COP)/Visualization TWG
  - COE Architecture Oversight Group (AOG), Configuration Review and Control Board (CRCB)
- JTA-related participation
  - JTA reviews
  - DoD JTA Development Group and subgroups
  - DoD JTA User Guide
  - Reviews of AF Implementation Plan for DoD JTA, JTA-AF, i-TRM
- Participation in other related activities
  - DoD Technical Reference Model (TRM) Working Group
Aerospace Program Support

- RFP template and other contractual language guidance
  - Aerospace has completed a draft template that covers RFP and contractual language for both JTA and COE
  - Organized according to the documents where JTA and COE language needs to be included
    - RFP: Section L, Section M, other
    - Related documents: ORD, TRD, CDRL, SOO, SOW, IMP, award fee plan, other
    - Background material that can be used wherever required
- Participation in JTA compliance assessments
- Assistance with COE implementation
- Information dissemination
  - Internal meetings
  - Email discussion lists
  - Briefings

Aerospace COE Testbed

- Goals
  - Provide an environment where customers can assess application software compatibility with COE
  - Understand segmentation process and COE impact on application performance
  - Evaluate COTS/GOTS products and COE tools
  - Test COE beta software
  - Demonstrate COE infrastructure and tools
  - Would like to become a secondary facility for COE compliance certification for the Air Force
- Status
  - Location: Aerospace Bldg. A3/2235 (Unclassified facility)
  - Configured for Unix and NT systems, RT hardware being considered
Outline

- Introduction
- Joint Technical Architecture (JTA)
- Common Operating Environment (COE)
- Program Actions
- Resources
  - Aerospace Capabilities
  - Reference Information
Web Sites

- DoD JTA Home Page  
  ❖ http://www-jta.itsi.disa.mil/
- DISA COE Home Page  
  ❖ http://diicoe.disa.mil/coe/
- USAF JTA Home Page (AF Implementation Plan, JTA-AF)  
  ❖ https://www.afca.scott.af.mil/jta-af/
- ESC JTA and COE Home Pages  
- SMC JTA Web Page  
  ❖ http://ax.losangeles.af.mil/chief_engineer/jta/nl_jta_sec02.htm

9-Aug-02

People

- Aerospace  
  ❖ Judy Kerner - (310) 336-6555  
  ❖ Brian Shaw - (310) 336-5134  
  ❖ Cheryl DeMatteis - (703) 633-5195  
  ❖ Matt Clark - (310) 336-1205  
  ❖ Sam Bowser - (703) 808-2492 (DSN 898-2492)  
  ❖ Jon Westergaard - (703) 808-2841 (DSN 898-2841)  
  ❖ Edward Aldava - (310) 336-0078
- Air Force  
  ❖ Nick Awwad SMC/AXE - (310) 363-0903 (DSN 833-0903)
Government Documents

- DoD Joint Technical Architecture
  - Version 3.1, 31 March 2000
  - Version 4.0, 2 April 2001
- DII COE Integration and Runtime Specification
  - Version 4.1, 3 October 2000
- DII COE User Interface Specifications
  - Version 4.0, October 1999
- JTA User Guide and Component JTA Management Plan
  - Version 1.0, 14 September 2001
- Air Force Implementation Plan for DoD JTA
  - Version 2.0, 1 December 1998
- Joint Technical Architecture-Air Force
  - Version 2.4, 7 June 2000
- DoD Technical Reference Model
  - Version 2.0, 9 April 2001

Backup Charts and Acronyms
JTA in DoD 5000.2-R  
April 5, 2002

- Implementation of the JTA is the use of applicable standards cited as mandated in the JTA.
- The implementation of the JTA is required for all new, or changes to existing, IT, including NSS.
- If the use of a JTA-mandated standard will negatively impact cost, schedule, or performance, a DoD CAE or cognizant OSD PSA may grant a waiver from use. For mission-critical or mission-essential programs, all granted waivers shall be submitted through ASD(C3I)/DoD CIO to USD(AT&L) for review.
- ... all requests for a waiver shall state the cost, schedule, and performance impacts that will occur if the waiver is not granted, and any resulting operational limitations.
  
  ✤ Mandatory Procedures for Major Defense Acquisition Programs and Major Automated Information System Acquisition Programs, Par. C.5.2.3.5.11.2.

---

JTA in CJCSI 6212.01B  
8 May 2000

- Chairman of the Joint Chiefs of Staff Instruction: Interoperability and Supportability of National Security Systems, and Information Technology Systems
- Information Technology Standards. New or modified NSS and ITS systems should be standards-based. NSS and ITS must comply with applicable information technology standards contained in the current DOD Joint Technical Architecture (JTA)
  
  ✤ Interoperability and Supportability of National Security Systems, and Information Technology Systems, 8 May 2000, Sect. 5, Par. h
Definitions of ITS and NSS in CJCSI 6212.01B

- Information Technology System (ITS). Any equipment or interconnected system or subsystem of equipment, that is used in the automatic acquisition, storage, manipulation, management, movement, control, display, switching, interchange, transmission, or reception of data or information. Information technology includes computers, ancillary equipment, software, firmware, and similar procedures, services (including support services), and related resources...

- National Security Systems (NSS). Telecommunications and information systems operated by the Department of Defense -- the functions, operation, or use of which (1) involves intelligence activities; (2) involves cryptologic activities related to national security; (3) involves the command and control of military forces; (4) involves equipment that is an integral part of a weapon or weapons systems; or (5) is critical to the direct fulfillment of military or intelligence missions...

COE Runtime Compliance Levels

- Level 1: Standards Compliance Level. A superficial level in which the proposed capabilities share only a common set of COTS standards. Sharing of data is undisciplined and minimal software reuse exists beyond the COTS. Level 1 may, but is not guaranteed to, allow simultaneous execution of the two systems.

- Level 2: Network Compliance Level. Two capabilities coexist on the same LAN but on different CPUs. Limited data sharing is possible. If common user interface standards are used, applications on the LAN may have a common appearance to the user.

- Level 3: Platform Compliance Level. Environmental conflicts have been resolved so that two applications may reside on the same LAN, share data, and coexist on the same platform as COE-based software. The COE kernel, or its equivalent, must reside on the platform. Segmenting may not have been performed, but some COE components may be reused. Applications do not use COE services (except for kernel services if the COE kernel is loaded) and are not necessarily interoperable.

[From I&RTS V4.1]
COE Runtime Compliance Levels (cont.)

- **Level 4: Bootstrap Compliance Level.** All applications are in segment format and share the COE kernel. Segment formatting allows automatic checking for certain types of application conflicts. Use of COE services is not achieved and users may require separate login accounts to switch between applications.

- **Level 5: Minimal DII Compliance Level.** All segments share the same COE kernel and functionality is available via the Executive Manager. Boot, background, session, and local processes are specified through the appropriate segment descriptors. (See Chapter 6 for a description of the types of processes.) Segments adhere to the basic "look and feel" of the native GUI, as defined in User Interface Specifications for the DII. Segments are registered and available through the online library. Applications appear integrated to the user, but there may be duplication of functionality and full interoperability is not guaranteed. Segments may be successfully installed and removed through the COE installation tools. Database segments are separated from application software, categorized according to their potential for sharing, and can peacefully coexist on a COE Data Server with other segments.

COE Runtime Compliance Levels (cont.)

- **Level 6: Intermediate DII Compliance Level.** Segments reuse one or more COE-component segments. Substantial security requirements are imposed upon segments at this level. Minor documented differences may exist between User Interface Specifications for the DII and the segment's GUI implementation. Database schema, business rules, valid values, element definitions, and other features associated with a database segment are fully documented.

- **Level 7: Interoperable Compliance Level.** Segments reuse COE-component segments to ensure interoperability. These include COE-provided communications interfaces, message parsers, database segments, track data elements, and logistics services. Access is through published APIs with documented use of few, if any, private APIs. Segments do not duplicate any functionality contained in COE-component segments. The data associated with a database segment is consistent with the logical data model for the applicable Community of Interest.
**COE Runtime Compliance Levels (cont.)**

- **Level 8: Full DII Compliance Level.** Proposed new functionality is completely integrated into the system (e.g., makes maximum possible use of COE services) and is available via the Executive Manager. The segment is fully compliant with the User Interface Specifications for the DII and uses only published public APIs. The segment does not duplicate any functionality contained elsewhere in the system whether as part of the COE or as part of another mission application or database segment. The data associated with a database segment is coordinated with the Defense Data Model (DDM) and does not overlap any existing COE component database segments.

---

**JTA Development Group Membership**

- Ballistic Missile Defense Organization (now Missile Defense Agency)
- Defense Advanced Research Projects Agency
- Defense Information Systems Agency
- Defense Intelligence Agency
- Defense Logistics Agency
- Defense Modeling and Simulation Office
- Defense Threat Reduction Agency
- Joint Staff/J6
- National Imagery and Mapping Agency
- National Reconnaissance Office
- National Security Agency
- Office of the Assistant Secretary of Defense
- Office of the Under Secretary of Defense OSJTF
- U.S. Air Force
- U.S. Army
- U.S. Coast Guard
- U.S. Marine Corps
- U.S. Navy
- U.S. Special Operations Command
- U.S. Transportation Command
Acronyms

- ACP  Allied Communications Publication
- AES  Advanced Encryption Standard
- AFIP  Air Force Implementation Plan
- AIS  Automated Information Systems
- ALEW  Acquisition and Logistics Excellence Week
- AOG  Architecture Oversight Group
- API  Application Program Interface
- ASD  Assistant Secretary of Defense
- AT&L  Acquisition, Technology, and Logistics
- ATM  Asynchronous Transfer Mode
- BOOTP  Bootstrap Protocol
- BUFR  Binary Universal Format for Representation
- C2  Command and Control
- C3  Consultation, Command, and Control
- C3I  Command, Control, Communications, and Intelligence
- C4I  Command, Control, Communications, Computers, and Intelligence

Acronyms (cont.)

- C4ISR  Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance
- CAE  Component Acquisition Executive
- CALS  Continuous Acquisition and Life Cycle Support
- CCEB  Combined Communications Electronic Board
- CCSDS  Consultative Committee for Space Data Systems
- CDE  Common Desktop Environment
- CDRL  Contract Data Requirements List
- CGI  Computer Graphics Interfacing
- CGM  Computer Graphics Metafile
- CINC  Commander-in-Chief
- CIO  Chief Information Officer
- CITA  Combined Interoperability Technical Architecture
- CJCSI  Chairman of the Joint Chiefs of Staff Instruction
- COE  Common Operating Environment
- COP  Common Operational Picture
### Acronyms (cont.)

- CORBA Common Object Request Broker Architecture
- COTS Commercial off the Shelf
- CPU Central Processing Unit
- CRCB Configuration Review and Control Board
- CS Combat Support
- CSMA/CD Carrier Sense Multiple Access with Collision Detection
- DCE Distributed Computing Environment
- DDDS Defense Data Dictionary System
- DDM Defense Data Model
- DES Data Encryption Standard
- DHCP Dynamic Host Configuration Protocol
- DICOM Digital Imaging and Communications in Medicine
- DII Defense Information Infrastructure
- DISA Defense Information Systems Agency
- DNS Domain Name System
- DoD Department of Defense

---

### Acronyms (cont.)

- DoDD Department of Defense Directive
- DoDI Department of Defense Instruction
- DSA Digital Signature Algorithm
- ESC Electronic Systems Center
- FIPS Federal Information Processing Standard
- FTP File Transfer Protocol
- GbE Gigabit Ethernet
- GeoSym Geospatial Symbols for Digital Displays
- GIF Graphics Interchange Format
- GIG Global Information Grid
- GOTS Government off the Shelf
- GPS Global Positioning System
- GRIB Gridded Binary
- GUI Graphical User Interface
- HCI Human Computer Interface
- HIDAR High Data Rate
# Acronyms (cont.)

- **HLA**  
  High-Level Architecture  
- **HTML**  
  Hypertext Markup Language  
- **HTTP**  
  Hypertext Transfer Protocol  
- **HW**  
  Hardware  
- **HQ**  
  Headquarters  
- **ICD**  
  Interface Control Document  
- **IDEF**  
  Integration Definition  
- **IDS**  
  Intrusion Detection System  
- **I/F**  
  Interface  
- **IFF**  
  Identification Friend or Foe  
- **IGES**  
  Initial Graphics Exchange Specification  
- **IIOP**  
  Internet Inter-ORB Protocol  
- **IMP**  
  Integrated Management Plan  
- **IP**  
  Internet Protocol  
- **IPv6**  
  Internet Protocol, Version 6  
- **I&RTS**  
  Integration and Runtime Specification

191

# Acronyms (cont.)

- **ISDN**  
  Integrated Services Digital Network  
- **ISO**  
  International Organization for Standardization  
- **IT**  
  Information Technology  
- **i-TRM**  
  Infrastructure Technology Reference Model  
- **ITS**  
  Information Technology Systems  
- **JFIF**  
  JPEG File Interchange Format  
- **JPEG**  
  Joint Photographic Experts Group  
- **JOA**  
  Joint Operational Architecture  
- **JSA**  
  Joint System Architecture  
- **JTA**  
  Joint Technical Architecture  
- **JTA-AF**  
  Joint Technical Architecture-Air Force  
- **JTADG**  
  Joint Technical Architecture Development Group  
- **JTF**  
  Joint Task Force  
- **KEA**  
  Key Exchange Algorithm  
- **KMP**  
  Key Management Protocol  
- **LAN**  
  Local Area Network

192
### Acronyms (cont.)

- **LDAP** Lightweight Directory Access Protocol
- **LMES** List of Mandated and Emerging Standards
- **LOM** Learning Object Metadata
- **MAIS** Major Automated Information System
- **MDAP** Major Defense Acquisition Program
- **MIDS** Multi-functional Information Distribution System
- **MILSATCOM** Military Satellite Communications
- **MIME** Multipurpose Internet Mail Extensions
- **MOIE** Mission-Oriented Investigation and Experimentation
- **MPEG** Motion Pictures Expert Group
- **NATO** North Atlantic Treaty Organization
- **NC3TA** NATO C3 Technical Architecture
- **NITF** National Imagery Transmission Format
- **NSS** National Security Systems
- **NTSDS** National Target/Threat Signature Data System
- **OA** Operational Architecture

---

### Acronyms (cont.)

- **ODMG** Object Database Management Group
- **OSD** Office of the Secretary of Defense
- **OS-JTF** Open System Joint Task Force
- **OSPF** Open Shortest Path First
- **PCMCIA** Personal Computer Memory Card International Association
- **PKCS** Public Key Cryptography Standard
- **PNG** Portable Network Graphics
- **POSIX** Portable Operating System Interface
- **PPP** Point-to-Point Protocol
- **Pub** Publication
- **RADIUS** Remote Authentication Dial In User Service
- **RDF** Resource Description Framework
- **RFP** Request for Proposal
- **RPF** Raster Product Format
- **RSVP** Resource ReSerVation Protocol
- **RT** Real-Time
Acronyms (cont.)

- RTAG: Real-Time Advisory Group
- SA: System Architecture
- SAAHM: Selective Availability Anti-Spoofing Module
- SCPS: Space Communications Protocol Specification
- SCSI: Small Computer Systems Interface
- SDN: Secure Data Network
- SGML: Standard Generalized Markup Language
- SHADE: SHAred Data Environment
- SMC: Space and Missile Systems Center
- S/MIME: Secure/Multipurpose Internet Mail Extensions
- SMTP: Simple Mail Transfer Protocol
- SNMP: Simple Network Management Protocol
- SONET: Synchronous Optical Network
- SOO: Statement of Objectives
- SOW: Statement of Work
- SPO: System Program Office

Acronyms (cont.)

- SQL: Structured Query Language
- SSH: Secure Shell
- SSL: Secure Sockets Layer
- Stds: Standards
- SW: Software
- TA: Technical Architecture
- TADIL: Tactical Digital Information Link
- TCP: Transmission Control Protocol
- TELNET: Telecommunications Network
- TLS: Transport Layer Security
- TRD: Technical Requirements Document
- TRM: Technical Reference Model
- TWG: Technical Working Group
- UCA: Unified Cryptologic Architecture
- UDP: User Datagram Protocol
- UIS: User Interface Specifications
Acronyms (cont.)

- UML Unified Modeling Language
- URL Uniform Resource Locator
- USAF United States Air Force
- USD Undersecretary of Defense
- USMTF United States Message Text Format
- VMF Variable Message Format
- VPF Vector Product Format
- VPN Virtual Private Network
- VRML Virtual Reality Modeling Language
- WGS World Geodetic System
- XHTML Extensible HyperText Markup Language
- XMI XML Metadata Interchange
- XML eXtensible Markup Language
- XSL Extensible Stylesheet Language

Abstract: Selecting a Capable Software Contractor: Evaluation Techniques to Support Acquisition and Logistics Excellence

Several techniques are currently in use to appraise a software contractor’s development capability. Internally, contractors use the Software Engineering Institute’s Capability Maturity Model® for Software (SW-CMM®) to assess and improve their processes for software development. Contractor capability evaluations, such as the Software Engineering Institute’s Software Capability Evaluation (SCE℠) or the Air Force’s Software Development Capability Evaluation (SDCE), are used during source selection at SMC and the NRO to identify strengths, weaknesses and risks in the offerors’ software development capabilities and processes, and provide key discriminators in contractor selection. This presentation discusses the SW-CMM® assessment method and the two evaluation techniques, and will describe the similarities and differences between the SDCE and SCE. Additionally, the presenters will discuss the impact of recent DoD acquisition policy changes on the use of capability evaluations in the source selection process.
Selecting a Capable
Software Contractor Team:
Evaluation Techniques to Support
Acquisition and Logistics Excellence

Karen Owens
Bonnie Troup
Software Systems Acquisition Department
Software Engineering Subdivision

Acknowledgements

This work would not have been possible without assistance from the following:

Office of the Chief Engineer (Ted Winer)
MOIE Program (Software Acquisition Task)
Suellen Eslinger, Software Engineering Subdivision
Richard Adams, Software Systems Acquisition Department
Sharon Hoting, Space-Based Surveillance Division

and the members of the National IPT for Software Equivalence:
DUSD (S&T), DCMA, Air Force, Army, Navy, SEI, MITRE, IDA,
Boeing, Computer Sciences Corporation, Harris,
Litton PRC, Lockheed Martin, Northrop Grumman, Raytheon, TRW
Trademarks and Service Marks

- The Software Engineering Institute (SEI) is a federally funded research and development center established by the U.S. Department of Defense. SEI is part of Carnegie Mellon University and is sponsored by the Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics [OUSD (AT&L)].
- This work contains excerpts from
  - "An Executive Introduction to CMM-Based Software Process Improvement", © 1999 by Carnegie Mellon University
- ® - Capability Maturity Model, CMM, Capability Maturity Modeling are registered in the U.S. Patent and Trademark Office
- SM - CMM Integration, CMMI, SCE are service marks (SM) of Carnegie Mellon University
- No Warranty
  - Any material furnished by Carnegie Mellon University and Software Engineering Institute material is furnished on an "as-is" basis. Carnegie Mellon University makes no warranties of any kind, either expressed or implied, as to any matter including, but not limited to, warranty of fitness for purpose or merchantability, exclusivity, or results obtained from use of the material. Carnegie Mellon University does not make any warranty of any kind with respect to freedom from patent, trademark, or copyright infringement.

Outline

- Purpose of Contractor Capability Evaluations
- Evaluation Methods
  - Capability Maturity Model for Software
  - Software Capability Evaluation
  - Software Development Capability Evaluation
- Policy Change Effects on Contractor Evaluations
- Conclusions
- Resources
- Acronyms
- Backup Charts
  - Supplemental Capability Maturity Model Information
  - Comparison of the SDCE and SCE
Why the Government Uses Capability Evaluations

- Primary Purpose
  - Source Selection: Increase the likelihood of selecting a contractor capable of developing the required software within the program constraints

- Secondary Purposes
  - Risk Identification: Identify risks associated with the selected contractor(s) to facilitate managing these risks beginning at contract award
  - Contractual Commitment: Obtain a contractual commitment from the selected contractor to adopt methods, tools, and processes that instill and support software engineering discipline

DoD 5000.2-R* Acquisition Guidance

In paragraph 2.6.6.3, Applying Best Practices:
... Examples of practices that support the implementation of these policies include ... results of software capability evaluations ...

In paragraph 5.2.3.5.6.1, Software Management, General:
The PM shall base software systems design and development on systems engineering principles, to include the following:

5.2.3.5.6.1.5. Select contractors with domain experience in developing comparable software systems; with successful past performance; and with a mature software development capability and process ...”

*10 June 2001
Evaluation Considerations

TIME

BUDGET

PERSONNEL

Placement in Source Selection Structure

- Recommended placement is as a single "Software Process" Subfactor under the Mission Capability Factor, as shown

Factors

<table>
<thead>
<tr>
<th>Past Performance</th>
<th>Mission Capability</th>
<th>Proposal Risk</th>
<th>Price/Cost</th>
</tr>
</thead>
</table>

Subfactors

- Software Process Subfactor
- Subfactor 2
- Subfactor 6
Outline

- Purpose of Contractor Capability Evaluations
- Evaluation Methods
  - Capability Maturity Model for Software
  - Software Capability Evaluation
  - Software Development Capability Evaluation
- Policy Change Effects on Contractor Evaluations
- Conclusions
- Resources
- Acronyms
- Backup Charts
  - Supplemental Capability Maturity Model Information
  - Comparison of the SDCE and SCE

Assessment vs. Evaluation

**Assessment**
- Focus is on organization-wide process improvement
- Assessors within company or externally contracted company
- Company cherry picks projects
- Examines selected projects for compliance

**Evaluation**
- Focus is on selecting a capable contractor team, program risks, and contractual commitment
- Assessors from Government team
- Uses contractor selected programs from multiple companies for substantiation
- Examines subset of programs for compliance

Legend:
- Project A
- Project B
- Project C
- Project D
- Project E
- Project F
- Project G
- Project H
- Project I
- Project J
- Project K
- Project L
- Project M
- Project N
- Project P
- Project Q
- Project R
- Project S
- Project T
- Project U
- Project V
- Project W
- Project X
- Project Y
- Project Z
- Company A
- Company B
- Company C
- Company D
- Company E
- Company F
- Company G
- Company H
- Company I
- Company J
- Company K
- Company L
- Company M
- Company N
- Company O
- Company P
- Company Q
- Company R
- Company S
- Company T
- Company U
- Company V
- Company W
- Company X
- Company Y
- Company Z

Results are Organizationally Focused

Results are Program Focused
## Appraisal Techniques

<table>
<thead>
<tr>
<th>Air Force Technique</th>
<th>SEI Technique</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>Software Development Capability Evaluation (SDCE) Model</td>
</tr>
<tr>
<td>Method for Assessment</td>
<td></td>
</tr>
<tr>
<td>Method for Evaluation</td>
<td>SDCE Process</td>
</tr>
</tbody>
</table>

## Capability Maturity Model for Software

- **Description**
  - The Assessment Process
  - RFP and Proposal Considerations
  - Integrated Capability Maturity Models
Terminology: Software Process Maturity

- Process: An organized set of activities performed for a given purpose; for example, the software development process.*
- Software process: a set of activities, methods, practices, and transformations that people use to develop software and the associated products and services**
- The quality of a software system is highly influenced by the quality of the process used to develop and maintain it**
- Process maturity: the extent to which a process is explicitly defined, managed, measured, controlled and effective**


The Capability Maturity Model® for Software (SW-CMM®)

- A five-level model for process management and quality improvement established by the Software Engineering Institute (SEI)
- Defines the stages through which software organizations evolve as they improve their software process
  - Organized into five "maturity" levels
  - A maturity level is a well-defined evolutionary plateau on the path to becoming a mature software development organization
  - Each level is a layer in the foundation for continuous process improvement
The Five Maturity Levels

- **Initial**
  - Process unpredictable and poorly controlled

- **Repeatable**
  - Process characterized for projects and is often reactive

- **Defined**
  - Process characterized for organization and is proactive

- **Managed**
  - Process measured and controlled

- **Optimizing**
  - Focus on process improvement


---

Improving Predictability

- **Level 5** - performance continuously improves

- **Level 4** - quantitative analysis of process and product leads to improved performance

- **Level 3** - well defined processes improve performance

- **Level 2** - plans based on past schedule and cost performance more realistic

- **Level 1** - schedule and cost typically overrun

### SW-CMM Key Process Areas*

<table>
<thead>
<tr>
<th>Level</th>
<th>Focus</th>
<th>Key Process Areas</th>
<th>Quality Productivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Continuous Process improvement</td>
<td>Defect Prevention, Technology Change Management, Process Change Management</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Quantitative management</td>
<td>Quantitative Process Management, Software Quality Management</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Process standardization</td>
<td>Organizational Process Focus, Organization Process Definition, Training Program, Integrated Software Management, Software Product Engineering, Intergroup Coordination, Peer Reviews</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Basic project management</td>
<td>Requirements Management, Software Project Planning, Software Project Tracking &amp; Oversight, Software Subcontract Management, Software Quality Assurance, Software Configuration Management</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Competent people and heroics</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


---

**SW-CMM Structure**

- **Maturity Levels**
  - Key Process Area
  - Key Process Area
  - Key Process Area

- **Goals**

- **Institutionalization**
  - Common Features
    - Commitment to perform
    - Ability to perform
    - Measurement & analysis
    - Verification

- **Activities**
Capability Maturity Model for Software

- Description

- The Assessment Process
  - RFP and Proposal Considerations
  - Integrated Capability Maturity Models

The SW-CMM Assessment

- Developer assembles existing process documents and personnel to be interviewed
- Team assesses process "evidence"
- Usually done "in-house" or by a third party hired by the contractor
  - A well-documented process is used to perform the appraisal called the CMM-Based Appraisal for Internal Process Improvement (CBA-IPI)*
  - Appraisers have been appropriately trained and have performed assessments
  - Appraisal teams examine documented evidence of compliance and perform interviews on three projects in an organization
  - Two pieces of evidence (written or oral) are required for compliance
  - Appraisers are required to send results to the SEI

SW-CMM Assessment Rating Pitfalls

- Internal assessments may result in a higher level rating than is deserved.
- The proposed project may be at a lower SW-CMM level than the one claimed in the proposal.
- There is no guarantee that the contractor is using the assessed processes on your project.

Organization Maturity Profile
Recent Assessments*

* Based on most recent assessment data reported to SEI from 1020 organizations since 1996.

"Process Maturity Profile of the Software Community 2000 Year End Update", March 2001, SEI,
Maturity Profile by Organization Type*

<table>
<thead>
<tr>
<th>% of Organizations</th>
<th>Initial</th>
<th>Repeateable</th>
<th>Defined</th>
<th>Managed</th>
<th>Optimizing</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>34.3</td>
<td>48.8</td>
<td>42.4</td>
<td>34.1</td>
<td>32.6</td>
</tr>
<tr>
<td></td>
<td>21.9</td>
<td>4.8</td>
<td>13.8</td>
<td>11.6</td>
<td>26.6</td>
</tr>
<tr>
<td></td>
<td>36.7</td>
<td>5.8</td>
<td>4.4</td>
<td>2.6</td>
<td>1.2</td>
</tr>
</tbody>
</table>

Commercial/In-house  DoD/Fed Contractor  Military/Federal


Number of Assessments
Reported by Organization Type and Year*

<table>
<thead>
<tr>
<th>Year</th>
<th>Commercial/In-house</th>
<th>DoD/Fed Contractor</th>
<th>Military/Federal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1987</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1989</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1991</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1993</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1995</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1997</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1999</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Commercial/In-house  DoD/Fed Contractor  Military/Federal

* Based on 1507 assessments. Note: Other/Unknown category, pre-1995, not included.
Data from "Process Maturity Profile of the Software Community 1999 Year End Update", March 2000, SEI.
Time to Move Up a Level

Number of months to move to next maturity level

Recommended time between appraisals

Largest observed value that is not an outlier

75th Percentile
Median

25th Percentile
Smallest observed value that is not an outlier

Time Period of Initial Assessment
Pre-1992 1992 to Present All (1997 to Present)
Level
Orgs 1 to 2 2 to 3 1 to 2 2 to 3 3 to 4 4 to 5 1 to 2 2 to 3 3 to 4 4 to 5
1 24 12 1 2 12 3 15 2 12 143 115 17 12
2 193 103 2 3 18 4 17 2 5 17 17 17 12
3 22 22 2 4 22 5 22 2

Contains community-wide appraisal results.

Capability Maturity Model for Software

- Description
- The Assessment Process
- RFP and Proposal Considerations
- Integrated Capability Maturity Models
RFP and Proposal Considerations: Frequent Issues and Questions

- Should SMC RFPs require a development contractor to be "certified" or "SEI certified" to at least Level 3?
- Can a contractor's "certification" be examined?
- Can a contractor's SW-CMM Level be checked with the SEI?
- How does a SW-CMM Level relate to other software process information provided by the contractor?
- Should the government evaluate the contractor using the SW-CMM or other model?

RFP and Proposal Issues Addressed - 1

- "Certification" of a SW-CMM Level is impossible
  - No SW-CMM Level "certificate" or piece of paper exists
  - No organization exists that is authorized to issue a SW-CMM Level certification
    - SEI does not certify any development organization
  - Unlike ISO certification, no certificate is given to an organization
  - RFPs therefore cannot contain "certification" language
- Verifying a claimed SW-CMM Level is impossible
  - SEI maintains a database of organizations and their reported SW-CMM levels
  - SEI guarantees anonymity to organizations who report their SW-CMM levels
    - SEI uses this database to report on process improvement
  - SEI's database cannot be accessed
RFP and Proposal Issues Addressed - 2

- Process information provided in the proposal often conflicts with a claimed level
  - Offerors frequently claim Level 3 compliance, but have no Level 3 processes in place (e.g., peer reviews, organizational standards, training)
  - Processes expected at the offeror's proposed level exist at an organizational level, but are not being used on the proposed project
- Government evaluation of offeror's software processes is recommended
  - Two effective methods exist:
    - SW-CMM-based Software Capability Evaluation (SCE)*
    - Software Development Capability Evaluation (SDCE)
  - Both methods require specific language in the RFP and an experienced software engineering team to perform the evaluation

More information on evaluation methods soon

*SCE is a service mark of Carnegie-Mellon University

Capability Maturity Model for Software

- Description
- The Assessment Process
- RFP and Proposal Considerations
  - Integrated Capability Maturity Models
Capability Maturity Model Integration℠

- SEI, with industry participation, integrated several models
  - Evaluation against each CMM was resource-intensive
  - Similar components exist among all models
- Capability Maturity Models Integration (CMMI℠) combined these models into the CMMI-SE / SW / IPPD Version 1.02
  - CMM for Software (SW-CMM)
  - Electronic Industries Alliance Interim Standard (EIA/IS) 731, Systems Engineering Capability Model (SECM)
  - Integrated Product Development (IPD-CMM)
- Draft CMMI-SE / SW / IPPD / A Version 1.02d added this model
  - Software Acquisition (SA-CMM)
- CMMI Version 1.1 (estimated complete in December 2001) will replace SW-CMM
  - Tools, e.g., Standard CMM Assessment Method for Process Improvement (SCAMPI) and training are being integrated with the new model
  - Phase over from SW-CMM to CMMI over two year period

Outline

- Purpose of Contractor Capability Evaluations
- Evaluation Methods
  - Capability Maturity Model for Software
  - Software Capability Evaluation
  - Software Development Capability Evaluation
- Policy Change Effects on Contractor Evaluations
- Conclusions
- Resources
- Acronyms
- Backup Charts
  - Supplemental Capability Maturity Model Information
  - Comparison of the SDCE and SCE
SCE\textsuperscript{SM} History

- Developed to support source selection in major government software acquisitions
  - Publicly baselined in Software Capability Evaluation Version 1.5 Method Description (SEI, 1993)
  - Current version is documented in Software Capability Evaluation Version 3.0 Method Description (SEI, 1996)
  - Major activities of interviewing and document review remain relatively unchanged from the original

SCE Overview

- Uses the Capability Maturity Model for Software (SW-CMM) as a basis for findings
  - Five-level model for software process management and improvement
  - Ordered collection of software best practices grouped into Key Process Areas (KPAs)
  - Identifies goals, key practices, and activities for each KPA, at each of the five maturity levels
  - Uses ~100 question Maturity Questionnaire to focus evaluators on specific areas

- Scope
  - Software engineering processes
  - Software management processes
The SCE Process

Plan and Prepare for Evaluation
- Select and train the SCE team
- Define scope of evaluation
- Define processes
- Develop interview scripts
- Select projects for evaluation
- Analyze contractors' Maturity Questionnaires
- Incorporate into RFP

Conduct Evaluation
- Collect data on-site for 3 existing projects
  - Document review
  - Interviews
  - Presentations
- Consolidate data
- Develop findings against the SW-CMM
- Determine strengths, weaknesses, risks, and improvement activities

Report Evaluation Results
- Develop report for the SCE sponsor organization
- Present the results to the SCE sponsor
- Conduct feedback to the contractors (optional)

SW-CMM Tailoring for SCEs

SW-CMM
- Maturity Levels
- Key Process Areas
- Goals
- Common Features
- Key Practices

Depth-Oriented SCE Tailoring
- Select RELEVANT KPAs
- Evaluate ALL goals, common features, and key practices for selected KPAs
- Maturity level rating not possible

Breadth-Oriented SCE Tailoring
- Select ALL KPAs within chosen maturity levels
- Evaluate RELEVANT goals, common features, and key practices
- May determine a maturity level rating

Combined SCE Tailoring
- Select RELEVANT KPAs
- Evaluate RELEVANT goals, common features, and key practices
- Maturity level rating not possible
Outline

- Purpose of Contractor Capability Evaluations
- Evaluation Methods
  - Capability Maturity Model for Software
  - Software Capability Evaluation
  - Software Development Capability Evaluation
- Policy Change Effects on Contractor Evaluations
- Conclusions
- Resources
- Acronyms
- Backup Charts
  - Supplemental Capability Maturity Model Information
  - Comparison of the SDCE and SCE

SDCE History

- Developed by an AFMC Process Action Team (PAT) with industry and FFRDC participation (1993)
  - Developed for use in source selection
  - Documented in:
    - "Revised and Augmented SDCE Model", Aerospace TR 98(8550)-1, R. Haddad
- Based on two precursor methods
  - Software Development Capability/Capacity Review (SDC/CR) developed by Aeronautical Systems Center (1983)
  - Software Capability Evaluation (SCE), based on the SW-CMM developed by the Software Engineering Institute (1987)
SDCE Overview

- **Approach**
  - Questions and evaluation criteria are tailored to a program's needs
  - Questions and evaluation criteria are provided to contractors
  - Contractors respond in essay form with supporting evidence from past projects and the current project, as available
  - SDCE evaluation team reviews responses, identifying strengths, weaknesses, and risks
    - Written responses to evaluation notices (ENs) may be requested
  - Optional site visit may be performed
  - Results are integrated with the source selection process

- **Scope**
  - Software and systems engineering processes
  - Software and systems management processes
  - Special technologies

Hierarchical Structure of the SDCE Model
Basic SDCE Model – Top Level

SDCE Model

130 CMM Core Questions
757 Total Questions

Program Management

- Management Authority, Responsibility, and Accountability
- Program Planning and Tracking
- Subcontractor Management
- Legal and Contracting Issues
- Risk Control

- Paired numbers indicate the number of CMM Core / Total questions in each component
- Red-shadowed CCAs contain Core Questions

239

The SDCE Process

Plan and Prepare for Evaluation

- Determine program risks and resources
- Define processes
- Prepare plan & schedule
- Tailor SDCE (determine questions and criteria)
- Select & prepare team
- Incorporate into RFP

Conduct Evaluation

- Review proposals/offeror responses to questionnaire
- Prepare ENs
- Perform site visits (optional) to confirm/clarify responses
- Analyze EN responses
- Establish SDCE results (determine strengths, weaknesses, risks)
- Integrate with source selection

Report Evaluation Results

- Transition SDCE results
- Conduct feedback (optional)
- Program follow-through

THE AEROSPACE CORPORATION

240
Tailoring* the SDCE Model

Model Tailoring Objectives

- Strengthen SDCE usefulness to specific application
  - Focus on discriminating elements for source selection
  - Select and tailor questionnaire to focus on project risks
- Concentrate on key management and technical concerns
- Enhance SDCE applicability to specific program, e.g.,
  - Integrated Product Teams (IPTs) and intergroup coordination
  - Reuse/re-engineering
  - Requirements management
  - Management of incremental development
  - Commercial-Off-The-Shelf (COTS) use
- Support acquisition reform
  - Avoid detailed questions that lead to specific processes or methods
Outline

- Purpose of Contractor Capability Evaluations
- Evaluation Methods
  - Capability Maturity Model for Software
  - Software Capability Evaluation
  - Software Development Capability Evaluation
- Policy Change Effects on Contractor Evaluations
  - Conclusions
  - Resources
  - Acronyms
  - Backup Charts
    - Supplemental Capability Maturity Model Information
    - Comparison of the SDCE and SCE

Acquisition Changes Affecting SDCEs

- USD policy changes (26 October 1999)
  - Full SEI SW-CMM Level 3 compliance, or its equivalent, is required for all offerors on Acquisition Category I (ACAT I) contract source selections
- SMC guidance (10 December 1999)
  - SDCE has been, and will continue to be, used for ASC and SMC source selections
  - Supports the DUSD (S&T) effort to define SW-CMM Level 3 equivalence
  - Directs the use of the SDCE during source selection on all SMC ACAT I programs
DUSD (S&T) Equivalence IPT

- Memorandum from Dr. Etter (DUSD (S&T)) on 2 November 1999
  - Request Commands and FFRDCs to support national Integrated Process Team (IPT) in defining equivalence
- Formed national IPT on 19 January 2000 to determine SW-CMM Level 3 equivalence requirements
  - Define software equivalency to provide unambiguous yardsticks for other tools
  - Establish single evaluation method and evaluator qualifications in response to industry concerns
  - Include DoD organizations, FFRDCs and Industry
- Formed a “Mapping Subteam” from national IPT to determine SDCE model equivalence
  - Map SDCE to SW-CMM level 2 and level 3 goals and key practices
  - Use the mapping to determine congruence and gaps
  - Analyze the data to prepare recommendations for tool equivalency requirements

Aerospace Contributions on the IPT

- Aerospace assumed leadership of Mapping Subteam
  - Mapped six of the 13 KPAs to SDCE questions and criteria
  - Developed and presented “Generic Approach to Level 3 Equivalency” briefing to Dr. Etter (DUSD (S&T)) before her briefing at STC
  - Instrumental in reducing number of questions from over 200 down to 130
  - Finished mapping “Core” SDCE questions and criteria in June 2000
- Participated on the Equivalence Requirements Group
  - IPT subteam determined the requirements for evaluation method processes
  - Completed this task in June 2001
Model Mapping  
CMM Key Practices and SDCE Questions

SW-CMM  
Level 3  
117 Common Features  
112 Activities

SDCE  
466 Criteria  
719 Questions

SDCE Core Set  
118 Criteria  
130 Questions

Mapping & Gap Analysis

The Equivalent SDCE

- 22 May 2000 Dr. Etter issued a memorandum approving the SDCE as a CMM Level 3 equivalent evaluation tool contingent upon use of the “Core Set” of criteria and questions
- 130 questions and 118 related criteria form “Core” SDCE questionnaire
  - Core questions and criteria cannot be tailored to fit program needs - used “as is”
- Team must document findings for each criterion and its related question(s)
  - No longer able to evaluate the responses at a higher level
- A specific program may add additional criteria and questions to cover specific program risks
DoD 5000.2-R Acquisition Guidance*  
Updated for Policy Change

- Paragraph 5.2.3.5.6.1, Software Management, General requires
  - "Contractors performing software development or upgrade(s) for use in an ACAT I or ACAT IA program shall undergo an evaluation,
  - using either the tools developed by the Software Engineering Institute (SEI), or those approved by both the DoD Components and the Deputy Under Secretary of Defense (Science and Technology) (DUSD(S&T)).
  - DUSD(S&T) shall define Level 3 equivalence for approved evaluation tools.
  - At a minimum, full compliance with SEI Capability Maturity Model Level 3, or its equivalent in an approved evaluation tool, is the Department’s goal.
  - However, if the prospective contractor does not meet full compliance, risk mitigation planning shall describe, in detail, the schedule and actions that will be taken to remove deficiencies uncovered in the evaluation process.
  - Risk mitigation planning shall require PM approval.
  - The evaluation shall examine the business unit proposed to perform the work.
- The reuse of existing evaluation results performed within a two-year period prior to the date of the government solicitation is encouraged."

*DoD 5000.2-R, 10 June 2001, paragraph 5.2.3.5.6.1.5

Policy Impacts on the Government

LONGER TIME

LARGER BUDGET

MORE PERSONNEL
Policy Impacts on Source Selection

- Increased time, resources, personnel needed
- Not able to perform ACAT I SDCE within time constraints
- Additional cost is not justified by level of influence of SDCE in contract award
  - Resources expended are not in proportion to the position of the SDCE in Evaluation Criteria, Section M

Factors

- Past Performance
- Mission Capability
- Proposal Risk
- Price/Cost

Subfactors

- Subfactor 1
- Subfactor 2
- Subfactor 6

Software Process

Policy Impacts on Non-ACAT I Programs

- Non-ACAT I programs are using pre-policy SDCE planning and preparation processes
- Select questions and criteria based on program risks, schedule, resources
  - Tailor questions to meet program needs
  - Limit the number of questions and criteria, with no consequent resource and personnel problems

SDCEs have proven to be very effective for identifying risks and have been discriminators in source selections
Other Options

- One alternative is to perform the ACAT I SDCE during System Development and Demonstration phase prior to Milestone B source selection
  - Results would be incorporated and ranked with the source selection evaluation
  - SBIRS Low used this option
- Use results of Contractor internal assessments with Government team members
  - Government-Industry IPT proposed using these results to satisfy DoD policy
- Reuse of evaluation results is in the policy
  - Government currently does not release source selection results
  - Contractors who are evaluated at less than Level 3 may not want this information released
  - Contractor mergers also affect reuse of results
  - Plan is for SEI to keep future appraisal results with strict non-disclosure agreements to store or release results

When to Perform Which Appraisals in the Future

- For source selection
  - For ACAT I - #1 Reuse results only from contractor's same geographical organization's government-assisted CBA-IPIs and SCAMPs held within 2 prior years, #2 perform SDCE, or #3 perform SCE
  - For ACAT I or Non-ACAT I - Perform breadth-oriented capability evaluation based on program risks
- During the contract perform software risk assessments

- Perform software risk assessment(s)
- Perform a goal-level, breadth-oriented evaluation
  - Focus is on the project under bid
- Perform a SW-CMM Level 3 or equivalent evaluation
  - Potential reuse of evaluation results

Select evaluation(s) commensurate with program's software risk
Conclusions

- Software process improvement is essential for good predictions of performance and reductions in risk and rework
- The SW-CMM is one effective tool to improve processes
  - Organizations may claim a higher SW-CMM level than is demonstrated by your specific program
  - Be aware of new process initiatives such as CMMI
- Contractor Capability Evaluations are an important SMC activity
  - The SDCE is the primary evaluation method at SMC
  - The SCE is popular with industry and is the primary method used by other government organizations
- A high-quality evaluation is resource and time intensive, for both the government and the contractor, and should be used with discretion
### Overall SMC SDCE Metrics

<table>
<thead>
<tr>
<th>Program</th>
<th>Date</th>
<th># of Questions</th>
<th>Site Visit</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMSP STT</td>
<td>FY94</td>
<td>Unknown</td>
<td>Yes</td>
</tr>
<tr>
<td>ACMS</td>
<td>FY94</td>
<td>Unknown</td>
<td>Yes</td>
</tr>
<tr>
<td>DMSP CDFSII</td>
<td>FY94</td>
<td>100</td>
<td>Yes</td>
</tr>
<tr>
<td>GPS OCS</td>
<td>FY95</td>
<td>118</td>
<td>No</td>
</tr>
<tr>
<td>RSA Phase II</td>
<td>FY95</td>
<td>72</td>
<td>No</td>
</tr>
<tr>
<td>GPS Block IIF</td>
<td>FY95</td>
<td>27</td>
<td>No</td>
</tr>
<tr>
<td>AFSCN RCDC</td>
<td>FY95</td>
<td>32</td>
<td>No</td>
</tr>
<tr>
<td>AFSCN NOUC</td>
<td>FY95</td>
<td>79</td>
<td>No</td>
</tr>
<tr>
<td>AFSCN CCSC</td>
<td>FY95</td>
<td>48</td>
<td>No</td>
</tr>
<tr>
<td>SBIRS High</td>
<td>FY95</td>
<td>98</td>
<td>No</td>
</tr>
<tr>
<td>Classified</td>
<td>FY96</td>
<td>~50</td>
<td>No</td>
</tr>
<tr>
<td>AirBorne Laser</td>
<td>FY96</td>
<td>~550</td>
<td>No</td>
</tr>
<tr>
<td>EELV Pre-EMD</td>
<td>FY96</td>
<td>36</td>
<td>No</td>
</tr>
<tr>
<td>GBS</td>
<td>FY97</td>
<td>66</td>
<td>No</td>
</tr>
<tr>
<td>EELV EMD</td>
<td>FY98</td>
<td>68</td>
<td>No</td>
</tr>
<tr>
<td>NPOESS OMPS</td>
<td>FY99</td>
<td>26</td>
<td>No</td>
</tr>
<tr>
<td>NPOESS CrIS</td>
<td>FY99</td>
<td>26</td>
<td>No</td>
</tr>
<tr>
<td>AFSCN SLRS</td>
<td>FY00</td>
<td>22</td>
<td>No</td>
</tr>
<tr>
<td>EDS 2001</td>
<td>FY00</td>
<td>11</td>
<td>No</td>
</tr>
<tr>
<td>Wideband Gapfiller System</td>
<td>FY00</td>
<td>37</td>
<td>No</td>
</tr>
<tr>
<td>MILSATCOM (CCS-C)</td>
<td>FY00</td>
<td>36</td>
<td>No</td>
</tr>
<tr>
<td>SBIRS Low</td>
<td>FY01</td>
<td>158 + 52 FA6</td>
<td>No</td>
</tr>
<tr>
<td>SCNC</td>
<td>FY02</td>
<td>32</td>
<td>No</td>
</tr>
</tbody>
</table>

### Outline

- **Purpose of Contractor Capability Evaluations**
- **Evaluation Methods**
  - Capability Maturity Model for Software
  - Software Capability Evaluation
  - Software Development Capability Evaluation
- **Policy Change Effects on Contractor Evaluations**
- **Conclusions**

### Resources

- **Acronyms**
- **Backup Charts**
  - Supplemental Capability Maturity Model Information
  - Comparison of the SDCE and SCE
External Resources
Publications - 1

• Publications on CMM and SCE
  ❖ Dunaway, D., Masters, S., CMM-Based Appraisal for Internal Process Improvement (CBA IPI): Method Description, Software Engineering Institute, CMU/SEI-96-TR-007, Apr. 1996
  ❖ An Executive Introduction to CMM-Based Software Process Improvement, 1999

External Resources
Publications - 2

• Publications on SDCE
  ❖ Revised and Augmented SDCE Model, Aerospace TR 98(8550)-1, R. Haddad
  ❖ Guidelines for the Use of the SDCE, Aerospace TR 98(8550)-2, R. Haddad
Resources
On-Line

- SCE
  - Software Engineering Institute website
    → http://www.sei.cmu.edu/
    → Contains SEI documents on SCE, SW-CMM, and CMMI

- SDCE
  - AFMC/ASC website
    → Contains AFMC SDCE Pamphlet Volumes 1 and 2

- OSD Deskbook website
  → http://www.deskbook.osd.mil/
  → Contains DoD 5000.2-R, Mandatory Procedures for Major Defense Acquisition Programs (MDAPS) and Major Automated Information System (MAIS) Acquisition Programs

---

Internal Resources

- At Aerospace: Software Engineering Subdivision
  - Linda Abelson, Engineering Specialist, (310) 336-7350
  - Richard Adams, Sr. Engineering Specialist, (310) 336-2907
  - Sergio Alvarado, Director, (310) 336-2019
  - Suellen Eslinger, Distinguished Engineer, (310) 336-2906
  - Karen Owens, Engineering Specialist, (310) 336-5909
Outline

- Purpose of Contractor Capability Evaluations
- Evaluation Methods
  - Capability Maturity Model for Software
  - Software Capability Evaluation
  - Software Development Capability Evaluation
- Policy Change Effects on Contractor Evaluations
- Conclusions
- Resources
- Acronyms
- Backup Charts
  - Supplemental Capability Maturity Model Information
  - Comparison of the SDCE and SCE

Acronyms - 1

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Acquisition</td>
</tr>
<tr>
<td>ACAT I</td>
<td>Acquisition Category I (Critical and $$$)</td>
</tr>
<tr>
<td>AFMC</td>
<td>Air Force Materiel Command</td>
</tr>
<tr>
<td>AFMCPAM</td>
<td>AFMC Pamphlet</td>
</tr>
<tr>
<td>AT&amp;L</td>
<td>Acquisition, Technology and Logistics</td>
</tr>
<tr>
<td>CBA-IPI</td>
<td>CMM-Based Appraisal for Internal Process Improvement</td>
</tr>
<tr>
<td>CMM®</td>
<td>Capability Maturity Model®</td>
</tr>
<tr>
<td>CMMI®SM</td>
<td>Capability Maturity Model Integration®SM</td>
</tr>
<tr>
<td>CMU</td>
<td>Carnegie Mellon University</td>
</tr>
<tr>
<td>COTS</td>
<td>Commercial-Off-The-Shelf</td>
</tr>
<tr>
<td>DCMA</td>
<td>Defense Contract Management Agency</td>
</tr>
<tr>
<td>DoD</td>
<td>Department of Defense</td>
</tr>
<tr>
<td>EIA</td>
<td>Electronic Industries Association</td>
</tr>
<tr>
<td>EN</td>
<td>Evaluation Notice</td>
</tr>
<tr>
<td>Fed</td>
<td>Federal</td>
</tr>
<tr>
<td>FFRDC</td>
<td>Federally Funded Research and Development Center</td>
</tr>
</tbody>
</table>
Acronyms - 2

IDA       Institute for Defense Analyses
IEEE      Institute of Electrical and Electronics Engineers
IPD-CMM® Integrated Product Development Capability Maturity Model®
IPPD      Integrated Product and Process Development
IPT       Integrated Process Team
ISO       International Standards Organization
KPA       Key Process Area
MOIE      Mission-Oriented Investigation and Experimentation
N/A       Not applicable
OUSD      Office of the Secretary of Defense
OUSD (AT&L) Office of the Secretary of Defense for Acquisition, Technology and Logistics
PAT       Process Action Team
P-CMM®    People Capability Maturity Model®

Acronyms - 3

RFP       Request for Proposal
SA        Software Acquisition
SA-CMM®  Software Acquisition Capability Maturity Model®
SBIRS     for Space-Based Infrared System
SCE       Software Capability EvaluationSM
SDC/CR    Software Development Capability / Capacity Review
SDCE      Software Development Capability Evaluation
SE        Systems Engineering
SECM      Systems Engineering Capability Model
SE-CMM®  Systems Engineering Capability Maturity Model®
SEI       Software Engineering Institute
SMC       Space and Missiles Systems Center
SSM       Software Subcontractor Management
STC       Software Technology Conference
SW        Software
SW-CMM®  Capability Maturity Model for Software®
Acquisition Category I (ACAT I)

- ACAT I programs are Major Defense Acquisition Programs (MDAPs) or programs designated ACAT I by the Milestone Decision Authority (MDA). An MDAP is an acquisition program that is not a highly sensitive classified program (as determined by the Secretary of Defense) and that is:
  - (1) designated by the Under Secretary of Defense (Acquisition and Technology) (USD(A&T)) as an MDAP, or
  - (2) estimated by the USD(A&T) to require an eventual total expenditure for research, development, test and evaluation (RDT&E) of more than 355 million in fiscal year (FY) 1996 constant dollars or, for procurement, of more than 2.135 billion in FY 1996 constant dollars (10 USC2430T223349804).

- ACAT I programs have two sub-categories
  - ACAT ID, for which the MDA is USD(A&T). The “D” refers to the Defense Acquisition Board (DAB), which advises the USD(A&T) at major decision points.
  - ACAT IC, for which the MDA is the DoD Component Head or, if delegated, the DoD Component Acquisition Executive (CAE). The “C” refers to Component.

Reference: DoD 5000.2-R; Mandatory Procedures for MDAPs and MAIS Acquisition Programs; (Includes Change 4); 11 M

Outline

- Purpose of Contractor Capability Evaluations
- Evaluation Methods
  - Capability Maturity Model for Software
  - Software Capability Evaluation
  - Software Development Capability Evaluation
- Policy Change Effects on Contractor Evaluations
- Conclusions
- Resources
- Acronyms
  - Backup Charts
    - Supplemental Capability Maturity Model Information
    - Comparison of the SDCE and SCE
SW-CMM Level Focus

- Level 5 Optimizing - Implemented continuous and measurable software process improvement
- Level 4 Managed - Established quantitative characterization of software process and software products
- Level 3 Defined - Established organization infrastructure to institutionalize effective software engineering and management processes across all projects
- Level 2 Repeatable - Established basic project management controls (Project Management 101)
- Level 1 Initial - Depends on heroes

Key Process Areas

- Each maturity level of the model contains Key Process Areas (KPAs)
- A key process area (KPA) contains a set of goals and best practices that are performed collectively to achieve those goals
- There are 18 KPAs in the SW-CMM
- Each KPA has been defined to reside at a given maturity level
Repeatable Level 2 KPAs - Purpose*

- **Requirements Management**
  - Establish common understanding between customer and project as to customer's requirements.
- **Software Project Planning**
  - Establish reasonable plans for performing software engineering and managing the project.
- **Software Project Tracking and Oversight**
  - Provide adequate visibility into actual progress to enable taking effective action when performance deviates significantly.
- **Software Subcontract Management**
  - Select qualified software subcontractors and manage them effectively.
- **Software Quality Assurance**
  - Provide management visibility into process used by the software project and into products being built.
- **Software Configuration Management**
  - Establish and maintain integrity of products during entire life cycle.


Defined Level 3 KPAs - Purpose*

- **Organizational Process Focus**
  - Establish responsibility for software process activities that improve organization's overall software process capability.
- **Organizational Process Definition**
  - Develop and maintain usable software process assets to improve performance across projects, including the organization's standard process.
- **Training Program**
  - Develop skills and knowledge of individuals to perform roles effectively and efficiently.
- **Integrated Software Management**
  - Integrate software engineering and management activities into a coherent process tailored from the organization's standard process.
- **Software Product Engineering**
  - Consistently perform a well-defined engineering process that integrates engineering activities to produce correct, consistent products.
- **Intergroup Coordination**
  - Establish means for engineering groups to interact so project is better able to satisfy customer need.
- **Peer Reviews**
  - Detect defects so that they can be removed from work products early and efficiently.

**Managed Level 4 and 5 KPAs - Purpose**

- **Managed Level 4 KPAs - Purpose**
  - Quantitative Process Measurement
    - Quantitatively control performance of process used by the software project.
  - Software Quality Management
    - Develop a quantitative characterization of the project's software products and achieve specific quality goals.

- **Optimizing Level 5 KPAs - Purpose**
  - Defect Prevention
    - Identify the cause of defects and prevent them from recurring.
  - Technology Change Management
    - Identify new technologies (i.e., tools, methods, and processes) and transition them into the organization in an orderly manner.
  - Process Change Management
    - Continually improve the software processes used in the organization with the intent of improving software quality, increasing productivity, and decreasing the cycle time for product development.


---

**Immature Process Characteristics**

- Improvised by practitioners and management
- Not rigorously followed, results difficult to predict
- Highly dependent on current practitioners
- Provides low visibility into progress and quality
- May compromise product functionality and quality to meet schedule
- Higher risk in using new technology
Mature Process Characteristics

- Consistent with the way work actually gets done
- Defined, documented, and continuously improving
- Supported visibly by management and others
- Well controlled — process fidelity is audited and enforced
- Constructive use of product and process measurement
- Disciplined use of technology

Institutionalized Process Characteristics

- The organization builds an infrastructure that contains effective, usable, and consistently-applied processes
- The organizational culture conveys the process
- Management nurtures the culture and processes
- The reward system is aligned with the culture
- Institutionalized processes endure after the people who originally defined them have gone
Typical Level 1 (Initial) Characteristics

- **General characteristics**
  - The environment is ad-hoc
  - Success is dependent on individuals
  - Some processes may be done consistently
  - Few processes are written
  - Process is unpredictable and poorly controlled
  - Processes often work, but budget and schedule are exceeded

- **SEI data**
  - About 30% satisfy Requirements Management
  - Some satisfy Project Planning, Tracking & Oversight, and Software Configuration Management
  - Software Quality Assurance is least likely to be satisfied


---

Typical Level 2 (Repeatabale) Characteristics

- **Project-specific processes are in place**
  - Different projects may have different processes
  - Project management is not proactive
  - Basic project management to track cost, schedule, and functionality is done
  - Process is often reactive

- **SEI data**
  - About 25% of Level 2 organizations satisfy two Level 3 KPAs: Peer Reviews and Organizational Process Focus
  - Integrated Software Management, Organization Process Definition, and Training Program are the least frequently satisfied Level 3 KPAs

Typical Level 3 (Defined) Characteristics

- All Level 2 processes are functional
- Organizational and project processes are in place
  - Management and engineering organizational processes are in place and well documented
  - A process organization exists
  - Project-specific processes are developed by tailoring organizational processes
- Project management is proactive
- Training is provided for all software engineering activities
- Effective peer reviews are conducted

Typical Level 4 (Managed) and Level 5 (Optimizing) Characteristics

- Level 4 (Managed) Characteristics
  - All Level 3 processes are functional
  - Measurements (metrics) are used to manage the development
  - Processes and products are measured and controlled
- Level 5 (Optimizing) Characteristics
  - All Level 4 processes are functional
  - Processes are continuously improved using measures already in place
  - Measures are used to assess and control new processes and technology
Outline

- Purpose of Contractor Capability Evaluations
- Evaluation Methods
  - Capability Maturity Model for Software
  - Software Capability Evaluation
  - Software Development Capability Evaluation
- Policy Change Effects on Contractor Evaluations
- Conclusions
- Resources
- Acronyms
- Backup Charts
  - Supplemental Capability Maturity Model Information
  - Comparison of the SDCE and SCE

SDCE and SCE Similarities

- Gather information using a defined model
- Use evidence from recent projects to establish capability
- Produce results in terms of strengths, weaknesses and risks
- Have a defined process for integrating into source selection
## Differences
### Origin, Focus, and Use

<table>
<thead>
<tr>
<th></th>
<th>SDCE</th>
<th>SCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer</td>
<td>AFMC PAT, including Aerospace, SEI,</td>
<td>SEI with DoD sponsorship and government and industry review</td>
</tr>
<tr>
<td></td>
<td>and industry</td>
<td></td>
</tr>
<tr>
<td>Focus</td>
<td>Specific software-intensive project</td>
<td>Organizational software process capabilities</td>
</tr>
<tr>
<td>Intended use</td>
<td>Source selection</td>
<td>Source selection and contract monitoring</td>
</tr>
<tr>
<td>Primary users</td>
<td>SMC, NRO and ASC</td>
<td>Government, commercial, international</td>
</tr>
</tbody>
</table>

## Differences
### Preparation and Implementation

<table>
<thead>
<tr>
<th></th>
<th>SDCE</th>
<th>SCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluator training</td>
<td>Guidelines available</td>
<td>Week-long class required; more for lead evaluator</td>
</tr>
<tr>
<td>Questionnaire size</td>
<td>700+ questions; usually tailored to &lt;100</td>
<td>~ 100 question Maturity Questionnaire; some may be “N/A”</td>
</tr>
<tr>
<td>Questionnaire responses</td>
<td>Essay with supporting data</td>
<td>Yes / No; Comment required for “Yes”</td>
</tr>
<tr>
<td>Site visit</td>
<td>Optional; no defined process</td>
<td>Mandatory; well-defined process</td>
</tr>
<tr>
<td>Results established by</td>
<td>Questionnaire responses and optional site visits</td>
<td>Site visits; not from questionnaire responses</td>
</tr>
<tr>
<td>Assessment basis</td>
<td>Process existence, use, and quality</td>
<td>Process existence and use only*</td>
</tr>
</tbody>
</table>

*Note: If existence of detailed process characteristics is verified, then some level of process quality is assessed
## Differences
### Model Coverage

<table>
<thead>
<tr>
<th></th>
<th>SDCE</th>
<th>SCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specific technology areas</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Software/System Engineering Environment (S/SEE) related processes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Integration of systems and software engineering</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

## Differences
### Tailoring Approach

<table>
<thead>
<tr>
<th></th>
<th>SDCE</th>
<th>SCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guidelines for tailoring</td>
<td>Minimal set</td>
<td>Extensive</td>
</tr>
<tr>
<td>Tailoring risk</td>
<td>Not acknowledged</td>
<td>“Appraisal Risk” identified, documented and accepted</td>
</tr>
<tr>
<td>Tailoring decisions</td>
<td>No guidance given</td>
<td>Restricted to trained evaluators</td>
</tr>
<tr>
<td>Model tailoring</td>
<td>Extensive tailoring required</td>
<td>Depth-oriented and/or breadth-oriented</td>
</tr>
<tr>
<td>Process tailoring</td>
<td>Permitted</td>
<td>Constrain number of projects and length of site visits</td>
</tr>
</tbody>
</table>
Differences
Revision and Adaptation

<table>
<thead>
<tr>
<th>Evolution of model and process</th>
<th>SDCE</th>
<th>SCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimal, due to insufficient resources</td>
<td>Continual, with dedicated resources</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Application base</th>
<th>SDCE</th>
<th>SCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Limited - mainly Air Force</td>
<td>Widespread - government and industry</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Extension of use</th>
<th>SDCE</th>
<th>SCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Limited potential for contract monitoring</td>
<td>CMM based self-appraisal; contract monitoring</td>
<td></td>
</tr>
</tbody>
</table>

Abstract: Human Systems Integration
Acquisition Policies and Concerns

Human Systems Integration (HSI) comprises a collection of concerns related to the integration of humans into complex systems. HSI is a systems science that mediates the capabilities of the human operators with the technology and the operational environment to achieve an effective system. The principal concerns of HSI include: Human Factors Engineering (HFE), the systems engineering processes; Human Computer Interface (HCI), the software interfaces; Human Machine Interface (HMI), the hardware interfaces; and the related issues of operational procedures, staffing, personnel safety and training. HSI issues play a key part in the implementation of effective system operations and reduction of Operations and Maintenance (O&M) costs. Many of the problems faced by the satellite operations community have HSI issues at their core. This lecture reviews the current DoD acquisition policies for HSI and identifies areas of concern for SPOs developing systems with human-in-the-loop operations.
Human Systems Integration
Acquisition Policies and Concerns

Brian Shaw
Software Systems Acquisition Department
Software Engineering Subdivision

Contact: Brian.E.Shaw@aero.org

Acknowledgements

• Direct funding: HSI Process Development & Standardization
  ♦ Chief Engineers Office
    ♦ Ted Winer, Aerospace
    ♦ Col. Giblin, Chief Systems Engineering Division
  ♦ Space Mission Integration Office
    ♦ Dr. Marc Dinerstein, Aerospace

• Ongoing support
  ♦ Human Systems Integration (HSI) Section
    ♦ Suzanne Dawes
    ♦ Barbara Jex Courter
    ♦ Ping Pan, Intern (Carnegie Mellon)
    ♦ Vicky Hsu, Intern (Herndon Science Fair Winning Team)
  ♦ All of the SMC and NRO programs that we support
Outline

• Background
  ❖ Role of people in space systems
  ❖ Human Systems Integration (HSI)
• DOD HSI Guidance
  ❖ HSI process
  ❖ User interface hardware
  ❖ User interface software
• Current HSI Standardization Activities
• Examples of HSI in SMC Programs
• Warning Signs of Potential HSI Problems
• Summary and Conclusion
Role of People in Space Systems

- Three Segments of Space Systems
  - Space Segment
    - Satellite(s)
    - (Humans Not Included)
  - Ground Segment
    - Satellite Command & Control
    - Mission Data Processing
    - Computer Support
    - Communications Support
  - User Segment (examples)
    - Tactical Data Processors
    - GPS Receivers

Human Systems Integration

- Human-Systems Integration is the process of acquiring the:
  - ... operator interface to the system
  - ... operator performance driving the system/mission performance

- Human factors originated and developed by the military
  - Human Engineering processes
  - Human Engineering design criteria:
    - originally focused on hardware and fielded systems
    - later added software issues
  - Documentation largely exists only in mil-specs

- HCI now driven by commercial industry
  - Commercial specifications for interface design and usability exist
    - Well designed but not necessarily consistent
  - Rate of change often exceeds military procurement cycles
Human Systems Integration Components

- HSI: HUMAN-SYSTEMS INTEGRATION
- HE or HFE: HUMAN (FACTORS) ENGINEERING
  - Requirements & Process
- ConOps: OPERATIONS CONCEPT
- HMI: HUMAN-MACHINE INTERFACE
  - User Interface Hardware
- HCI: HUMAN-COMPUTER INTERFACE
  - User Interface Software
- RELATED ISSUES:
  - Personnel Safety
  - Documentation
  - Training

Human Systems Integration
Goals and Objectives

- Designing Human - System Interface for Compatibility with Human Capabilities
  - Easy to Learn
  - Easy to Use
  - Easy to Maintain
  - Safe
  - Error Resistant
  - Cost Effective

- Operations and Maintenance
  - Fewer personnel
  - Fewer human errors
  - Greater mission flexibility
  - Reduced training time and complexity

- Acquisition and Development
  - Reduced O&M costs
  - Reduced hardware/software development
Human Systems Integration
HSI in Operational Requirements Documents

- HMI specified in Satellite Operations ORD
  - 1993
    - 50% reduction in personnel
    - Level 5 military operators
    - Life-cycle cost savings goal
  - 1995
    - Removed the life-cycle cost savings goal

- These, alone, are incomplete requirements. Fail to
  - consider system interoperability needs
  - identify effective operational concept
  - lead contractor toward acceptable standardized solutions

Human Systems Integration
Examples of HSI Program Requirements (1 of 2)

<table>
<thead>
<tr>
<th>Requirement Topic</th>
<th>Space Lift Range</th>
<th>Air Force Satellite Control Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human Engineering Process</td>
<td>IMPACTS Program IAW DOD 5000.2 AFR 800-15 MIL-H-46885</td>
<td>DOD 5000.2 assumed</td>
</tr>
<tr>
<td>Operations Concept</td>
<td>AF Skill Level 5 Enlisted Tech oCiv Error Reduction Cognitive, Physical, Sensory, Skill Reduction, Physical/Accessibility, Complexity, Manpower, Training Reduction</td>
<td>AF Skill Level 3 One Operator Per Sat *</td>
</tr>
<tr>
<td>Human Machine Interface (User Interface Hardware)</td>
<td>ANS/HFS-100-1988</td>
<td></td>
</tr>
<tr>
<td>Human-Computer Interface (User Interface Software)</td>
<td>MIL-HDBK-761A</td>
<td>DoD HCI Style Guide ESRAIAA R-023A-1995</td>
</tr>
</tbody>
</table>
## Human Systems Integration
### Examples of HSI Program Requirements (2 of 2)

<table>
<thead>
<tr>
<th>Requirement Topic</th>
<th>Space Lift Range</th>
<th>Air Force Satellite Control Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>HSI Related Issues</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Personnel Safety</td>
<td>Meet Intent of EWR-127-1 and AFI 91-301 &amp; 302 (AFOSH)</td>
<td></td>
</tr>
<tr>
<td>- Documentation</td>
<td>Include Documentation To Support O&amp;M</td>
<td></td>
</tr>
<tr>
<td>- Training</td>
<td>Provide Processes, Procedures And Techniques For Training Systems Requirements Analysis (TSRA) As Part Of Requirements Analysis</td>
<td></td>
</tr>
</tbody>
</table>

---

## Human Systems Integration
### Ideal Characteristics

- **HSI must be part of the system development trade space**
  - Should be included in
    - Integrated Program Plan (IPP)
    - Mission Area Plans
      - Space Control
      - Space Support
      - Force Enhancement
    - Operational Requirements Document (ORD)
      - Key Performance Parameter
      - Part A - Validated Requirement
      - Part B - Implementation Details
      - Mission Needs Statement (MNS)
Human Systems Integration
Ideal Characteristics

- Specific goals must be identified
  - Operational and functional requirements
  - Verifiable exit criteria
    - System and HMI performance
    - Operator performance
  - Benchmark test scenarios
    - Especially for testing COTS for compliance and integration

- Detailed operational concept is needed
  - Architecture and modes of use
  - Staffing and operations

Human Systems Integration
Ideal Characteristics

- Organized Formalized Process
  - Explicit standardization goals and requirements
  - All stakeholders represented
  - Moderated by HMI experts
  - Disseminate lessons-learned outward and upward

- Task- and User-Based Requirements
  - System and “person-in-the-loop” task requirements
  - Operator performance requirements

- Iterative Development
- Rapid Prototyping
- User Evaluations
Outline

- Background
  - Role of people in space systems
  - Human Systems Integration (HSI)
- DOD HSI Guidance
  - HSI process
  - User interface hardware
  - User interface software
- Current HSI Standardization Activities
- Examples of HSI in SMC Programs
- Warning Signs of Potential HSI Problems
- Summary and Conclusion

Human (Factors) Engineering Guidance

HSI Hierarchy of Guidance

Mandatory DoD Acquisition Procedures
DOD 5000.2-R

Section 4.3.8 requires comprehensive and management strategies for HSI to assure human performance; reduce the burden that a design may impose on manpower, personnel, and training; and comply with safety and health issues.

HSI Process
HUMAN ENGINEERING

Software Acquisition
HUMAN COMPUTER INTERFACE

Hardware Acquisition
HUMAN MACHINE INTERFACE

Staffing, Safety and Training
Human (Factors) Engineering Guidance

- Current Human Engineering (Process) "Standards"
  - Establishment of Human Engineering Program
    - DoD 5000.2
    - AFR 800-15 (no longer in force)
  - Human Engineering Products and Processes
    - MIL-H-46855B and Data Item Descriptions (DIDs)
      - As guidance documents for processes and contractor-format products

Human (Factors) Engineering Guidance

- Evolving Human Engineering (Process) Standards
  - Critical Process Action Tool (CPAT): Human Engineering (2 Jan 97)
    - SMC/AX response to acquisition reform
    - Supports program officers for
      - RFP preparation
      - source selection
      - contract monitoring
Operations Concept Guidance

• Current Operational Concept Standards
  - Few Standards (content, & format) Available
    - AIAA Concept of Operations specification is the exception
    - Current mission area conops are "thin" or outdated about HMI
    - Some high-level guidance provided by government in ORDs
    - Few validated HMI functional and performance requirements

• Evolving Operational Concept Standards
  - Object-Oriented Programming (OOP) “Use Cases”
    - Used by software developers to describe how system is used
    - Similar to traditional task analytic tools
      - flow charts and operational sequence diagrams
    - NOT full system-level concept of operations, however!

Human Machine Interface Guidance

• Current HMI (Hardware) Standards
  - MIL-STD-1472E
    - Primarily for tactical/fielded non-office systems
  - ANSI/HFS-100-1988 (currently in revision)
    - Primarily for office-based computer hardware
  - AFSC-DH-1-3 (Rev. 2)
    - Includes aviation/cockpit issues
  - NASA-STD-3000
    - Includes manned space flight issues

• Evolving HMI (Hardware) Standards
  - ISO 9241 (17 Volumes)
    - Derived largely from Mil-Std 1472 with HCI sections added
Human Computer Interface Guidance

- Current HCI (Software) Standards
  - HCI Standardization Can Be Achieved At Various Levels
    - DoD currently requires standardization
    - Implemented under JTA and DII COE
  - Even With DoD-Mandated Standardization, Various HCI Designs Exist
    - Operational/procedural incompatibilities
    - Interchangability of operators may not be feasible

Human Machine Interface Guidance

- Levels of Standardization
  - GUI Look & Feel
    - X-Windows/Motif
    - - or - Windows NT
    - - or - Win32
    - CDE
  - GUI Style Guidance
    - DoD HCI Style Guide (TAFIM Volume 8)
  - Domain Area standard(s)
    - e.g., SMC Standard Practice Human Computer Interface Display Conventions for Space Systems Operations
  - Screen Design(s)
    - e.g., SMC/AFSPC Satellite Operations Screen Design Library

Diagram taken from the Joint Technical Architecture
Human Computer Interface Guidance

- Differences in Standardization Levels
- Objective/Requirement
  - Space System: Telemetry Display
  - Analogy: Paragraph of Prose
- Standards

<table>
<thead>
<tr>
<th>Standards</th>
<th>SPACE SYSTEM</th>
<th>ANALOGY</th>
</tr>
</thead>
<tbody>
<tr>
<td>GUI Look &amp; Feel</td>
<td>Graphical Functionality</td>
<td>Alphabet</td>
</tr>
<tr>
<td></td>
<td>Display Objects (widgets)</td>
<td>Numerals</td>
</tr>
<tr>
<td></td>
<td>Navigation Objects</td>
<td>Punctuation Symbols</td>
</tr>
<tr>
<td>GUI Style Guidance</td>
<td>Elemental Data/Information Structure and Format</td>
<td>Dictionary</td>
</tr>
<tr>
<td></td>
<td>- Date Formats</td>
<td>- Spelling</td>
</tr>
<tr>
<td></td>
<td>- Time Formats</td>
<td>- Definitions</td>
</tr>
<tr>
<td></td>
<td>- Numeric DI displays</td>
<td>- Pronunciation</td>
</tr>
<tr>
<td></td>
<td>- Labels</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Navigation Styles</td>
<td></td>
</tr>
<tr>
<td>Screen Design</td>
<td>Screen Organization/Layout</td>
<td>Syntax/Grammar</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Sentence Design</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Paragraph Structure</td>
</tr>
</tbody>
</table>

Human Computer Interface Guidance

INDUSTRY

DEPARTMENT OF DEFENSE

USAF

PROGRAM LEVEL (Examples)


Significant HMI Standards
Outline

- Background
  - Role of people in space systems
  - Human Systems Integration (HSI)
- DOD HSI Guidance
  - HSI process
  - User interface hardware
  - User interface software
- Current HSI Standardization Activities
- Examples of HSI in SMC Programs
- Warning Signs of Potential HSI Problems
- Summary and Conclusion

---

Current HSI standardization activities

- International
  - ISO: Human-centred design process

- Department of Defense
  - OSD: Acquisition procedures
  - SAF/AQ: Acquisition procedures
  - Air Force Inspection Agency (AFIA): Eagle Look - HSI in Air Force Acquisition
  - JTA & DII COE: Implementation, guidance, and style guide evolution

- USAF: Cross-Center
  - Space Mission Integration Office: Standardized HSI study
Current HSI standardization activities (continued)

- USAF: Satellite Control Domain
  - Human Machine Interface Review Board
  - SMC/TEO (Test and Evaluation Office) COTS testbed
  - HMI Conventions, Developer’s Style Guide, and templates

- Satellite Control: Satellite/Mission Family (selected examples)
  - NRO
    - Joint Site Implementation Standard
  - SMC
    - SBIRS style guide
    - GPS style guide

Examples of HSI in SMC programs

- Human Engineering: Process
  - “Human Machine Interface” Review Board

- Human Computer Interface: User Interface Software
  - Task design
  - Screen design

- Human Machine Interface: User Interface Hardware
  - Set-up/tear-down of fielded systems
  - Operator and maintainer safety

- Staffing, Safety, and Training
  - Staffing and loading analysis
Examples of HSI in SMC programs

- **Human Engineering: Process**
  - "Human Machine Interface" Review Board

- **Human Computer Interface: User Interface Software**
  - Task design
  - Screen design

- **Human Machine Interface: User Interface Hardware**
  - Set-up/tear-down of fielded systems
  - Operator and maintainer safety

- **Staffing, Safety, and Training**
  - Staffing and loading analysis

---

Human Engineering: Satellite Control HSI Community

An integrated effort toward common, cost-effective Human-Systems Integration
HSI Tasking

- Advocate position & priorities
- HSI operations concepts
- Incorporation into Integrated Program Plan
- Validated HSI operational requirements
- HMI RB & CERES support

---

HSI Tasking

- Support development of operations concepts
- Support development of operations requirements
- Support development of standards
- Review/evaluate COTS products
- Support fly-before-you buy
HSI Tasking

- Track COTS trends/operational concepts
- Provide potential COTS solutions
- Nominate COTS products for evaluation

Centers

- Support databases, working groups, HM RB
- Implement standards
- Formalize standardization documents
- Support mission area expansion of standards
- Implement contracting standards

Programs

- Support expansion of domain standards
- Coordinate with HM RB and CERES
- Support mission area expansion of standards
HSI Tasking

- Coordinate cross Center HSI efforts
- Oversee conventions and standards process
- Industry liaison for common conventions and standards

HMI RB

- Develop and update style guides
- Oversee HSI waiver process

Human Engineering:
HMI Review Board

---

Executive
Board

Implement

---

The Aerospace Corporation
Examples of HSI in SMC programs

- Human Engineering: Process
  - “Human Machine Interface” Review Board

- Human Computer Interface: User Interface Software
  - Task design
  - Screen design

- Human Machine Interface: User Interface Hardware
  - Set-up/tear-down of fielded systems
  - Operator and maintainer safety

- Staffing, Safety, and Training
  - Staffing and loading analysis

Human Computer Interface: Task Design

- Task Objective
  - Contact during support interval

- Controlled Display Variables
  - Input requirements for support (time, duration, etc.)
  - Equipment available for support
  - Equipment/components status
  - Equipment string status
  - Default equipment/string
  - Support files (Ephemeris)
  - Result of constraint check

- Operator Actions
  - Modify configuration (add/subtract components)
  - Select more detailed information about components
  - Select appropriate support files
  - Start a constraint check on configuration

- External Influences
  - Equipment failure
  - Schedule change (mission/resource)
**Human Computer Interface:**

**Screen Design**

---

**Examples of HSI in SMC programs**

- **Human Engineering: Process**
  - "Human Machine Interface" Review Board

- **Human Computer Interface: User Interface Software**
  - Task design
  - Screen design

- **Human Machine Interface: User Interface Hardware**
  - Set-up/tear-down of fielded systems
  - Operator and maintainer safety

- **Staffing, Safety, and Training**
  - Staffing and loading analysis
Examples of HSI in SMC programs

- Human Engineering: Process
  - "Human Machine Interface" Review Board

- Human Computer Interface: User Interface Software
  - Task design
  - Screen design

- Human Machine Interface: User Interface Hardware
  - Set-up/tear-down of fielded systems
  - Operator andmaintainer safety

- Staffing, Safety, and Training
  - Staffing and loading analysis
Staffing, Safety, and Training: Staffing Analysis

Two Operators

Three Operators

Staffing, Safety, and Training: Loading Studies

Weekly Free Time
2 SMCS/6 Sats Normal Ops

- SMCS 1  76 Supports
- SMCS 2  72 Supports

Free Time (min)
Outline

- Background
  - Role of people in space systems
  - Human Systems Integration (HSI)
- DOD HSI Guidance
  - HSI process
  - User interface hardware
  - User interface software
- Current HSI Standardization Activities
- Examples of HSI in SMC Programs
- Warning Signs of Potential HSI Problems
- Summary and Conclusion

Warning Signs of Potential HSI Problems

- No HSI or lack of qualified HSI oversight in the design process

- Incomplete Requirements
  - Human performance criteria are rarely specified

- Standards inappropriate or not properly specified
  - Out of date standards or not specified at all

- Training
  - No Instructional Systems Development (ISD) process used
  - Training development deferred
Outline

- Background
  - Role of people in space systems
  - Human Systems Integration (HSI)
- DOD HSI Guidance
  - HSI process
  - User interface hardware
  - User interface software
- Current HSI Standardization Activities
- Examples of HSI in SMC Programs
- Warning Signs of Potential HSI Problems
- Summary and Conclusion

Summary

- People are essential and costly components of space systems
- HSI is the system science that seeks to optimize human performance in complex systems
- HSI comprises many issues: acquisition/development process, hardware, software, operational procedures, personnel safety, and training
- HSI is REQUIRED in space system development and operations and is most effective if implemented early in development
- Tools and expertise to support HSI efforts are readily available
Conclusion

- Designing Human - System Interface for Compatibility with Human Capabilities
  - Easy to Learn
  - Easy to Use
  - Easy to Maintain
  - Safe
  - Error Resistant
  - Cost Effective

- Operations and Maintenance
  - Fewer personnel
  - Fewer human errors
  - Greater mission flexibility
  - Reduced training time and complexity

- Acquisition and Development
  - Reduced O&M costs
  - Reduced hardware/software development

Failure to properly consider HSI in system acquisition will reduce the chances of achieving these goals!

Backup Charts

- Resources
  - Aerospace
  - United States Air Force (USAF)
  - Department of Defense (DOD)
  - Professional Societies

- Acronym List
Resources

- **Aerospace**
  - Computer Systems Division
    Rami Razouk, General Manager (x-66644)

  - Software Engineering Subdivision
    Mary A. Rich, Principal Director (x-65313)

  - Software Systems Acquisition Department
    Sergio Alvarado, Director (x-62019)

  - Human Systems Integration Section
    Brian Shaw, Manager (x-65134)

Resources

- United States Air Force (USAF)
  - SMC/AXE (Chief Engineers Office) provides cross-program guidance for HSI standardization. Colonel Michael Giblin co-chairs, with AFSPC, the Human Machine Interface Review Board.

  - The Human Systems Integration Office at the 311th Human HSW (Brooks AFB) is the coordinating agent for HSI in risk management, acquisition planning, and requirements generation development guidance for Air Force programs. The POC is Dr. Suzanne Lipscomb (DSN 240-4452; HSIO@brooks.af.mil)

  - Air Force acquisition policy documents can be obtained from the Secretary of the Air Force for Acquisition on their world-wide web site at http://www.safaq.hq.af.mil/.
Resources

- Department of Defense (DOD)
  - DoD Directives are available at the Washington Headquarters Services Directives And Records Branch (Directives Section) word wide web site at http://web7.whs.osd.mil/corres.htm

  - The Assistant Secretary of Defense’s (Command, Control, Communications, and Intelligence) Information Technology Directorate (ITD) is the focal point for information systems’ interface, interoperability, and technology standards; technology insertion; technical integration requirements; data management policy, procedures and standards; software reuse policy and procedures; information technology plan; open systems hardware and software standards. Human computer interaction (HCI) is an area of major concern spanning many of the areas assigned to ITD. The DoD HCI Implementation Plan can be obtained from the Office of the Secretary of Defense for Command, Control, Communications, and Intelligence word wide web site at http://www.c3i.osd.mil/bpr/dodim/hciplan.html.

341

Resources

- DoD Resources (continued)
  - The Joint Technical Architecture documents and DOD HCI Style Guide (TAFIM Volume 8, in TAFIM archive) are available at the DISA world wide web site at http://www-jta.itsi.disa.mil/.

  - Defense Information Infrastructure Common Operating Environment documents are available at the DII COE world wide web site at http://diicoe.disa.mil/coe/.
Resources

- Professional Societies
  - Human Factors and Ergonomics Society (HFES) is the international society for human factors and ergonomics professionals. They issue three monthly publications: a journal reporting academic advances in the field; a monthly magazine that discusses applications of human factors in day-to-day life; and a monthly bulletin that reports society-related issues to the membership. There are many regional chapters of HFES including one in Los Angeles. The Los Angeles Chapter of HFES holds monthly technical meetings addressing the full breadth of local members interests. Additionally, HFES holds an annual one-week convention. Information about HFES can be obtained on the world-wide web at http://www.hfes.org.
  - Other professional societies with human systems integration interests or components include: Association of Computing Machinery (ACM), Special Interest Group, Computer Human Interface (SIGCHI), and the International Committee on Systems Engineering (INCOSE).

Acronym List (A - D)

- AFIA  Air Force Inspection Agency
- AFMC  Air Force Materiel Command
- AFPSC  Air Force Space Command
- AIAA  American Institute of Aviation and Aeronautics
- ANSI  American National Standards Institute
- AWE  Alarm/Warning/Event
- BATT  Battery
- CERES  Center for Research (operated by SMC/TEO)
- ConOps  Concept of Operations
- COTS  Commercial Off The Shelf Software
- DID  Data Item Description
- DII COE  Defense Information Infrastructure Common Operating Environment
- DoD  Department of Defense
### Acronym List (E - J)

- **ESC** Electronic Systems Command
- **GPS** Global Positioning System
- **GUI** Graphical user Interface
- **HCI** Human Computer Interface
- **HE** Human Engineering
- **HFE** Human Factors Engineering
- **HFES** Human Factors and Ergonomics Society (current name)
- **HFS** Human Factors Society (former name)
- **HMI** Human Machine Interface
- **HM RB** Human Machine Interface Review Board
- **HSI** Human Systems Integration
- **ISO** International Standards Organization
- **JTA** Joint Technical Architecture

### Acronym List (M - Smc)

- **MIL-STD** Military Standard
- **MMI** Man Machine Interface
- **MNVR** Maneuver
- **NRO** National Reconnaissance Office
- **O&M** Operations and Maintenance
- **Ops** Operations
- **SAF/AQ** Secretary of the Air Force for Acquisition
- **Sat** Satellite
- **SBIRS** Space-Based Infrared System
- **SMC** Space and Missile Systems Center (AFMC)
- **SMC/TEO** Space and Missile Systems Center Test and Evaluation Office
- **SMCS** Satellite Mission Control System (MILSTAR operational terminal)
Acronym List (SmI - U)

- SMIO  Space Mission Integration Office
- SOH   State of Health
- Spec  Specification
- TDRSS Tactical Data Relay SS
- USAF  United States Air Force