school of
SYSTEMS
and
LOGISTICS
▶ education ▶ service ▶ research

Engineering Quality Software

10 Recommendations for Improved Software Quality Management

22nd SSTC
Salt Lake City, UT
27 Apr 2010

Lt Col Marcus W. Hervey, USAF
AFIT/LSS
marcus.hervey@us.af.mil

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **27 APR 2010**	2. REPORT TYPE	3. DATES COVERED **00-00-2010 to 00-00-2010**

4. TITLE AND SUBTITLE **Engineering Quality Software: 10 Recommendations for Improved Software Quality Management**	5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Air Force Institute of Technology,AFIT/LSS,Wright Patterson AFB,OH,45433**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR’S ACRONYM(S) 11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
Presented at the 22nd Systems and Software Technology Conference (SSTC), 26-29 April 2010, Salt Lake City, UT.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**32**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

Disclaimer

"The views expressed in this presentation are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government."

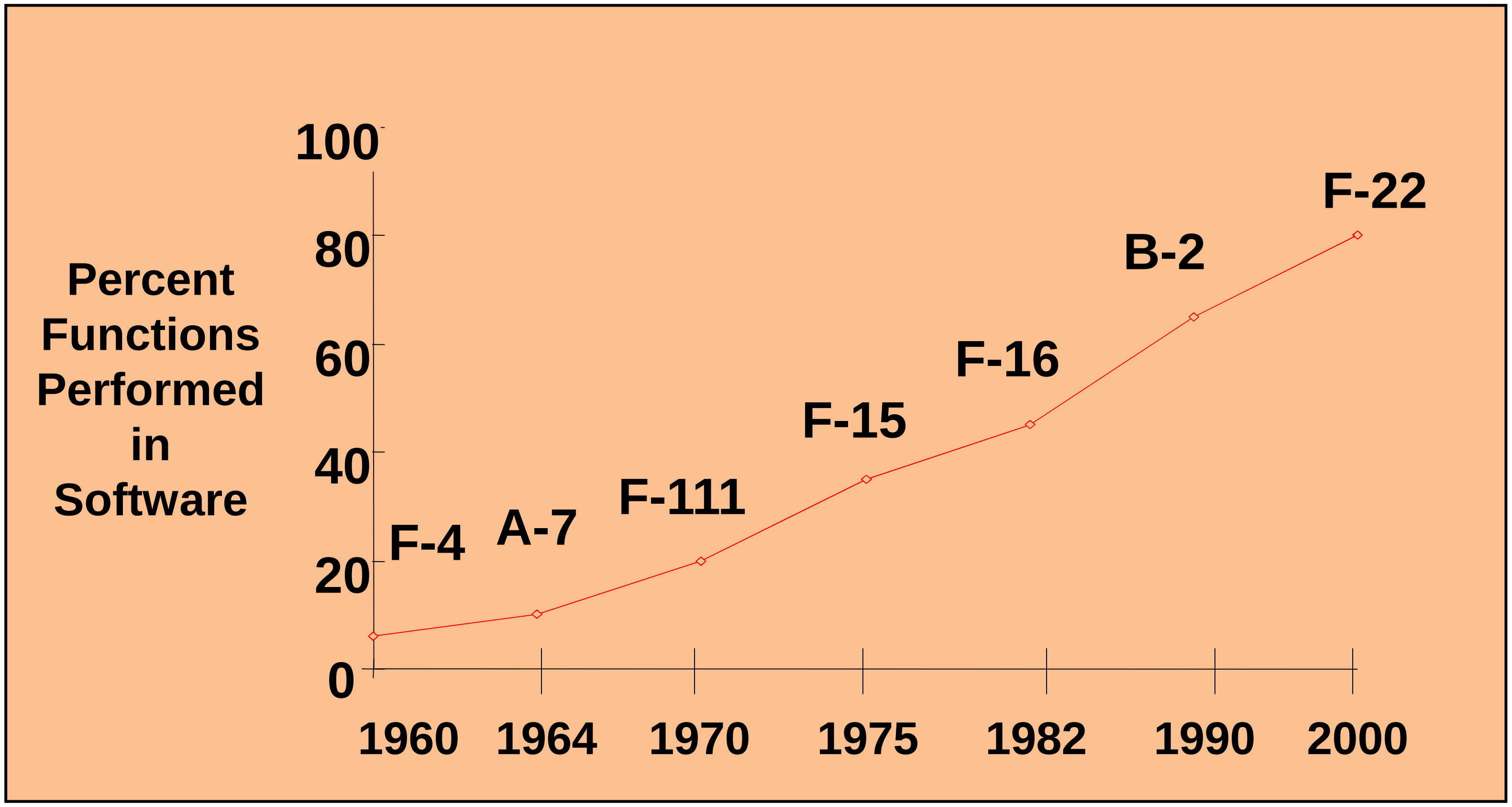
Outline

- Software Trends & Motivation
- What is Software Quality?
- Why is Software Quality Important?
- Software Quality Framework
- Ten Focus Recommendations
- Summary

Software Trends

- More complex systems
  - More functionality
  - More diverse, larger teams
- Heterogeneous architectures
- Parallel programming
  - Assure correctness and performance

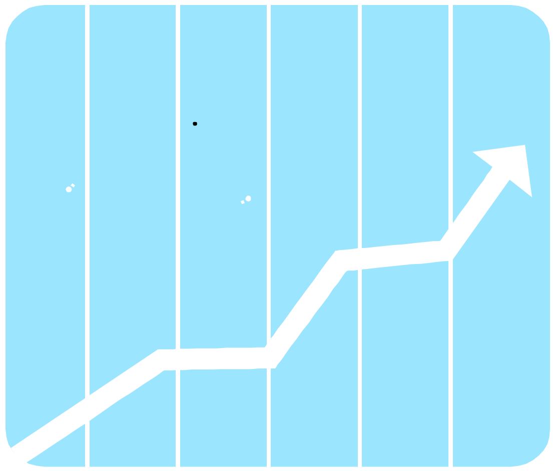
Weapon System Software Dependence



Ref: Crouching Dragon, Hidden Software

Increasing Code Size

Manufacturer	System	Code Size
Lockheed Martin/Boeing	F-22 Raptor	~1.7M LOC
Lockheed Martin	F-35 Joint Strike Fighter	~5.7M LOC
Boeing	787 Dreamliner	~ 6.5M LOC



Ref: This Car runs on code

DoD Software Challenges - 1994

- **Lack of Consistent Attention to Software Process**
- **Poor Requirements Definition – lack of user involvement**
- Inadequate Software Process Management & Control By Contractors
  - No “Team” of Vendors and users; little SME participation
- Ineffective Subcontractor Management
- **Software Architectures Ignored**
- **Poorly Defined and Controlled Interfaces (HW, Comm, Software)**
- Assumption That Software Upgrades Can “Fix” Hardware Deficiencies
- **Focus on Innovation Rather than Cost and Risk**
- Limited or No Tailoring of Military Specifications Based on Continuing Cost-Benefit Evaluations

Ref: Report of the DSB Task Force on Acquiring Defense Software Commercially

NDIA Top SWE Issues - 2006

- **The impact of system requirements upon software is not consistently quantified and managed in development or sustainment.**
- Fundamental system engineering decisions are made without full participation of software engineering.
- Software life-cycle planning and management by acquirers and suppliers is ineffective.
- **The quantity and quality of software engineering expertise is insufficient to meet the demands of government and the defense industry.**
- **Traditional software verification techniques are costly and ineffective for dealing with the scale and complexity of modern systems.**
- There is a failure to assure correct, predictable, safe, secure execution of complex software in distributed environments.
- Inadequate attention is given to total lifecycle issues for COTS/NDI impacts on lifecycle cost and risk.

Ref: NDIA Top 7 SWE Issues Report

Standish Group Report

	Year 1994	Year 1996	Year 1998	Year 2000	Year 2002	Year 2004	Year 2006	Year 2009
Successful	16%	27%	26%	28%	34%	29%	35%	32%
Challenged	31%	40%	28%	23%	15%	53%	19%	44%
Failed	53%	33%	46%	49%	51%	18%	46%	24%
Challenged+ Failed	84%	73%	74%	72%	66%	71%	67%	68%

↑

Quality Improvement Opportunities

Ref: The Rise and Fall of Chaos Report Figures

What is Software Quality?

IEEE defines as …

(1) The degree to which a system, component, or process meets specified requirements;

(2) The degree to which a system, component, or process meets customer or user needs or expectations.

Ref: IEEE Std 610.12-1990

Quality Perspectives

- **Process Quality (CMMI)**
- **Product Quality (ISO/IEC 2500x)**
  - Internal Quality Attributes
  - External Quality Attributes
  - Quality in Use (Customer's View)

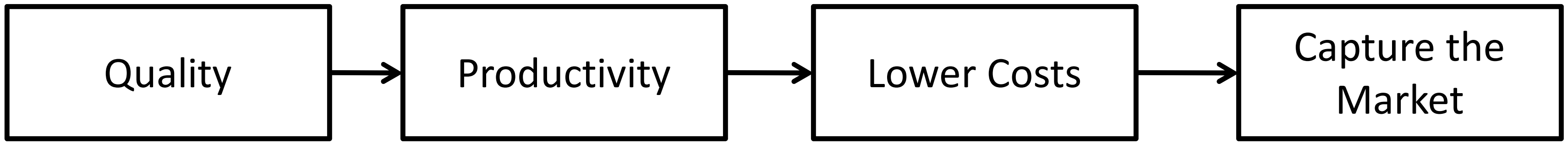
$$\begin{tikzcd}\boxed{\text{Process}} \arrow[r] & \boxed{\text{Internal Quality}} \arrow[r] & \boxed{\text{External Quality}} \arrow[r] & \boxed{\text{Quality in Use}}\end{tikzcd}$$

Why is Software Quality Important?

- Military
  - Affects ability to deliver and sustain superior capability
  - Quality focus needed for to improve stewardship and productivity
- Industry
  - Affects competitive advantage, reputation and market share

Quality can Make or Break You

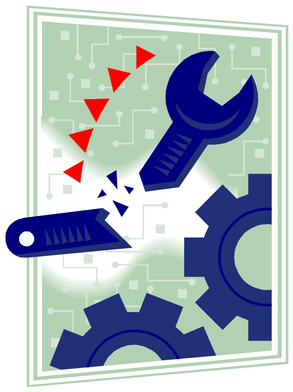
Deming's Quality Chain Reaction



Ref: Out of the Crisis

Quality Problems at Toyota

- Reputation for producing high-quality vehicles
  - Toyota Production System based on "The Toyota Way"
  - 4-P Model: Problem Solving, People/Partners, Process, Philosophy
- Software quality problems
  - Hybrid Anti-lock braking software: 2010
    - Toyota Sai, MY 2010 Toyota Prius, MY 2010 Lexus HS 250h
  - Sudden stall and shut down – recalled 160,000 cars : 2005
    - Recalled 160,000 of 2004 /2005 Prius hybrids

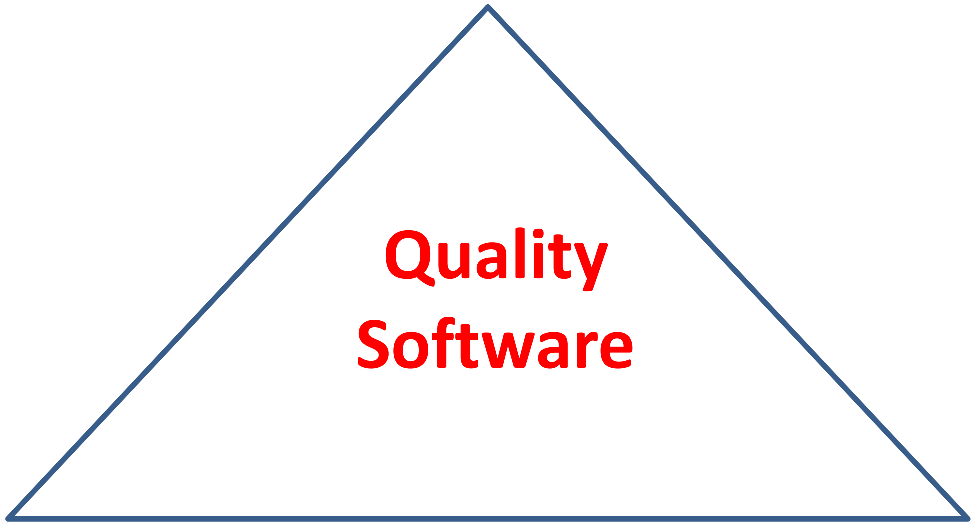


Ref: This Car Runs on Code

The Quest for Software Quality

Process

Tailored, Defined, Measurable & Repeatable



Technology

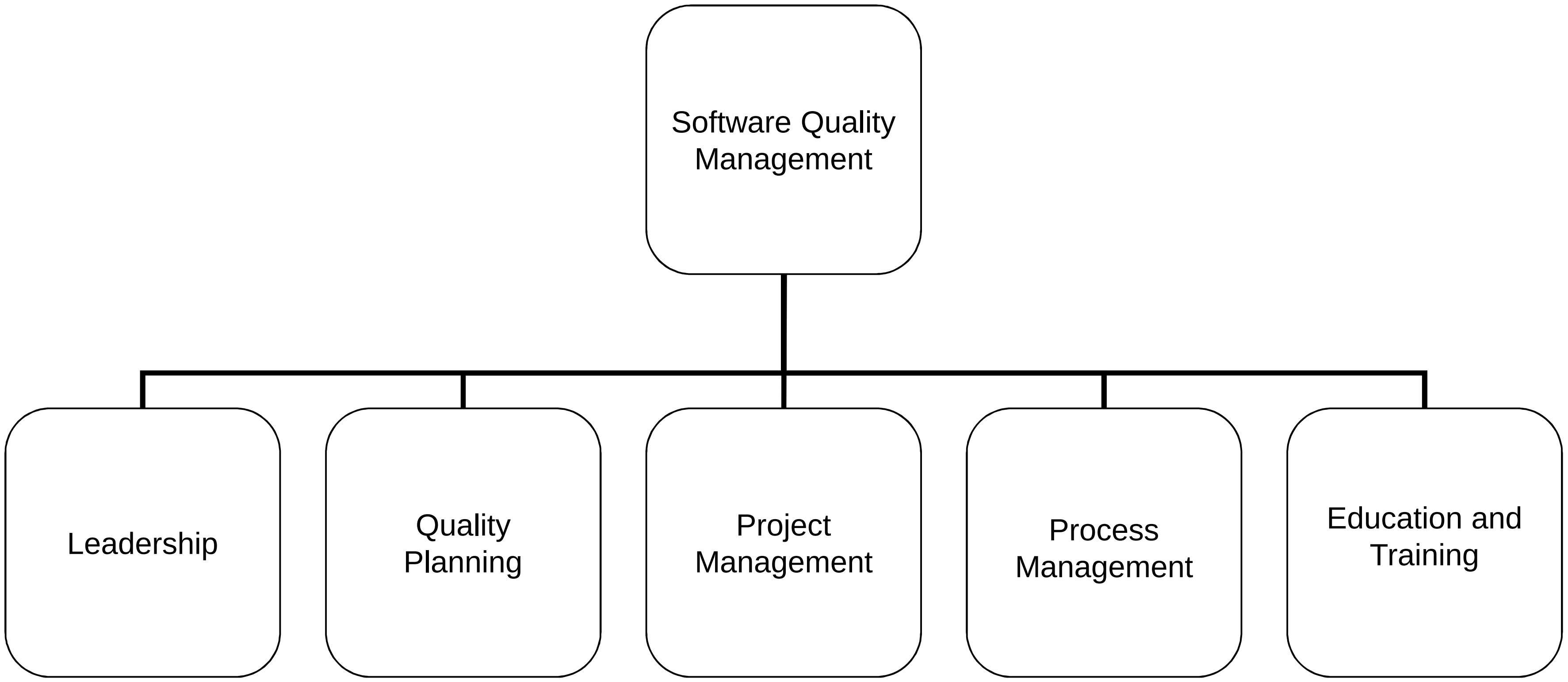
Effective Technology Insertion

People

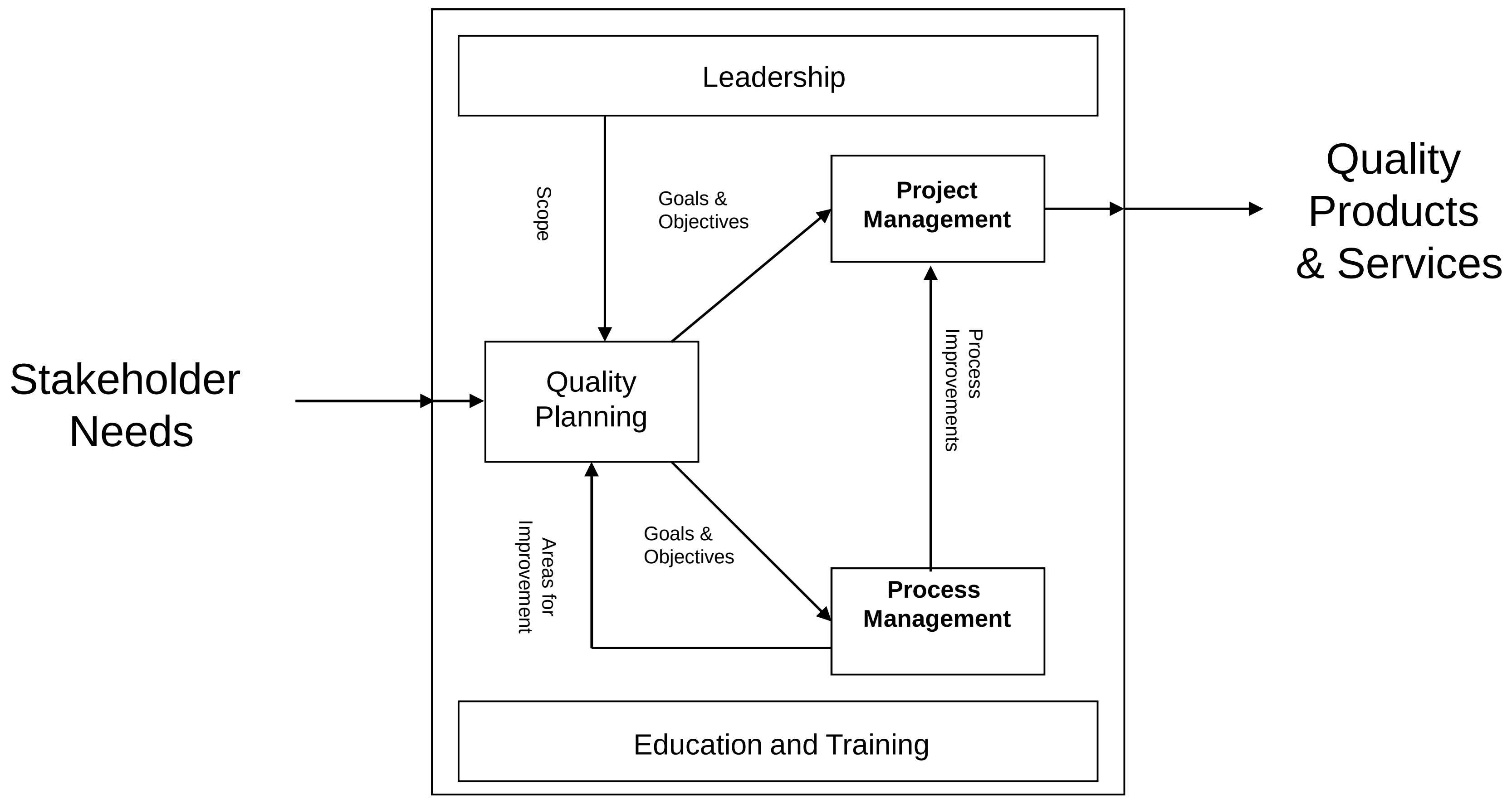
Technical and Process Training,
Process Discipline

Result : Predictable Cost, Schedule and Performance

Software Quality Components



Software Quality Framework



Ten Focus Recommendations

1. Focus on a common software quality definition
2. Focus on software quality planning
3. Focus on developing “quality” people
4. Focus on quality assessments
5. Focus on requirements
6. Focus on creating an effective SQA group
7. Focus on risk mitigation
8. Focus on defect prevention
9. Focus on software quality metrics
10. Focus on teamwork

#1 – Common Quality Definition

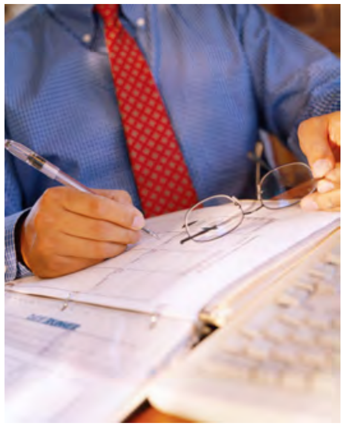
- Issue:
  - Software quality means different things to different people
  - Resolve competing priorities

- Recommendation:
  - Achieve consensus on quality definition
  - Create organizational software quality policy

Reach for the same quality goal

#2 – Software Quality Planning

- Issue:
  - Lack of appreciation of planning for quality initiatives
- Recommendation:
  - V&V focuses on the quality of products
    - IEEE Std 1059
  - QA focuses on the quality of processes
    - IEEE Std 730



Quality does not just happen, it has to be planned

#3 – Developing "Quality" People

- Issue:
  - Software is highly prone to human errors
  - Lack of "quality" development skills
- Recommendation:
  - Enable professionals to hone their craft
  - Encourage professional certifications
    - PMI PMP, IEEE CSDP, INCOSE CSEP, ASQ CSQE
  - Advance the discipline and practice

Create a culture of software professional excellence

#4 – Quality Assessments

- Issue:
  - Process and Product problems go unnoticed
- Recommendation:
  - CMMI/ISO 9000 Assessments
  - Capture organizational knowledge
    - Identify best practices, lessons learned

Know where you are, and where you need to be

#5 – Requirements

- Issue:
  - Unrealistic expectations – undefined scope
  - Poor requirements engineering

- Recommendation:
  - Effective communication is the key
  - Requirements management plan

Know your stakeholders

#6 – Effective SQA group

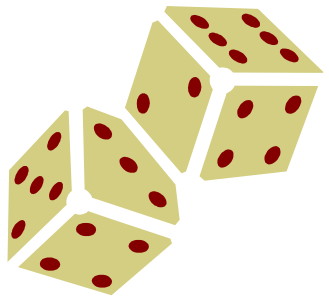
- Issue:
  - Lack of understanding of status of quality initiatives

- Recommendation:
  - Empower and embrace QA activities
  - Learn to effectively use walkthroughs, inspections, audits and reviews

QA is your friend

#7 – Risk Mitigation

- Issue:
  - Problem areas not identified and acted on early enough
  - Don't prepare for contingencies



- Recommendation:
  - Ask "what if this happens"
  - Prioritize based on project objectives

Anticipate problems and develop ready solutions

#8 – Defect Prevention

- Issue:
  - Quality defined as detection of defects
  - Reactive focused – identify, correct
- Recommendation:
  - Adopt a proactive approach to quality
    - Prevention works better than detection
    - It's easier to do it right the first time
  - Start earlier, look upstream for improvements

It's easier to do it right the first time

#9 –Software Quality Metrics

- Issue:
  - Limited indicators for process and product status
- Recommendation:
  - Tailored product and process measures should be used
    - Process – # of reviews, audits, inspections
    - Product – internal, external, quality in use
    - Project – earned value

That which gets measured, gets managed

#10 – Teamwork

- Issue:
  - Software is involved in increasingly diverse functions

- Recommendation:
  - Precisely define roles and responsibilities
  - Create ”sweet” spot
    - Successfully integrate professional functional bodies of knowledge

It takes a “village” to deliver quality software

Summary

- Systems will continue to increase in complexity and software dependence
  - Increasing software functionality; larger, more diverse teams
- Quality must remain in the forefront
  - Primary factor in Superior Capability & Competitive Advantage
- Quality is a leadership choice
  - Everyone's job, but leader's responsibility
- Lifecycle Approach to Quality Management
  - Focus on prevention rather than detection
- Quality management systems must evolve
  - Even the best quality management systems can have challenges

Focus on QUALITY!

References

- Charette, Robert N. “This Car Runs on Code” IEEE Spectrum, February 2009.
- Deming, W. Edward. Out of the Crisis. The MIT Press, 2000.
- Eveleens, J. Laurenz and Chris Verhoef. “The Rise and Fall of the Chaos Report Figures,” IEEE Software, vol. 25 no. 1, Jan/Feb 2010, pp. 30-36
- Ferguson, J. “Crouching Dragon, Hidden Software: Software in DoD Weapon Systems,“ IEEE Software, vol. 18, no. 4, Jul/Aug 2001, pp. 105-107.
- IEEE Standard 610.12-1990 IEEE Standard Glossary of Software Engineering Terminology, IEEE, 1990.
- National Defense Industrial Association. Top Software Engineering Issues in the Defense Industry, 2006.
- Office of the Under Secretary of Defense for Acquisition & Technology. Report of the Defense Science Board Task Force on Acquiring Defense Software Commercially, June 1994.

For More Information

Lt Col Marcus W Hervey, USAF

AFIT/LSS

marcus.hervey@us.af.mil

937-255-7777 x3248

Acronym List

- ASQ – American Society for Quality
- CSDP – Certified Software Development Professional
- CSEP – Certified Systems Engineering Professional
- CSQE – Certified Software Quality Engineer
- DSB – Defense Science Board
- IEEE – Institute of Electrical and Electronics Engineers
- IEC – International Electrotechnical Commission
- ISO – International Organization for Standardization
- MY – Model Year
- NDIA – National Defense Industrial Association
- SWE – Software Engineering
- PMI – Project Management Institute
- PMP – Project Management Professional