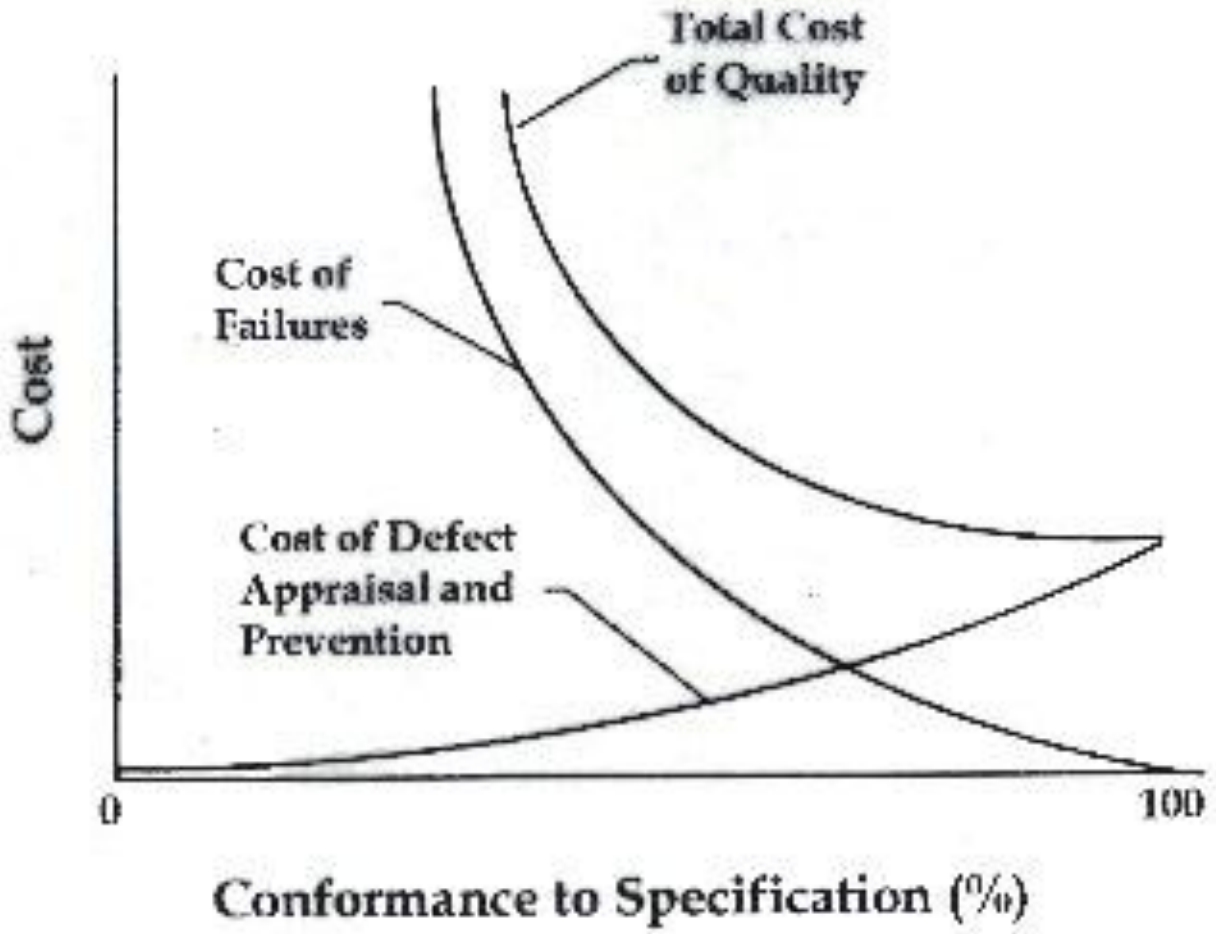
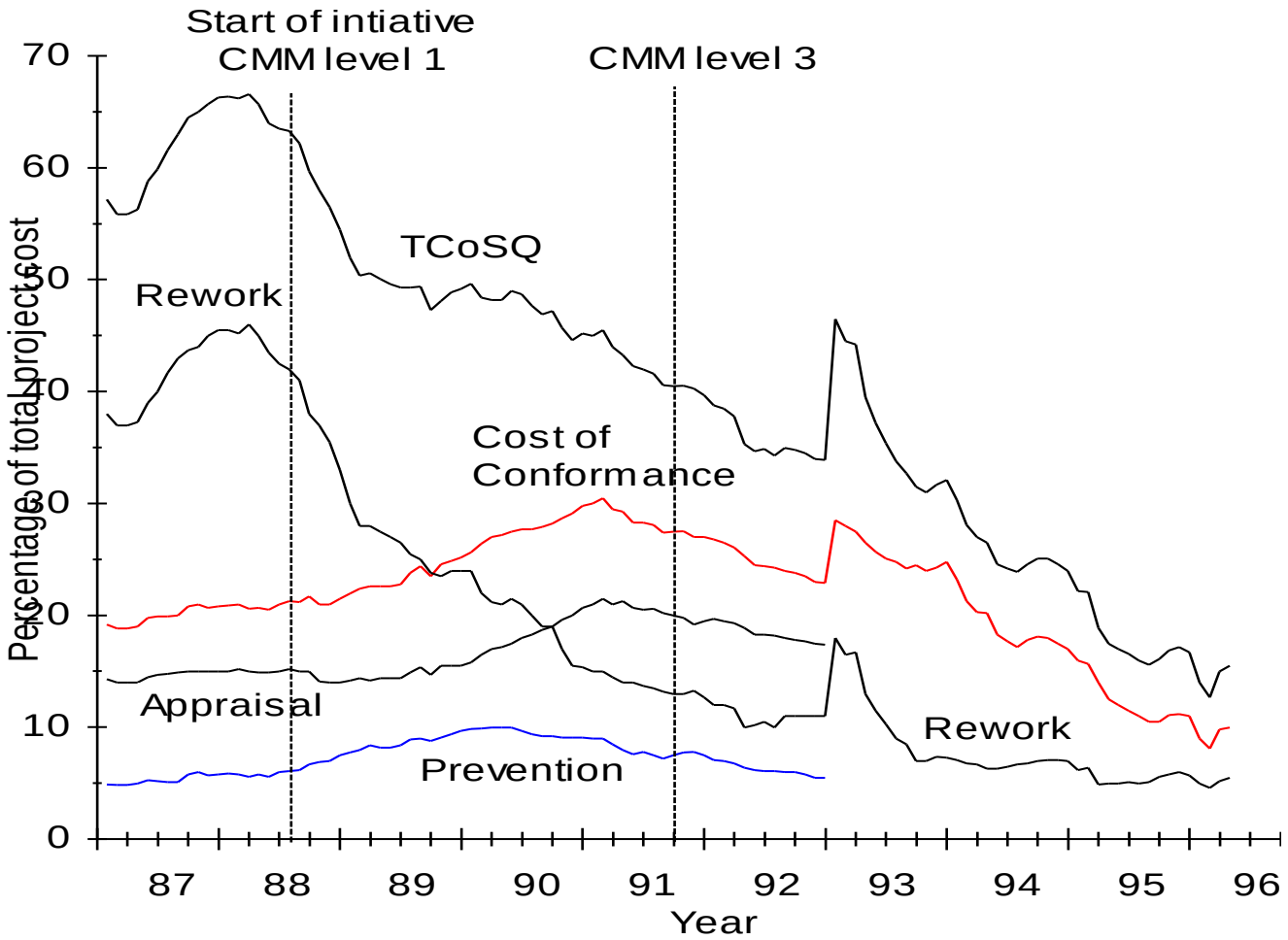
Revised Cost of Quality Model



Juran's Quality Control Handbook, 4th Edition (1988)



Software process improvement at Raytheon. T.J.Haley. *IEEE SOFTWARE (November 1996).*

Investments in Prevention

- Rules: Security Policy, Procedures
- Tools: Passwords, Firewalls, Encryption
- Awareness, Training, Education

Investments in Response

- Contingency Plans
- Backup Capabilities
- Emergency Drills

Verification

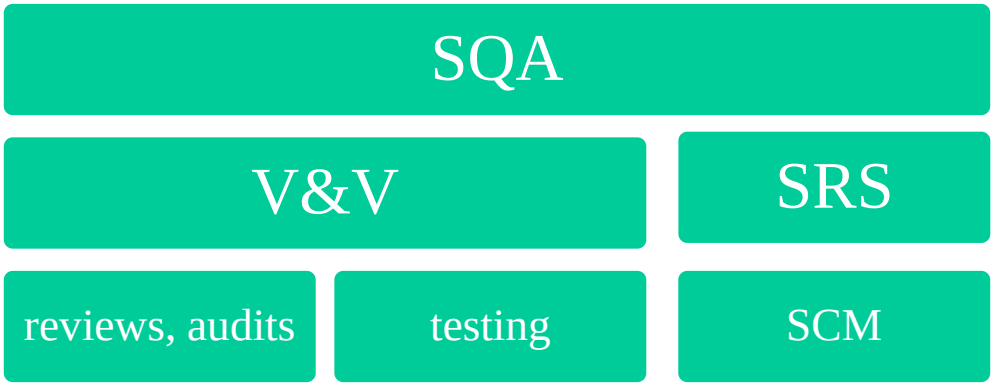
"Are we doing the job right?"

Validation

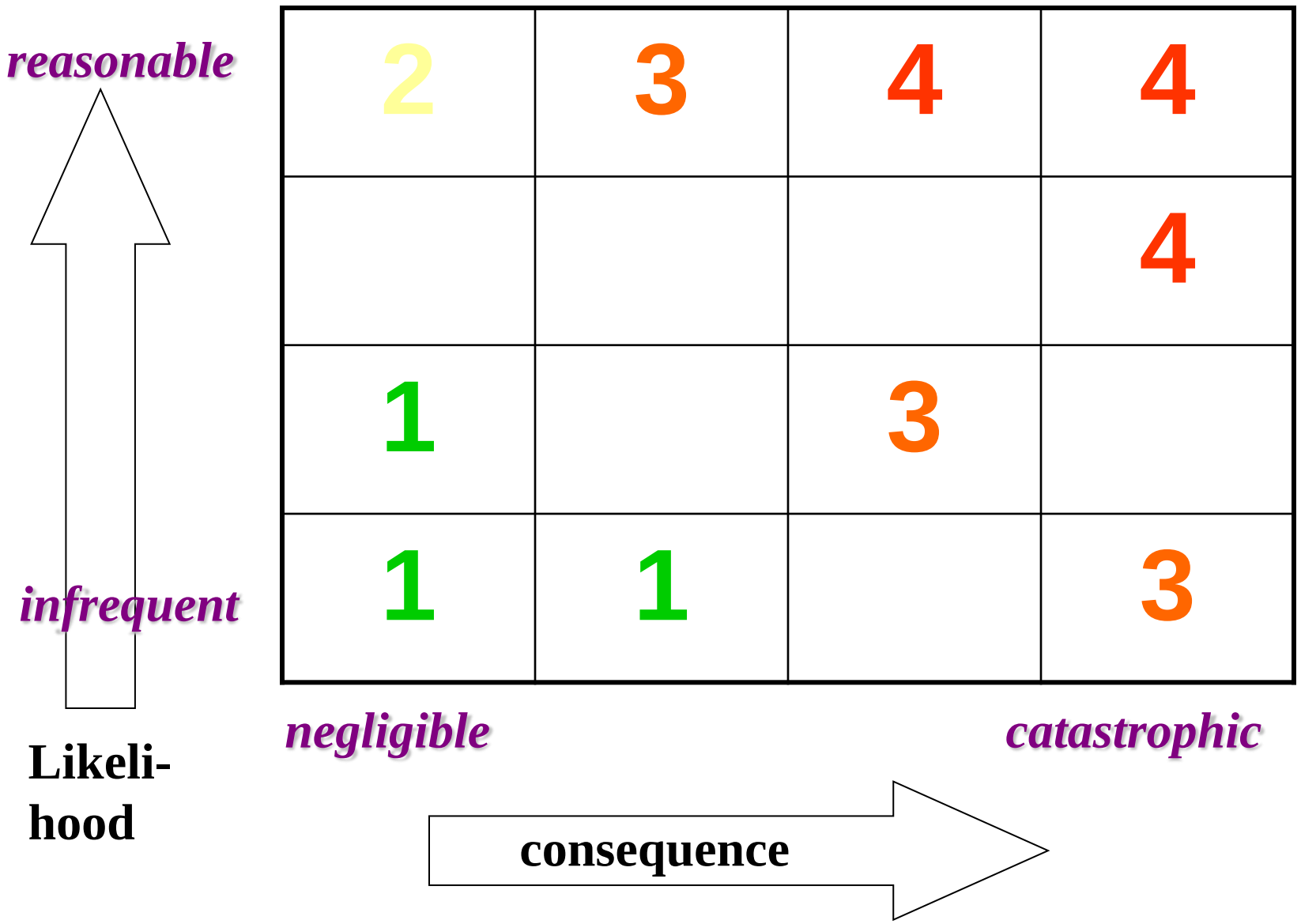
"Are we doing the right job?"

IEEE Std 1012

IEEE Standard for Software Verification and Validation



foundational component within
IEEE Software Engineering standards series


reasonable
infrequent
Likeli-
hood
2
3
4
4
4
1
3
1
1
3
negligible
catastrophic
consequence

RISK: “probability that some adverse circumstance will actually occur”

RISK: “any threat to the achievement of one or more key aims of the project”

RISK: “changes in the future that would lead to unacceptable situations”

“Yesterday’s *problems* are today’s ***risks***.”

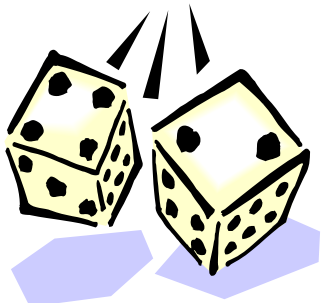
“Today’s ***risks*** are tomorrow’s *problems*.”

Risk Exposure can be ***unacceptable***

... even with **low probability** of occurrence

if **too great a consequence** of occurrence

Risk Management



Risk Exposure =

Likelihood of occurrence

X

Consequence of occurrence

Risk Management

Risk Avoidance ➔

reducing **probability**

of occurrence

Risk Mitigation ➔

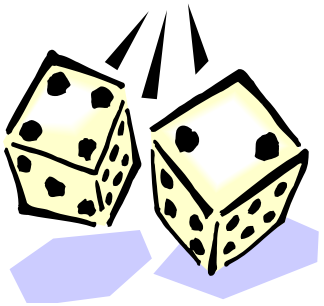
reducing **consequence**

of occurrence

$$ROI = \frac{\text{return}}{\text{investment}}$$

$$RORI = \frac{\text{risk exposure reduction}}{\text{reliability investment}}$$

Security Risk Exposure =



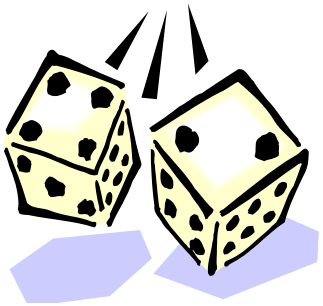
Probability of occurrence

~ frequency of exploitable defects ("vulnerabilities")

X

Consequence of occurrence

Security Risk Exposure =



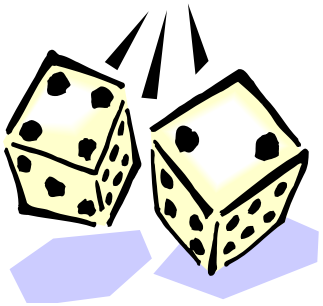
Probability of occurrence

(knowledge * skill * resources * motivation)

X

Consequence of occurrence

Security Risk Exposure =

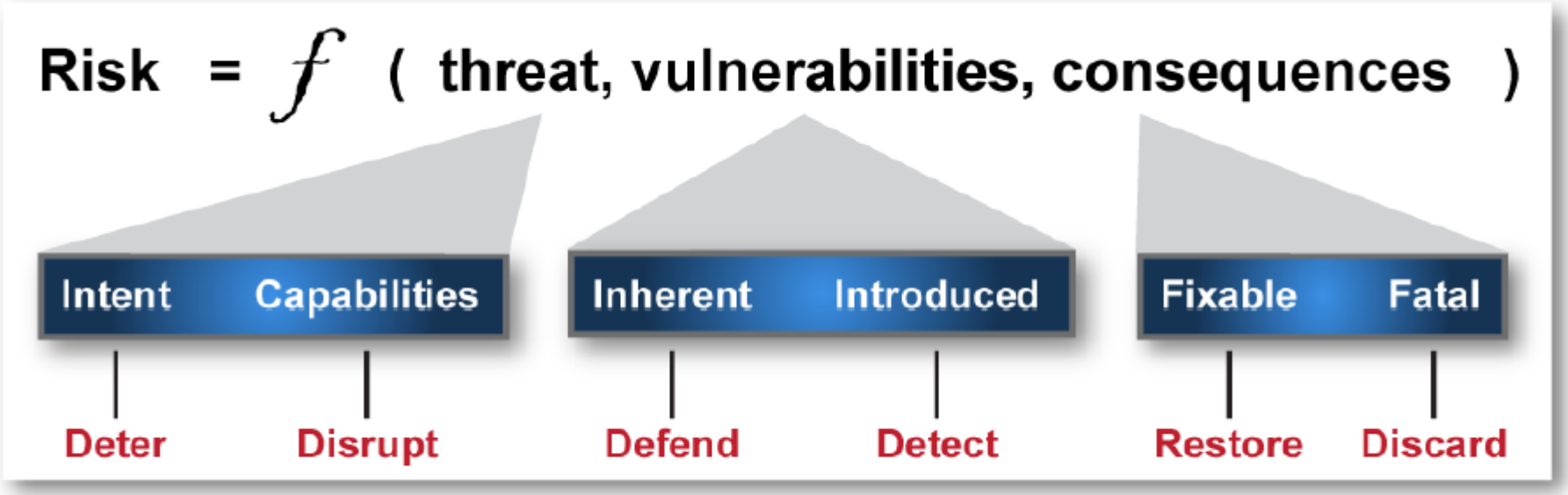


Probability of occurrence

(knowledge * skill * resources * motivation)

X

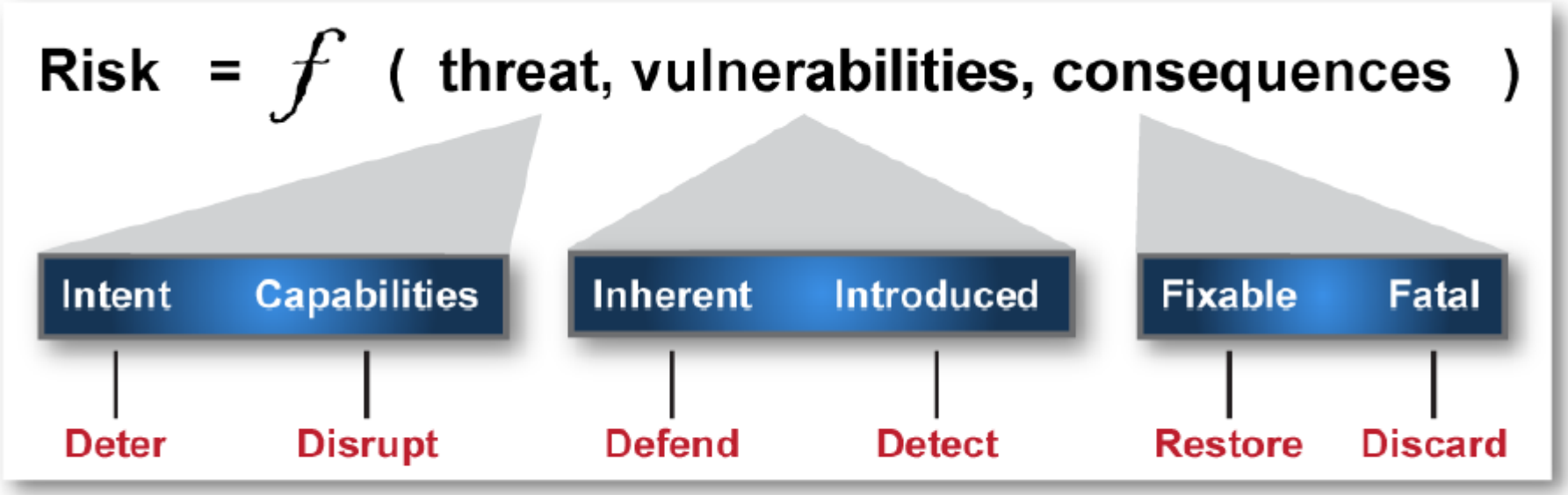
Consequence of occurrence



Risk Management Parameters

Resilient Military Systems and the Advanced Cyber Threat

Defense Science Board Task Force Report: January 2013



Risk Management Parameters

Intent = desire + expectance

Capabilities = resources + knowledge

inspections

walkthroughs

audits

reviews

static assessments

prototyping

simulation

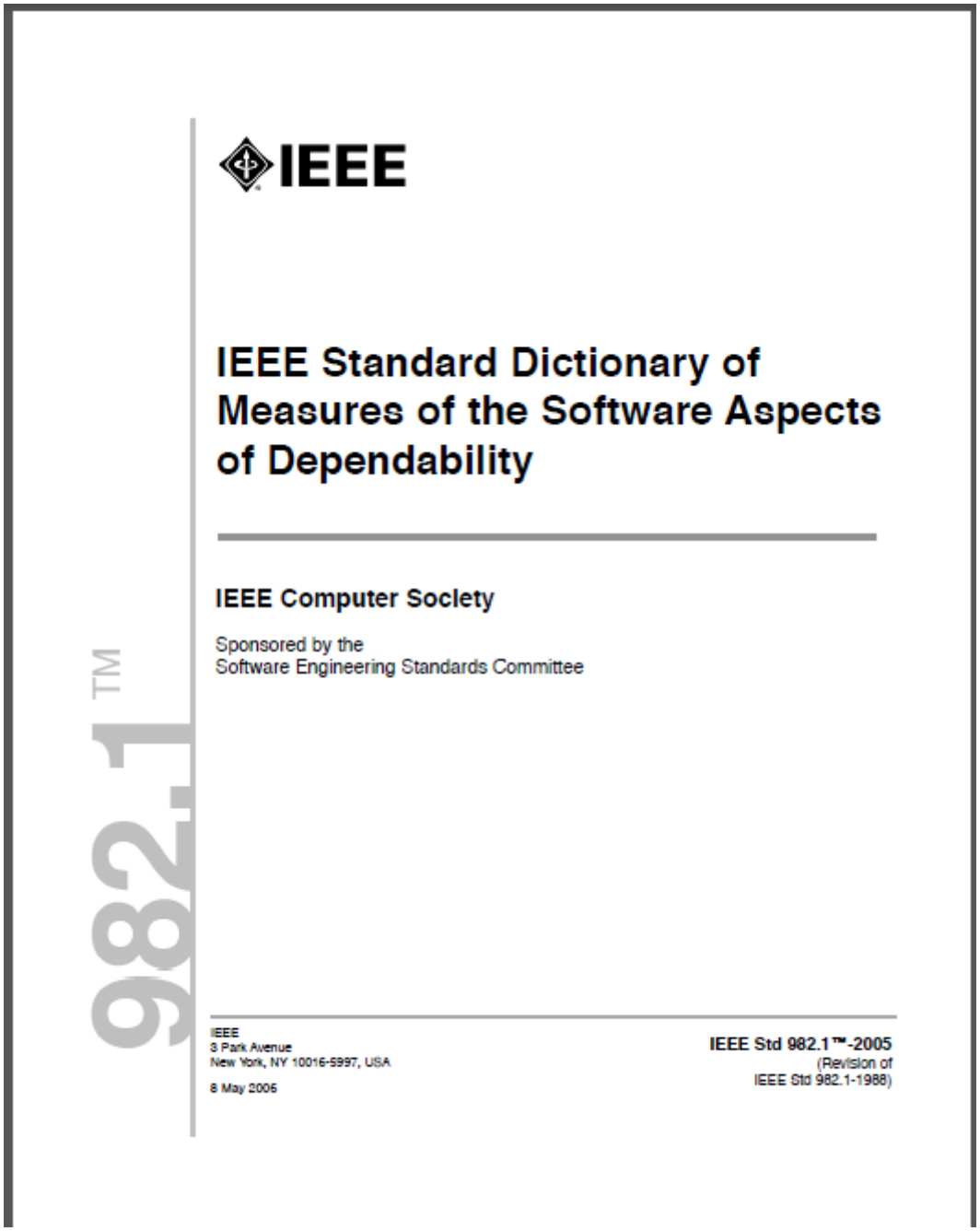
unit testing

integration testing

system testing

acceptance testing

dynamic assessments


IEEE
IEEE Standard Dictionary of Measures of the Software Aspects of Dependability
IEEE Computer Society
Sponsored by the
Software Engineering Standards Committee
982.1™
IEEE
3 Park Avenue
New York, NY 10016-5997, USA
8 May 2006
IEEE Std 982.1™-2005
(Revision of
IEEE Std 982.1-1988)

3. New reliability measures 4

3.1 General 4
3.2 Time to next failure (*s*) (Lyu [B12]) 4
3.3 Risk factor regression model (Schneidewind [B22]) 5
3.4 Remaining failures (Keller and Schneidewind [B11]) 6
3.5 Total test time to achieve specified remaining failures (Schneidewind [B20]) 7
3.6 Network reliability (Schneidewind [B19]) 8

4. Modified reliability measures 10

4.1 Defect density (982 #2) (Fenton and Pfleeger [B5] and Nikora et al. [B17]) 10
4.2 Test coverage index (982 #5) (Binder [B2]) 11
4.3 Requirements compliance (982 #23) (Fischer and Walker [B6]) 11
4.4 Failure rate (982 #31) (Lyu [B12]) 12

5. Retained reliability measures 13

5.1 Fault density (982 #1) (Musa [B14] and Nikora and Munson [B16]) 13
5.2 Requirements traceability (982 #7) (Fenton and Pfleeger [B5]) 15
5.3 Mean time to failure (MTTF) (982 #30) (Lyu [B12] and Musa et al. [B15]) 16

IEEE Std 982.1-2005
IEEE Standard Dictionary of Measures of the Software Aspects of Dependability

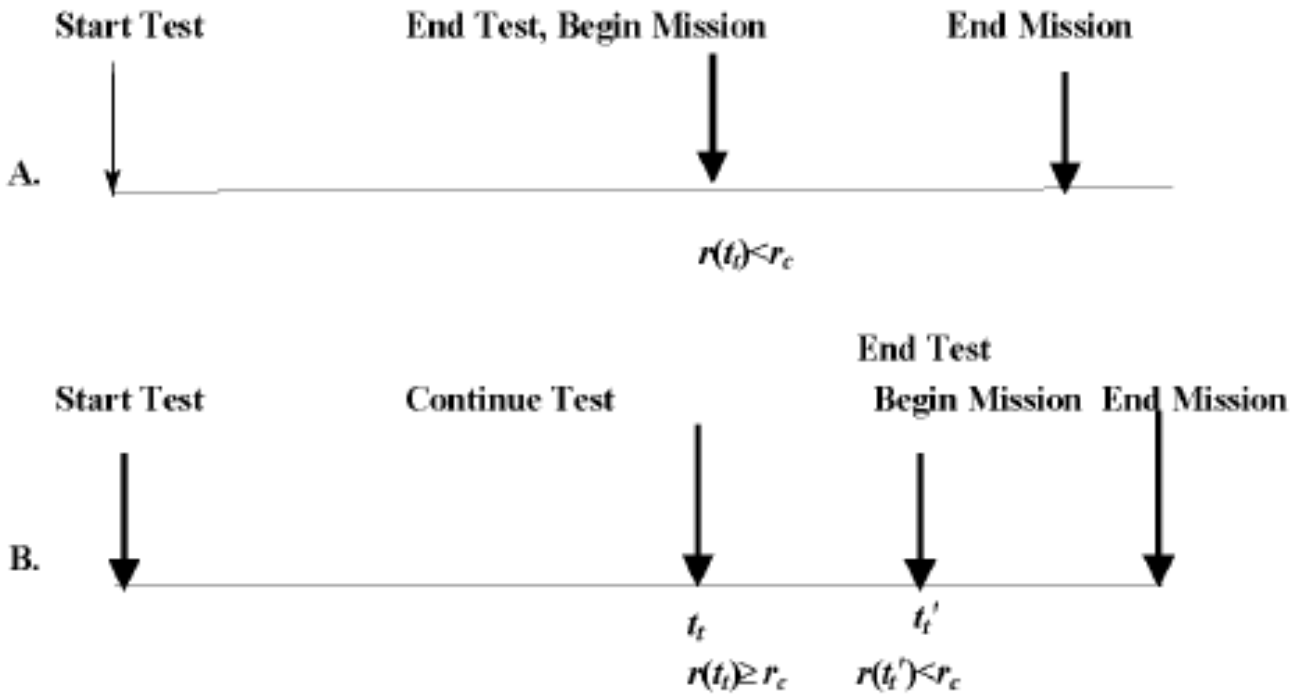


Figure B.1—Remaining failures criterion scenario

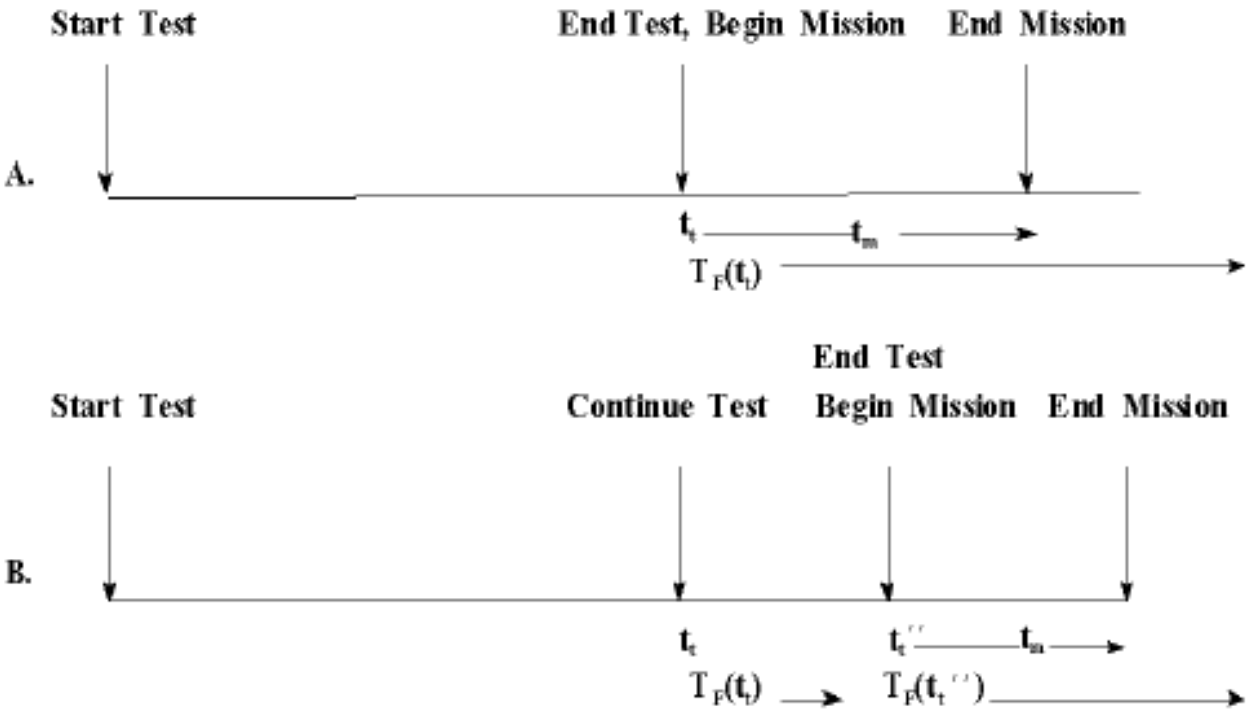
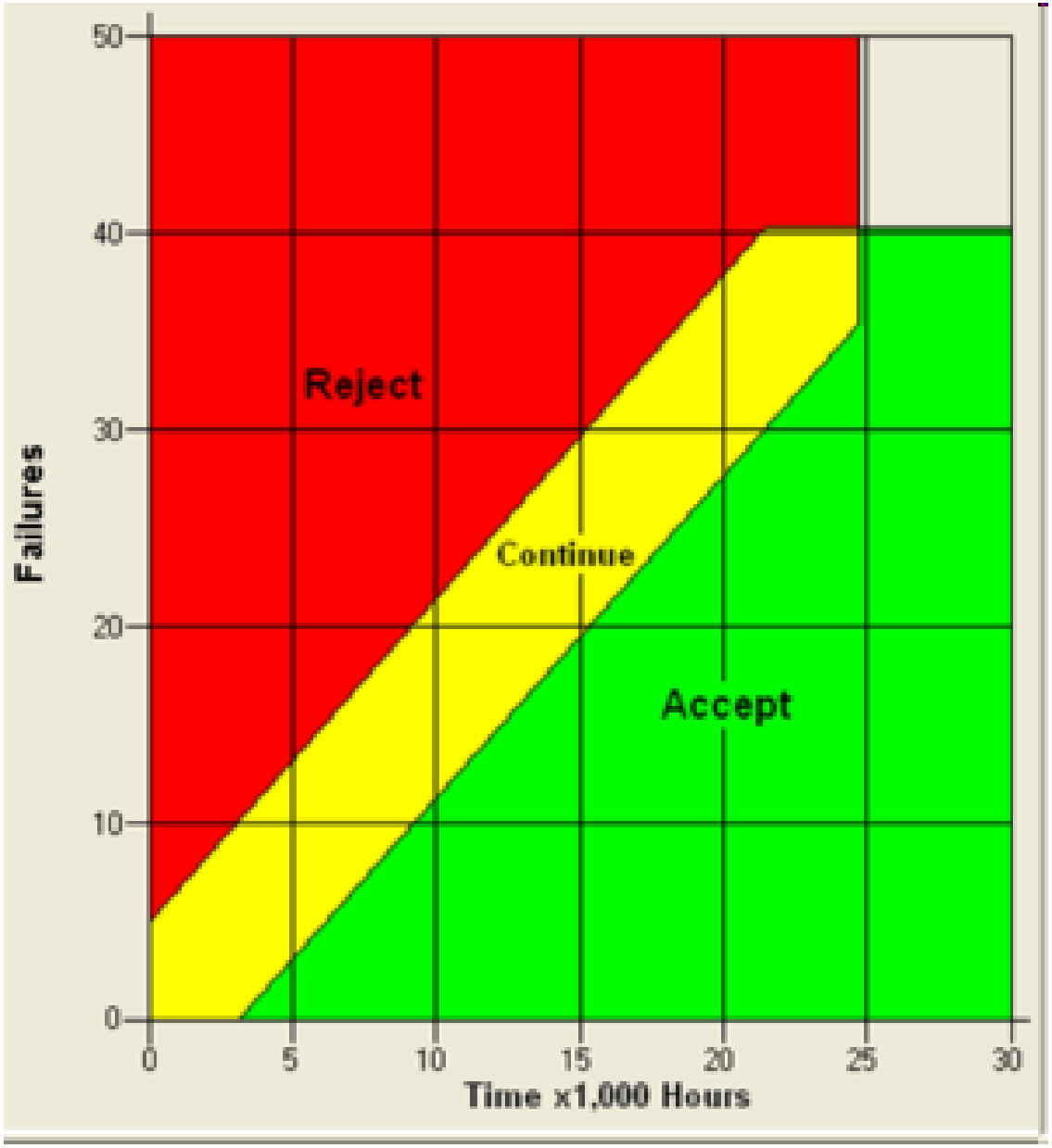


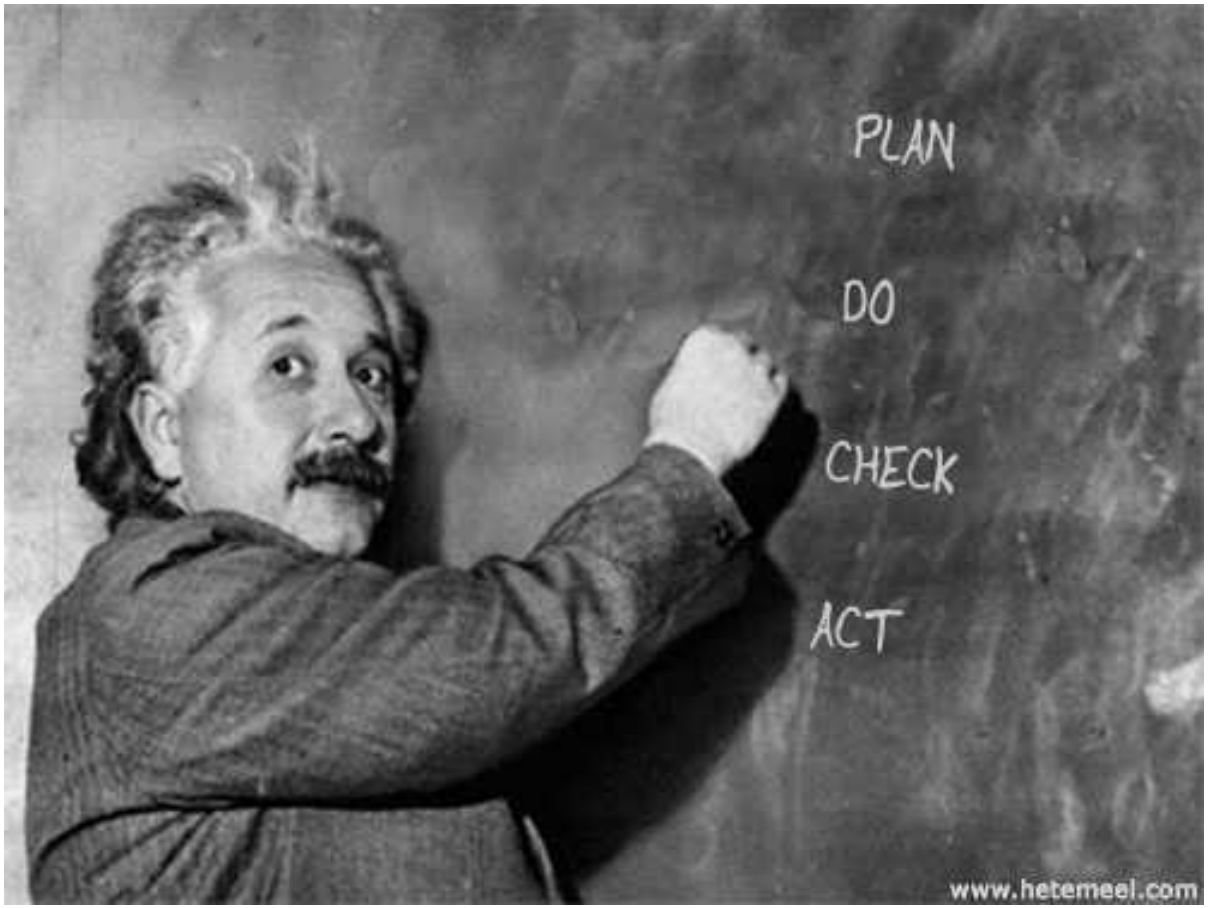
Figure B.2—Time to Next Failure criterion scenario

Operational Profile: Distribution of *Uses*

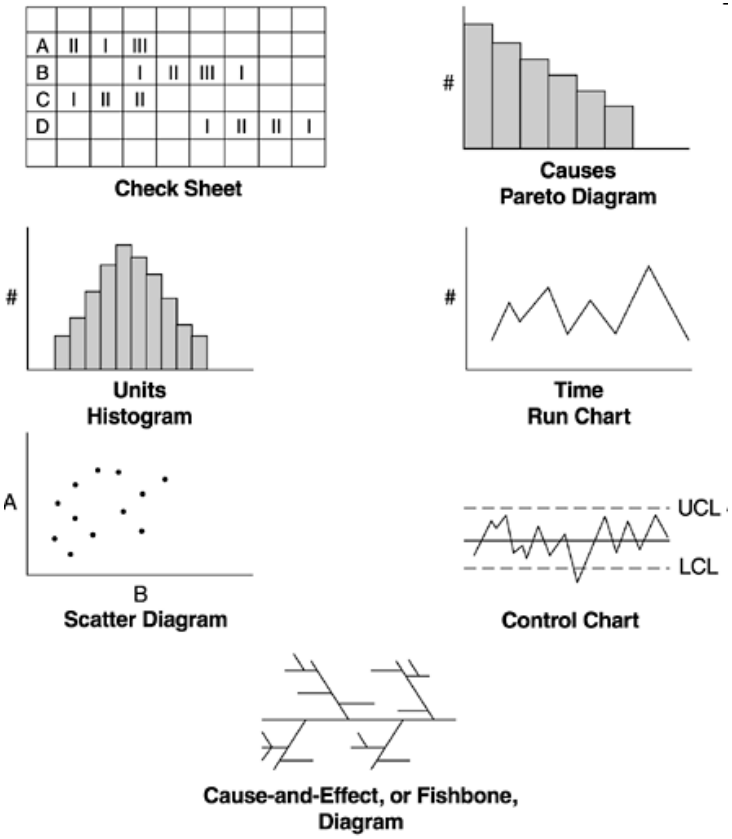
Operation	Occurrence probability	Initial test cases
Enter card	.332	66
Verify PIN	.332	66
Withdraw checking	.199	40
Withdraw savings	.066	13
Deposit checking	.040	8
Deposit savings	.020	4
Query status	.00664	1
Test terminal	.00332	1
Input to stolen card list	0.00058	0
Backup files	0.000023	0
Total	1	199



http://en.wikipedia.org/wiki/User:Wyatts/Draft_article_C


PLAN
DO
CHECK
ACT
www.hetemeel.com

Apply each “classic” tool to software ….

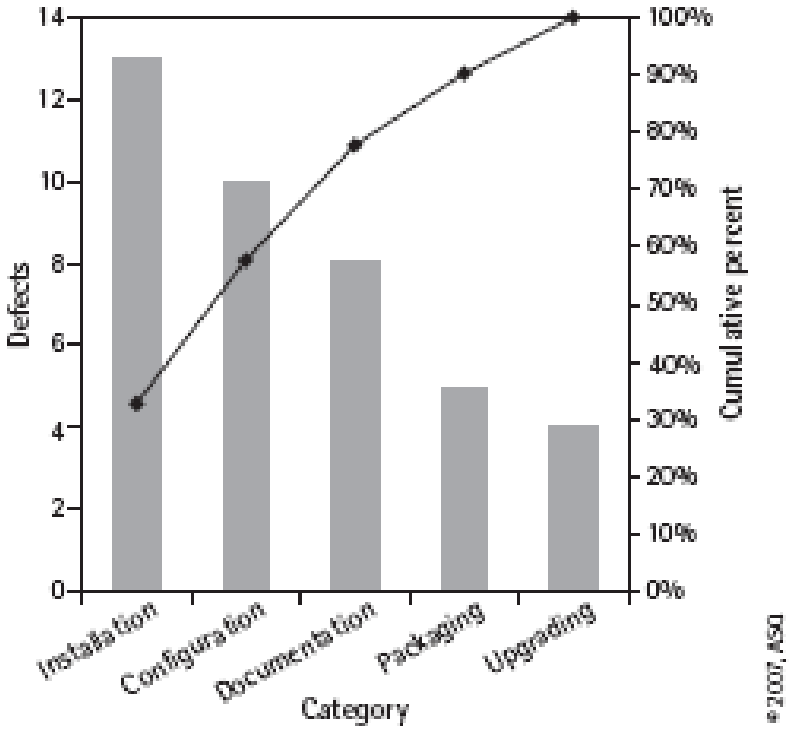


Apply **checklist/sheet** to software ….

Practice/organization	O1	O2	O3	O4	O5	O6	O7	O8
Level 3								
Allocated resources	–	○	–	–	–	–	–	–
Assigned responsibilities	○	○	○	○	○	○	–	○
Organizational policy	–	●	●	●	●	–	–	○
Data collection and use	–	–	–	–	–	–	–	–
Customized material	–	○	○	–	–	–	–	–
Training all	–	–	–	–	–	–	–	–
Defined process	–	●	●	●	○	●	–	–
Code inspections	–	○	–	○	○	–	–	–
Test case inspections	○	●	○	●	●	○	–	–
Level 2								
Training for leaders	–	–	–	–	–	–	–	–
Design inspections	○	●	○	●	●	○	○	○
Requirement inspections	○	●	●	●	●	○	○	○
Own estimate/average	2,8	2,8	3,3	3	2,7	2	2,5	2

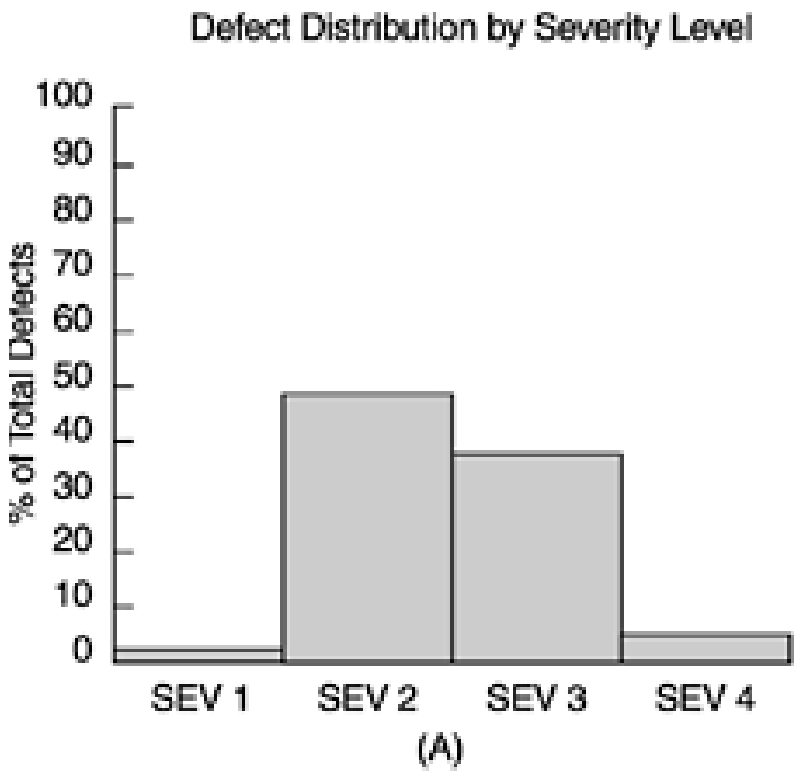
Sami Kollanus
Experiences from using ICMM in inspection process assessment
SOFTWARE QUALITY JOURNAL Volume 17, Number 2, 177-187, DOI: 10.1007/s11219-008-9067-2

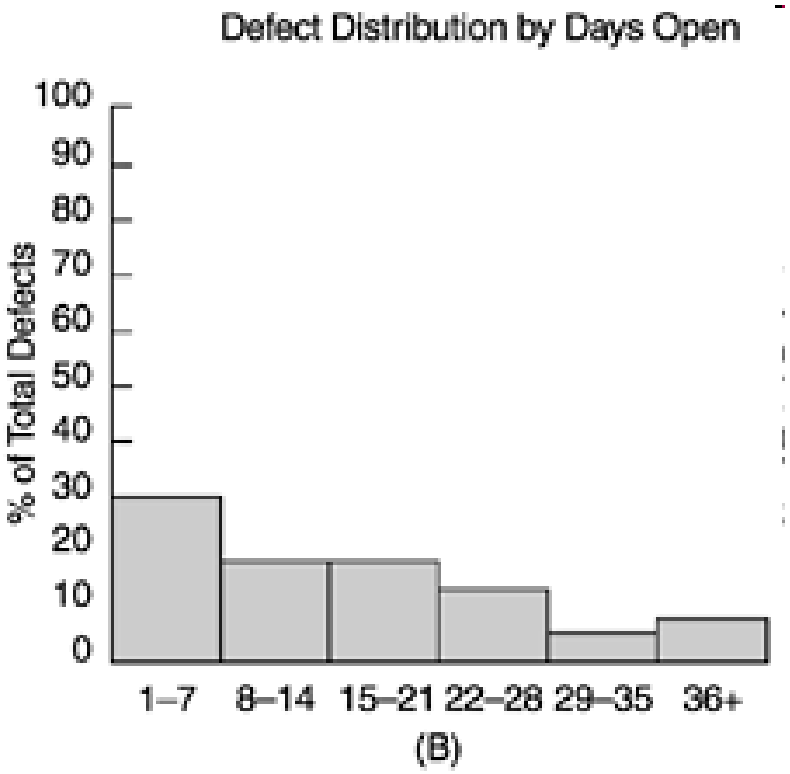
Apply **Pareto diagram** to software ….



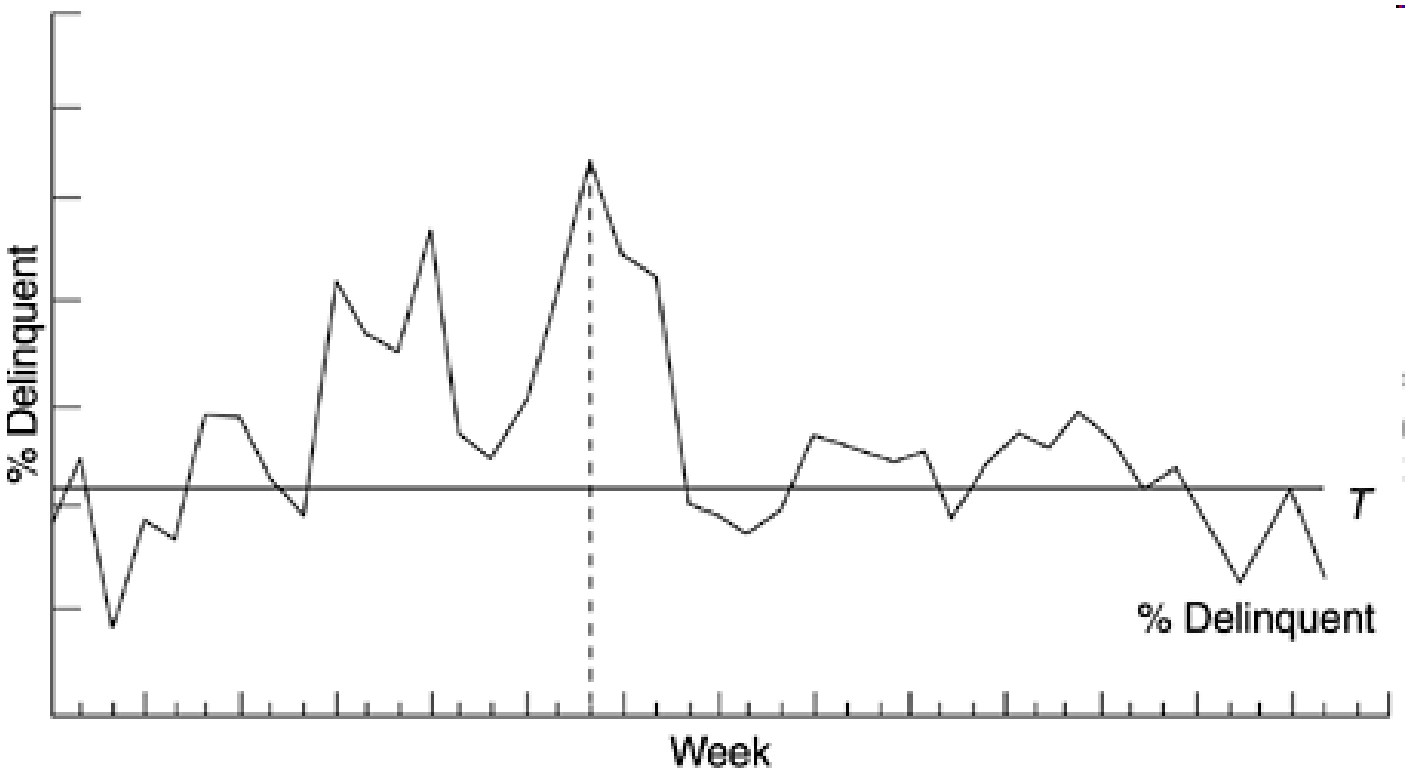
Heimann, David. A Bipartite Empirically Oriented Metrics Process for Agile Software Development. *Software Quality Professional.* Vol 9. No.2 (2007)

Apply **histogram** to software ….



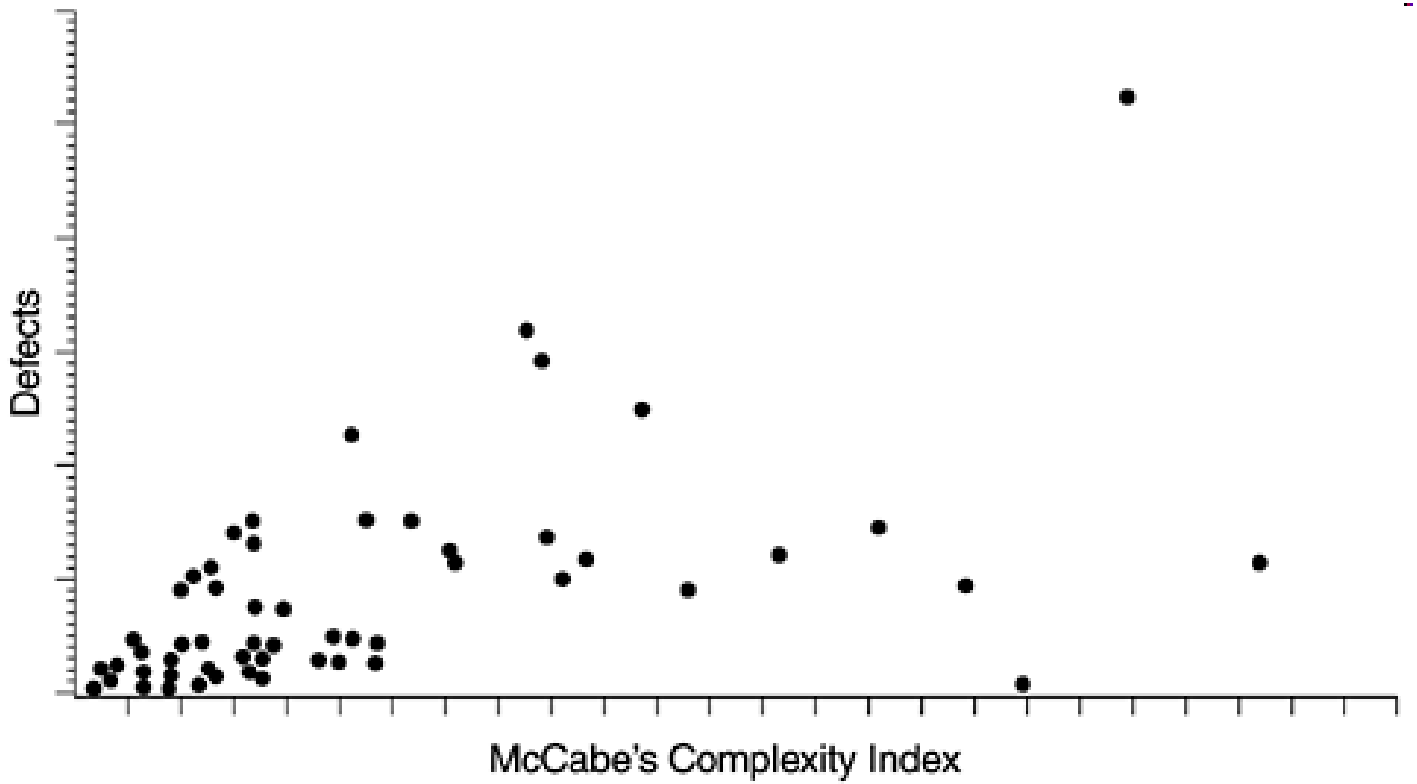


Apply **run chart** to software ….



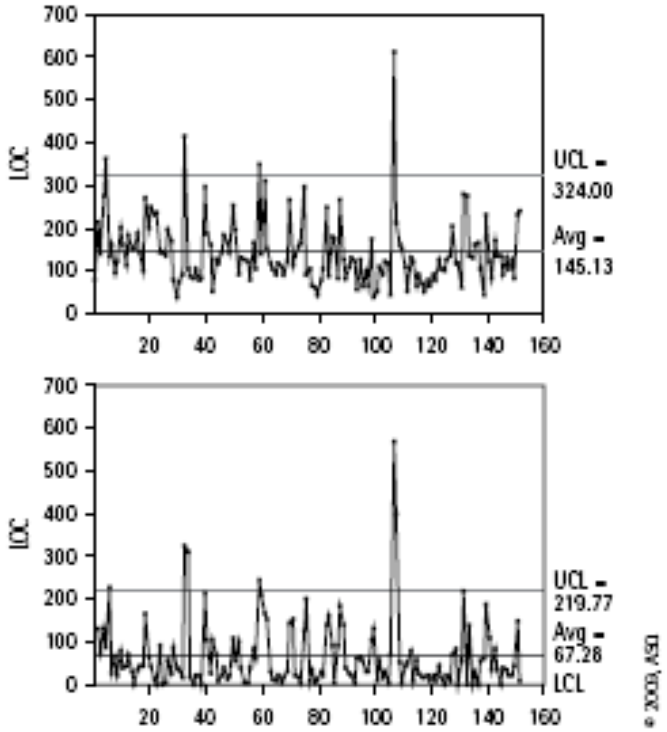
"Run Chart of Percentage of Delinquent Fixes" in **Metrics and Models in Software Quality Engineering**, 2nd edition, by Stephen H. Kan (2002). Used by permission.

Apply **scatter diagram** to software ….



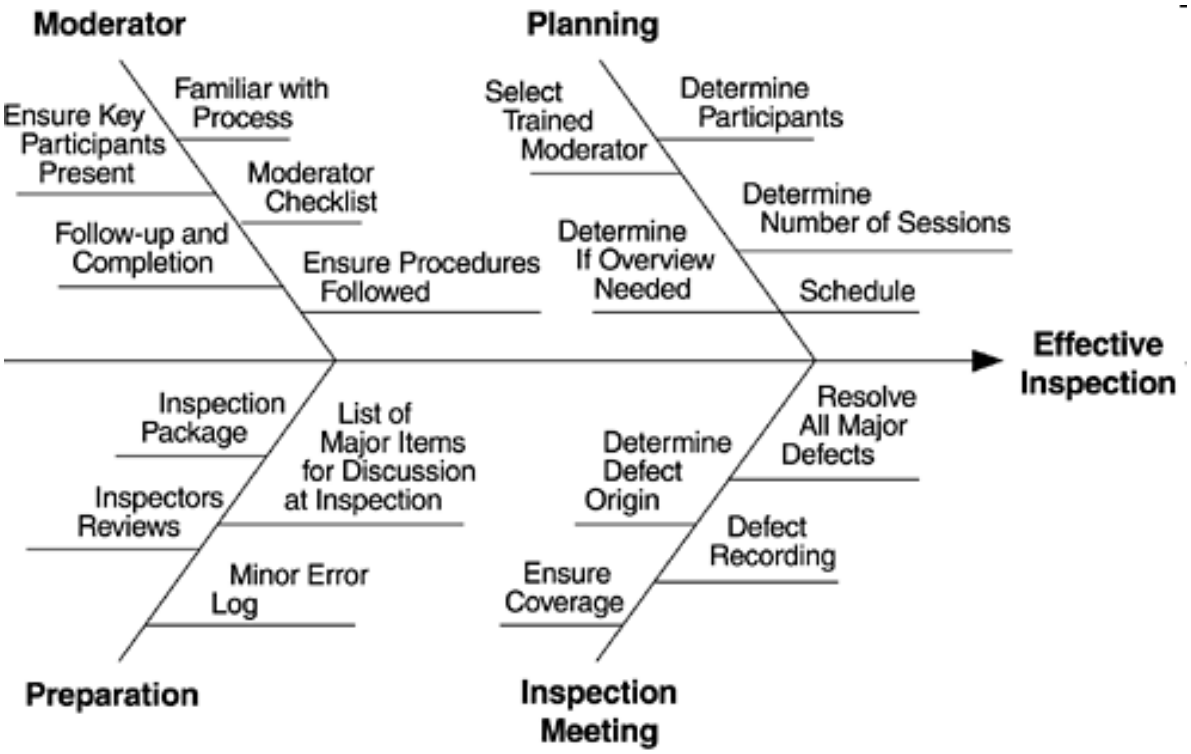
"Scatter Diagram of Program Complexity and Defect Level" in **Metrics and Models in Software Quality Engineering**, 2nd edition, by Stephen H. Kan (2002). Used by permission

Apply **control chart** to software ….



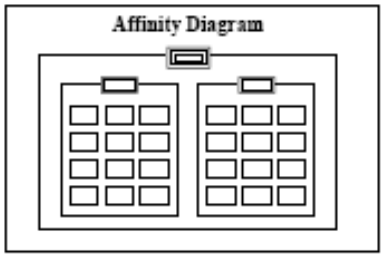
Paulk, Mark C., Kim LaScola Needy, and Jayant Rajgopal. Identify Outliers, Understand the Process. *Software Quality Professional.* Vol. 11, No.2 (2009)

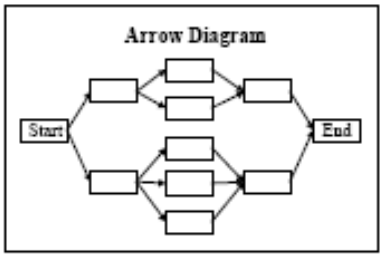
Apply **fishbone diagram** to software ….

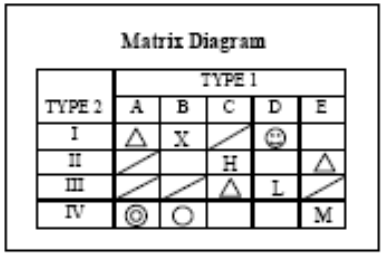


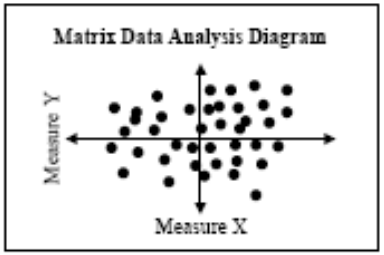
"Cause-and-Effect Diagram of Design Inspection" in **Metrics and Models in Software Quality Engineering**, 2nd edition, by Stephen H. Kan (2002). Used by permission

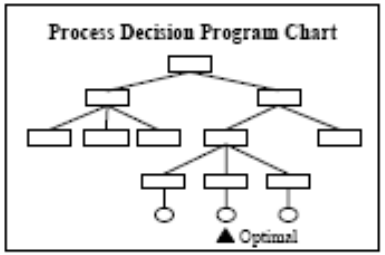
“NEW” PLANNING / MANAGEMENT TOOLS

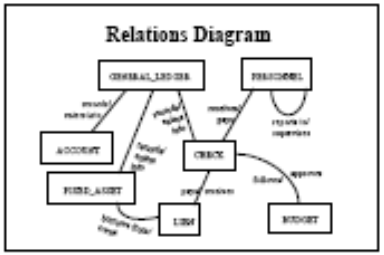


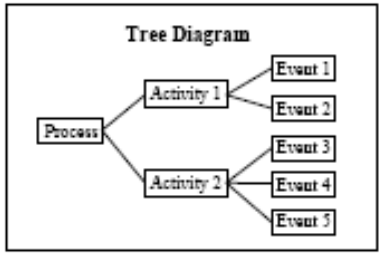




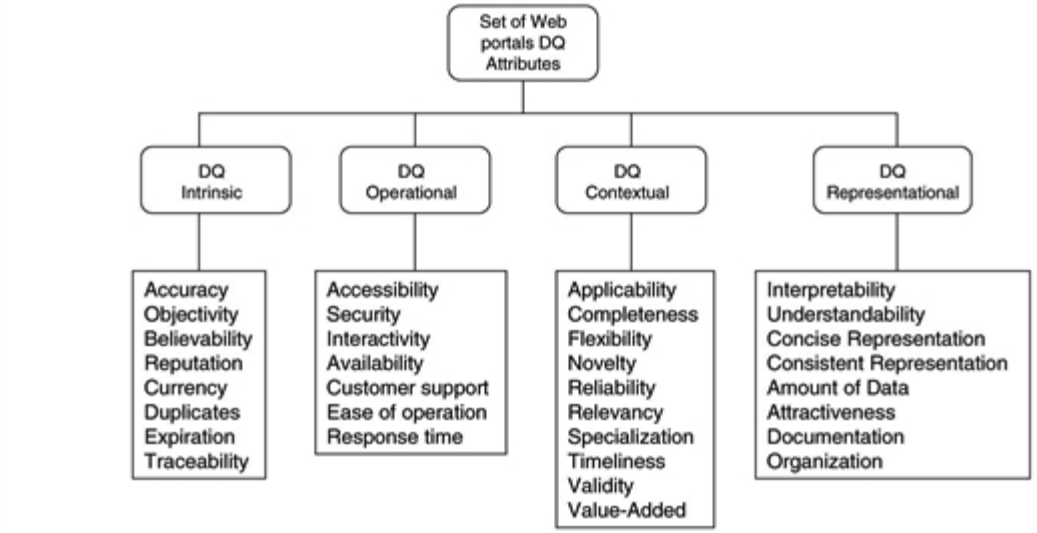








Apply **affinity diagram** to software ….

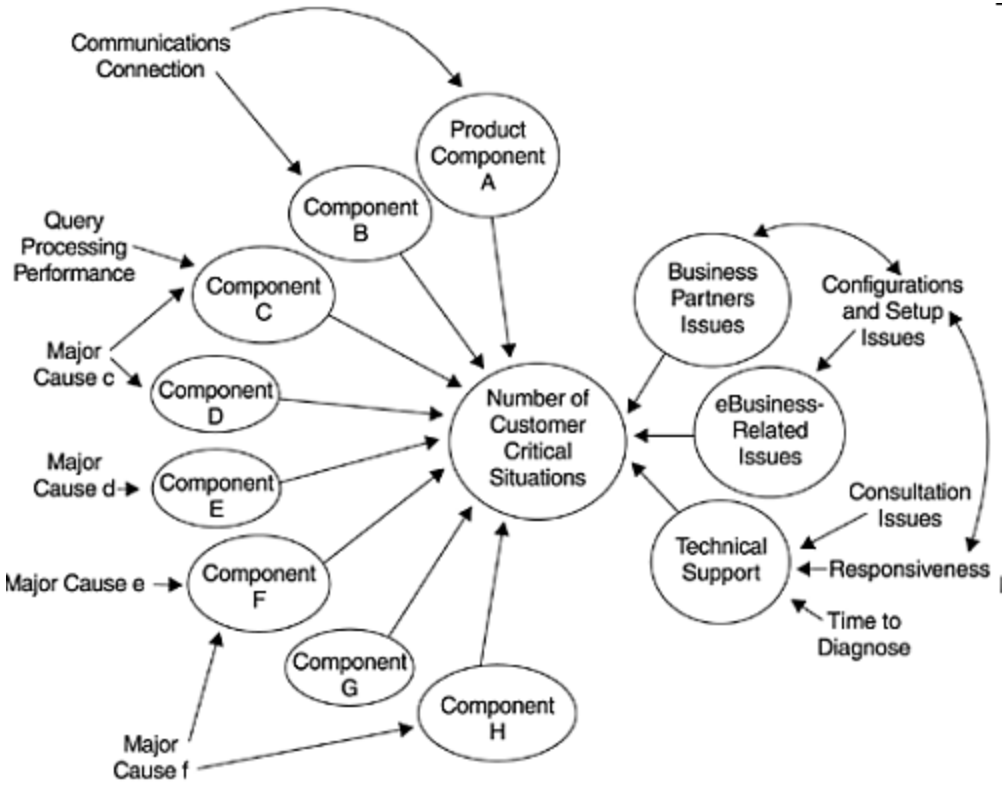


Angélica Caro et al

A proposal for a set of attributes relevant for Web portal data quality.

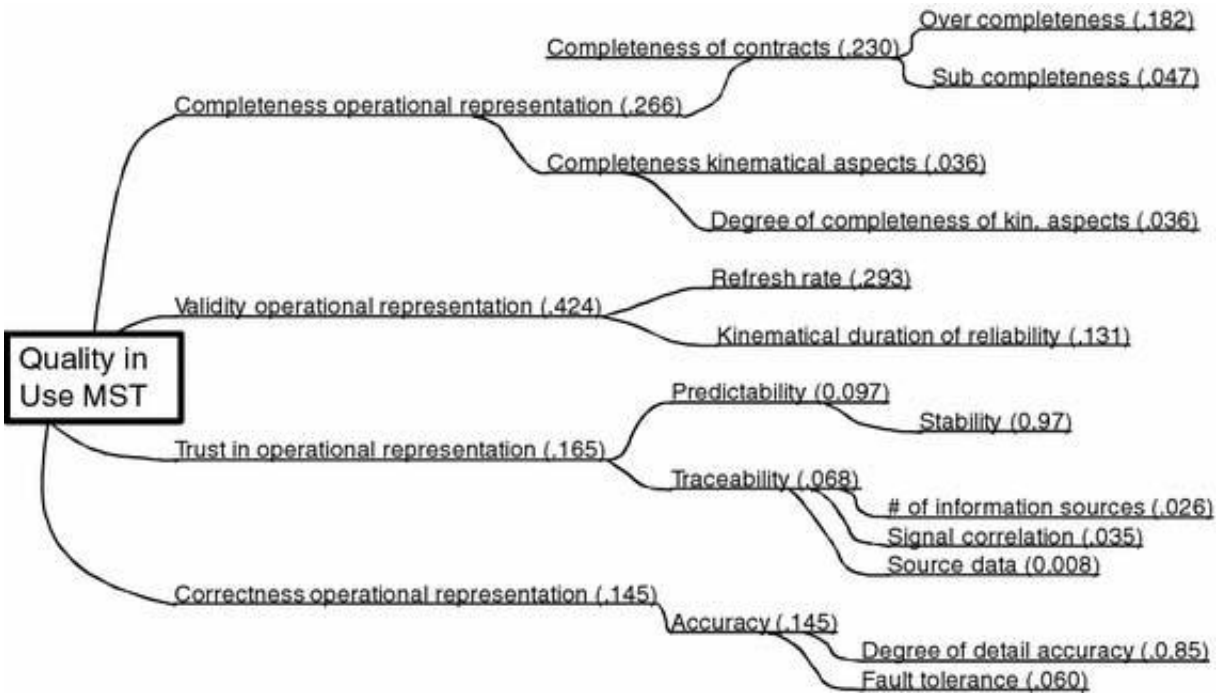
SOFTWARE QUALITY JOURNAL Volume 16, Number 4, 513-542, DOI: 10.1007/s11219-008-9046-7

Apply **relations diagram** to software ….



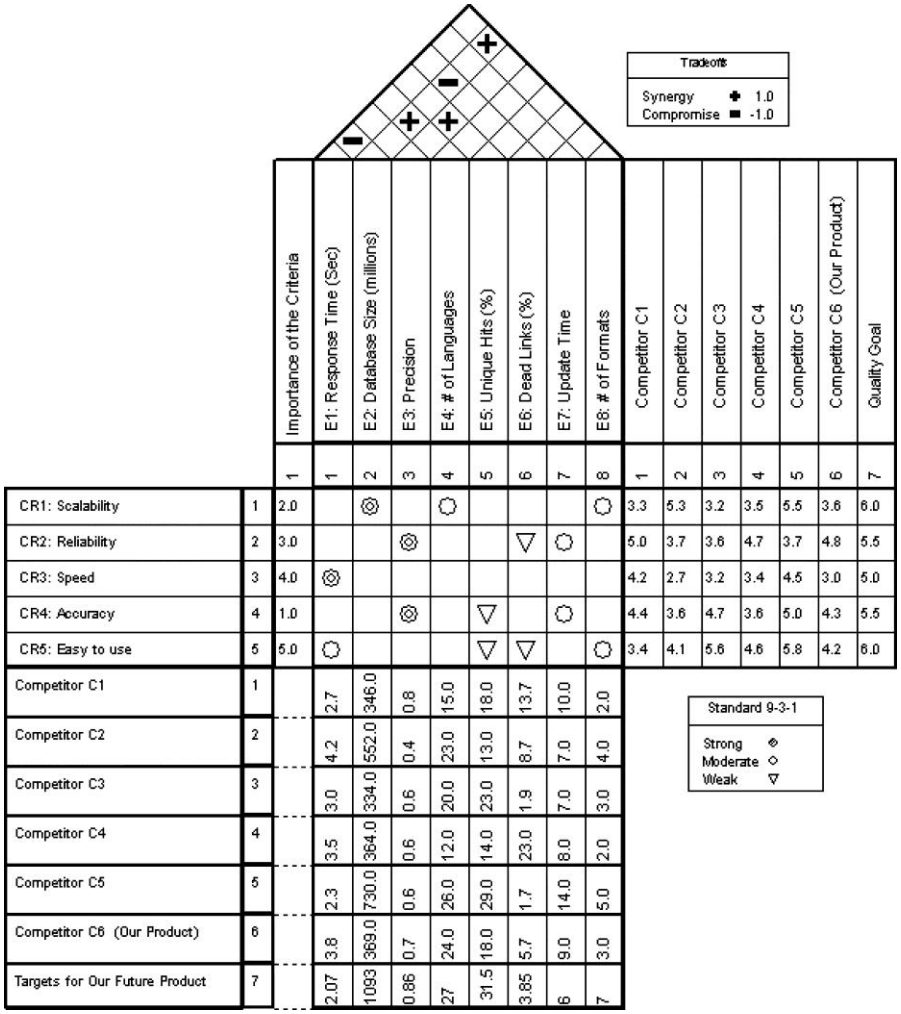
"A Diagram of Complex Relationships Associated with Customer-Critical Situations of a Software Product" in **Metrics and Models in Software Quality Engineering**, 2nd edition, by Stephen H. Kan (2002). Used by permission.

Apply **tree diagram** to software ….



Jos J. M. Trienekens • Rob J. Kusters • Dennis C. Brussel
Quality specification and metrication, results from a case-study in a mission-critical software domain
Software Qual J (2010) 18:469–490 DOI 10.1007/s11219-010-9101-z

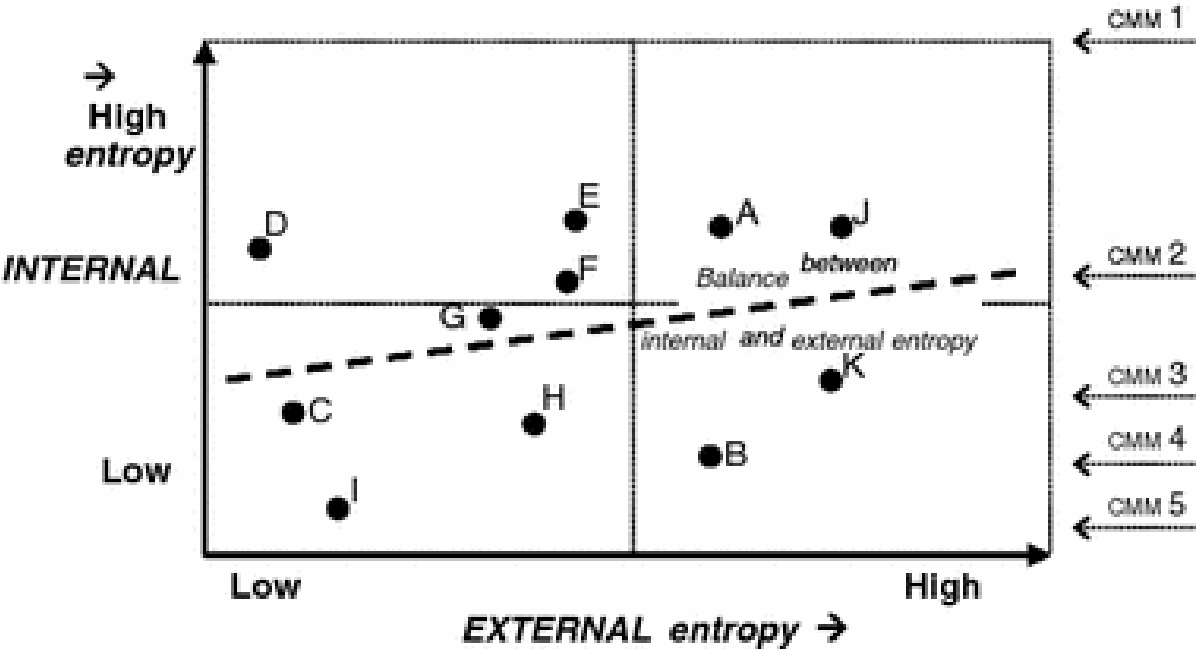
Apply **matrix diagram** to software ….



		Importance of the Criteria	E1: Response Time (Sec)	E2: Database Size (millions)	E3: Precision	E4: # of Languages	E5: Unique Hits (%)	E6: Dead Links (%)	E7: Update Time	E8: # of Formats	Competitor C1	Competitor C2	Competitor C3	Competitor C4	Competitor C5	Competitor C6 (Our Product)	Quality Goal
		1	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7
CR1: Scalability	1	2.0		◎		○				○	3.3	5.3	3.2	3.5	5.5	3.6	6.0
CR2: Reliability	2	3.0			◎			▽	○		5.0	3.7	3.6	4.7	3.7	4.8	5.5
CR3: Speed	3	4.0	◎								4.2	2.7	3.2	3.4	4.5	3.0	5.0
CR4: Accuracy	4	1.0			◎		▽		○		4.4	3.6	4.7	3.6	5.0	4.3	5.5
CR5: Easy to use	5	5.0	○				▽	▽		○	3.4	4.1	5.6	4.6	5.8	4.2	6.0
Competitor C1	1		2.7	346.0	0.8	15.0	18.0	13.7	10.0	2.0							
Competitor C2	2		4.2	552.0	0.4	23.0	13.0	8.7	7.0	4.0							
Competitor C3	3		3.0	334.0	0.6	20.0	23.0	1.9	7.0	3.0							
Competitor C4	4		3.5	364.0	0.6	12.0	14.0	23.0	8.0	2.0							
Competitor C5	5		2.3	730.0	0.6	26.0	29.0	1.7	14.0	5.0							
Competitor C6 (Our Product)	6		3.8	369.0	0.7	24.0	18.0	5.7	9.0	3.0							
Targets for Our Future Product	7		2.07	1093	0.86	27	31.5	3.85	6	7							

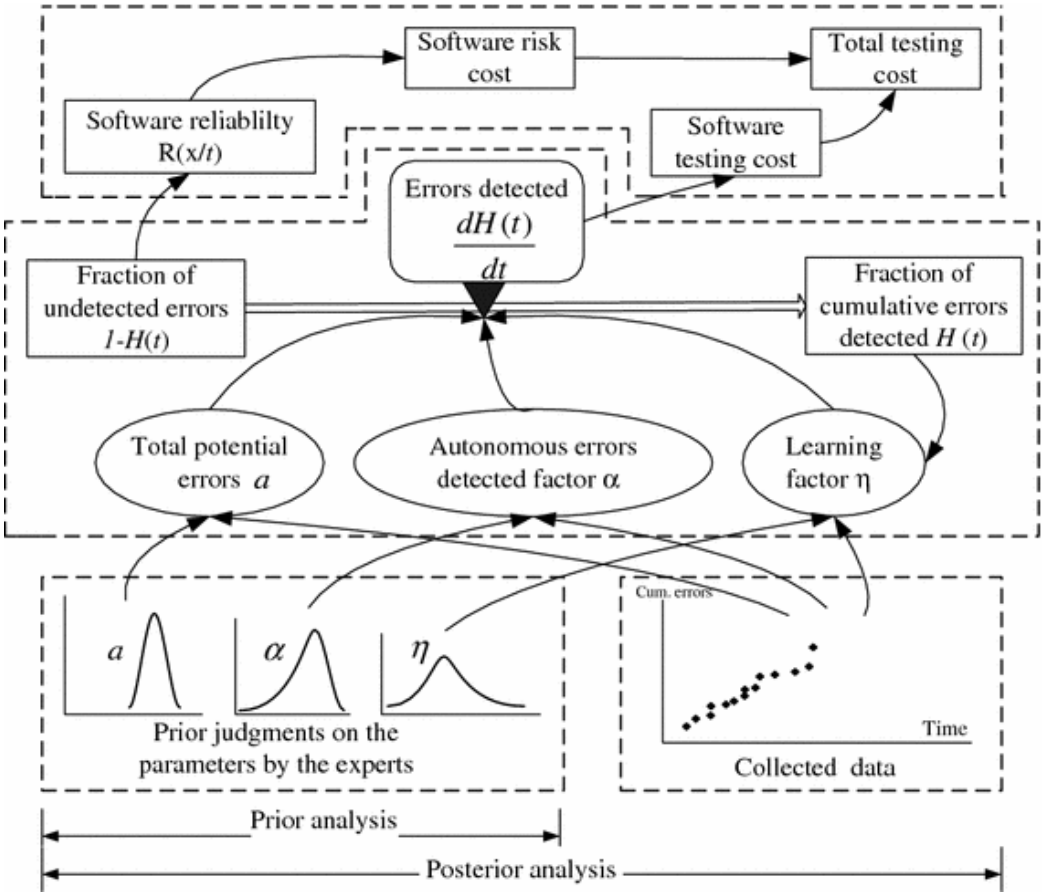
Frank Liu et al
A quantitative approach for setting technical targets
based on impact analysis in software quality function
deployment (SQFD)
Software Qual J (2006) 14: 113–134
DOI 10.1007/s11219-006-7598-y

Apply **matrix data analysis** to software ….



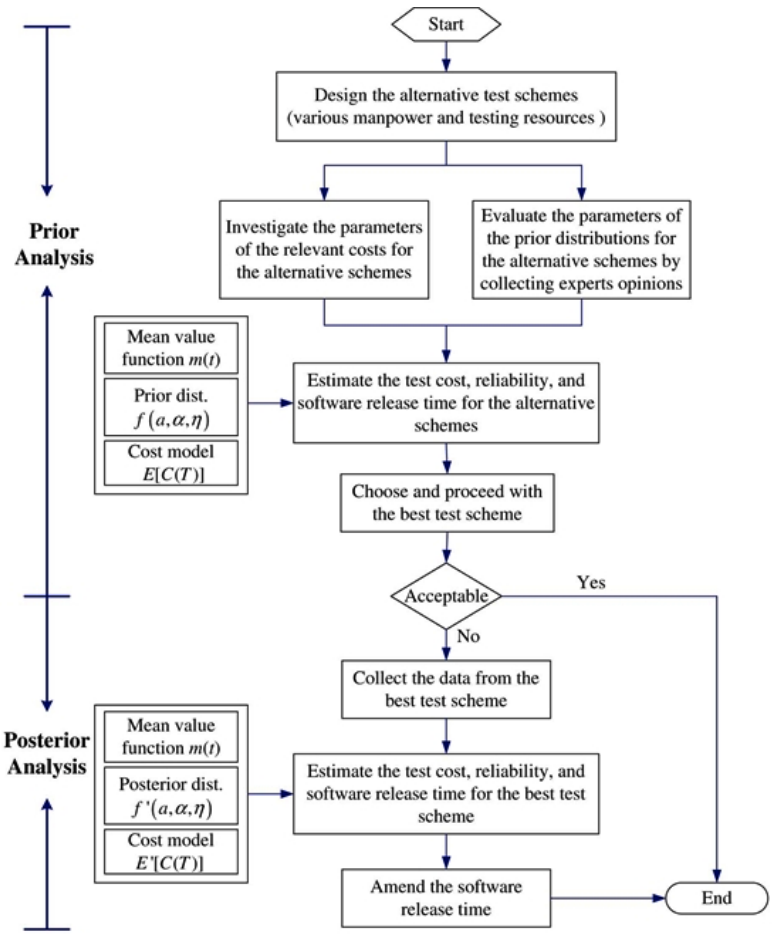
Jos J. M. Trienekens et al
Entropy based software processes improvement
SOFTWARE QUALITY JOURNAL Volume 17, Number 3, 231-243, DOI: 10.1007/s11219-008-9063-6

Apply **arrow diagram** to software ….



Kuei-Chen Chiu et al
Bayesian updating of optimal release time for software systems
SOFTWARE QUALITY JOURNAL Volume 17, Number 1, 99-120, DOI: 10.1007/s11219-008-9060-9

Apply **process decision program chart** to software ….



Kuei-Chen Chiu et al
Bayesian updating of optimal release time for software systems
SOFTWARE QUALITY JOURNAL Volume 17, Number 1, 99-120, DOI: 10.1007/s11219-008-9060-9

2:30 - 3:30 pm	Resources for the Journey
3:30 - 4:30 pm	Consolidation and Commitment … *Where to Go From Here*

stakeholder agreement

operational profiles

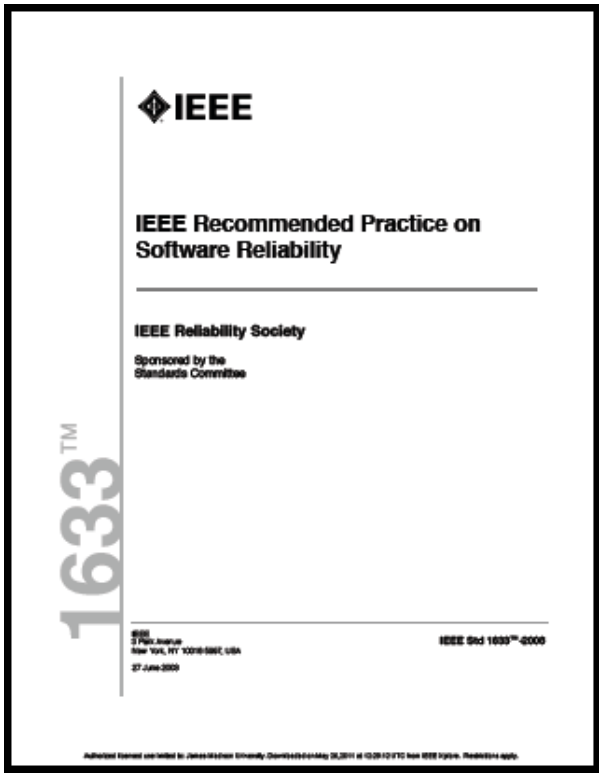
reviews

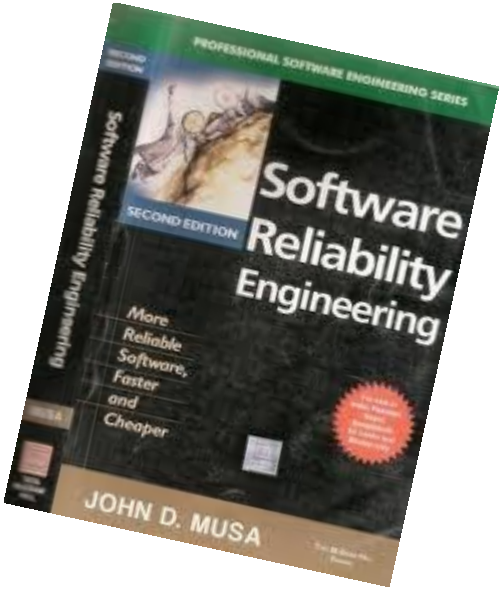
tests

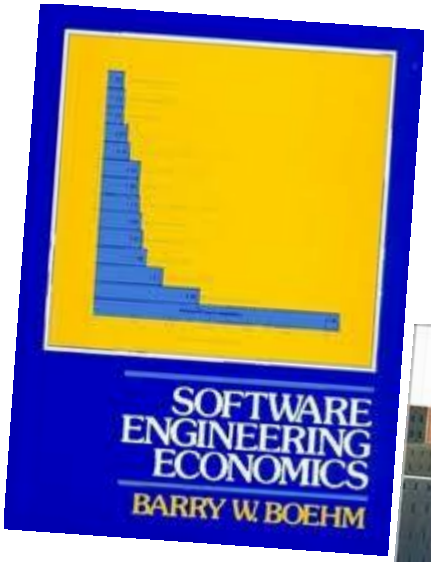
verifiable requirements

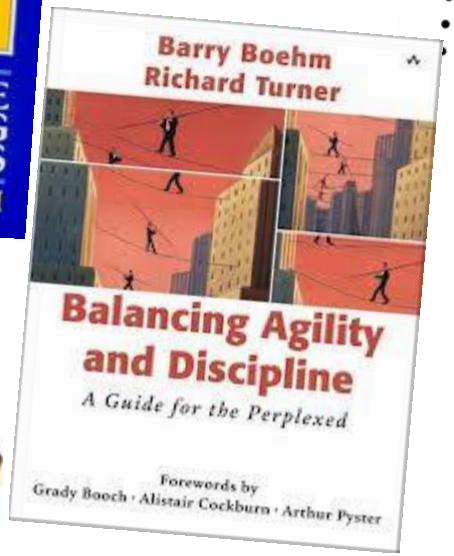
fault-tolerant design

Software Reliability



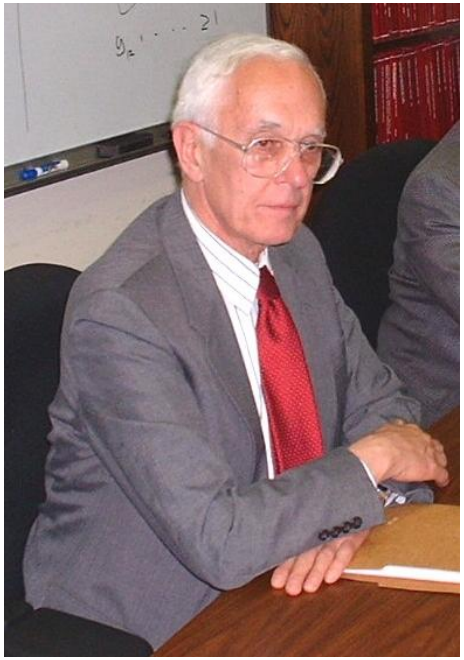


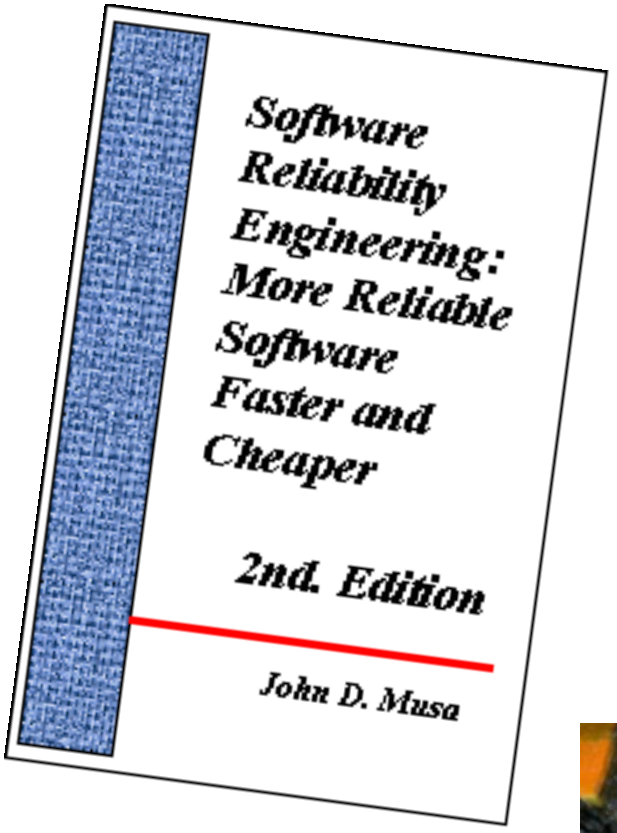

SOFTWARE
ENGINEERING
ECONOMICS
BARRY W. BOEHM

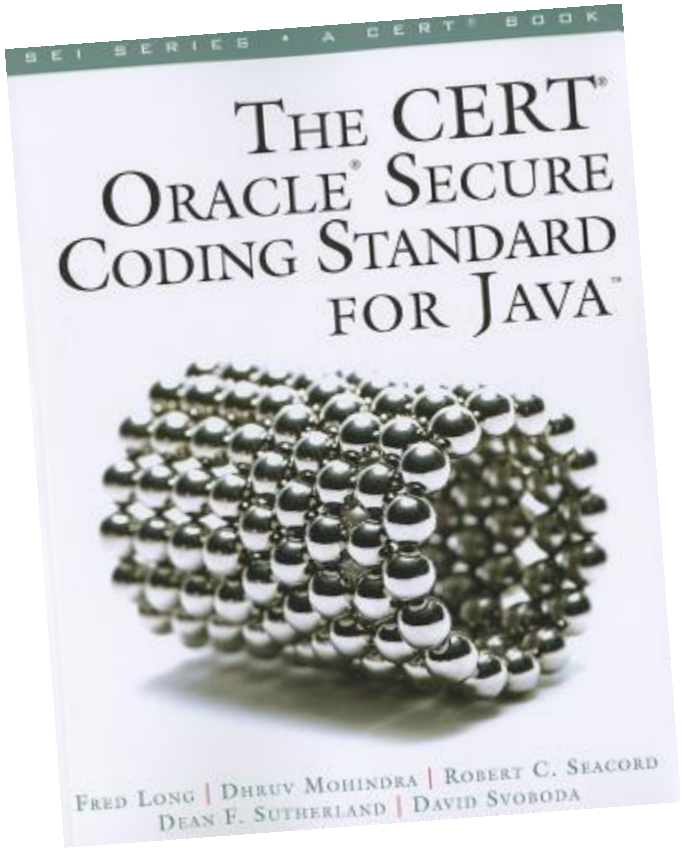

Barry Boehm
Richard Turner
Balancing Agility
and Discipline
A Guide for the Perplexed
Forewords by
Grady Booch · Alistair Cockburn · Arthur Pyster

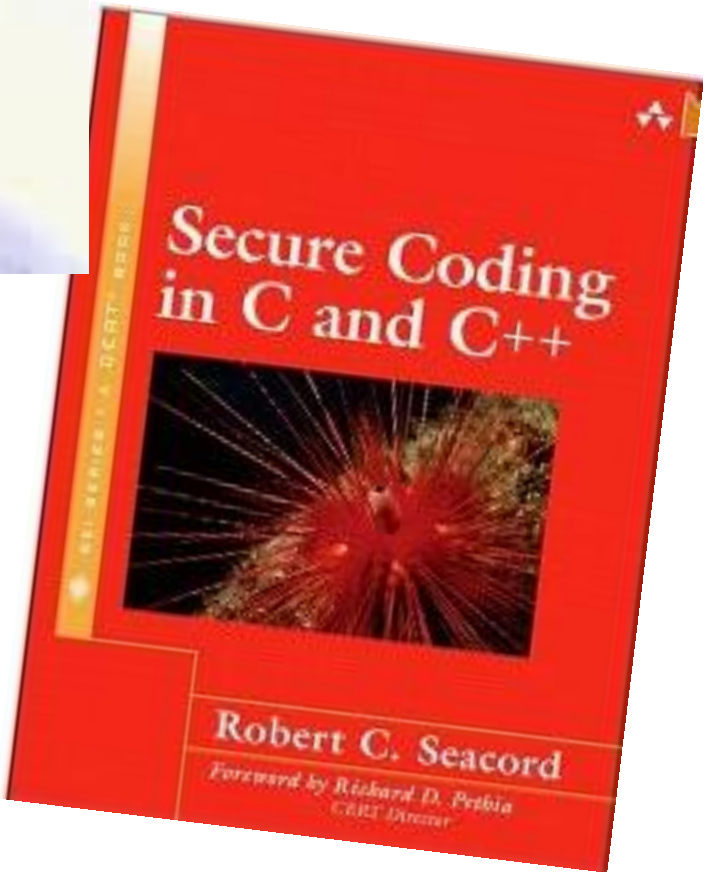

THE RISKS DIGEST
Forum On Risks To The Public In Computers And Related Systems
ACM Committee on Computers and Public Policy, Peter G. Neumann, moderator
Search RISKS using swish-e
The RISKS Forum is a moderated digest. Its USENET equivalent is comp.
Vol 26 Issue 96 (Wednesday 1 August 2012)
Vol 26 Issue 95 (Wednesday 25 July 2012) <= Latest Issue
Vol 26 Issue 94 (Tuesday 24 July 2012)

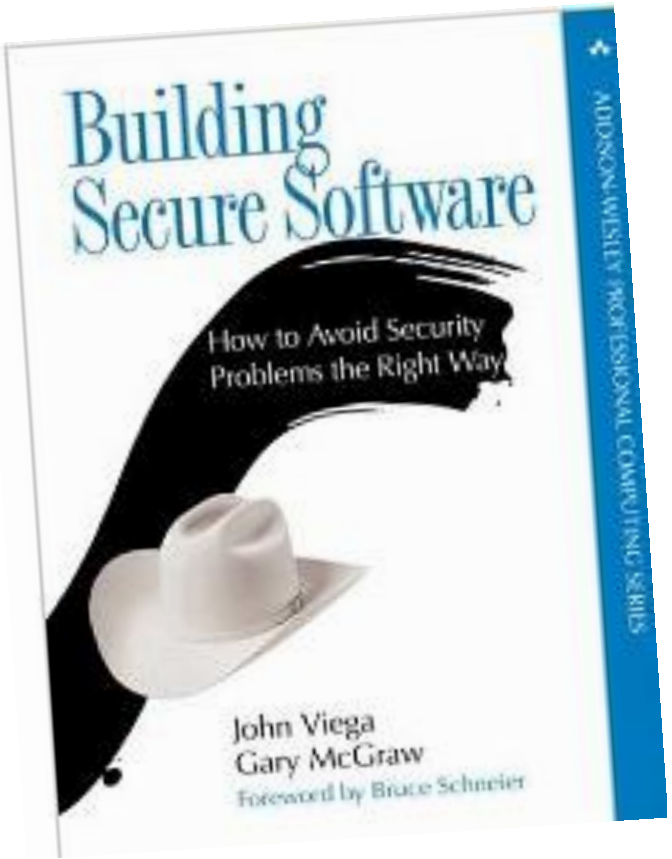

Build Security In
Setting a higher standard for software assurance
Homeland Security
Sponsored by DHS National Cyber Security Division
Search BSI:
Search
Privacy and Use
Actions
Navigational Links
Home
Mission
Articles [by Content Area]
Events
About Us
FAQs
Secure Coding Sites
Additional Resources
DHS SwA Web Site
DHS Software Assurance Resources
RSS Feeds
Contact Us
Build Security In Home
What is Build Security In?
Build Security In is a collaborative effort that provides practices, tools, guidelines, rules, principles, and other resources that software developers, architects, and security practitioners can use to build security into software in every phase of its development.
Introduction to Software Security
The Software Assurance Curriculum Project
Improve Security and Software Assurance: Tackle the CWE Top 25 Most Dangerous Software Errors
The Top 25 CWEs represent the most significant exploitable software constructs that have made software so vulnerable. Addressing these will go a long way in securing software, both in development and in operation. Read more and see the list of Top 25 CWE Most Dangerous Software Errors on the Software Assurance Community Resources and Information Clearinghouse website.
Consistent with this list is the Top 10 Project by the Open

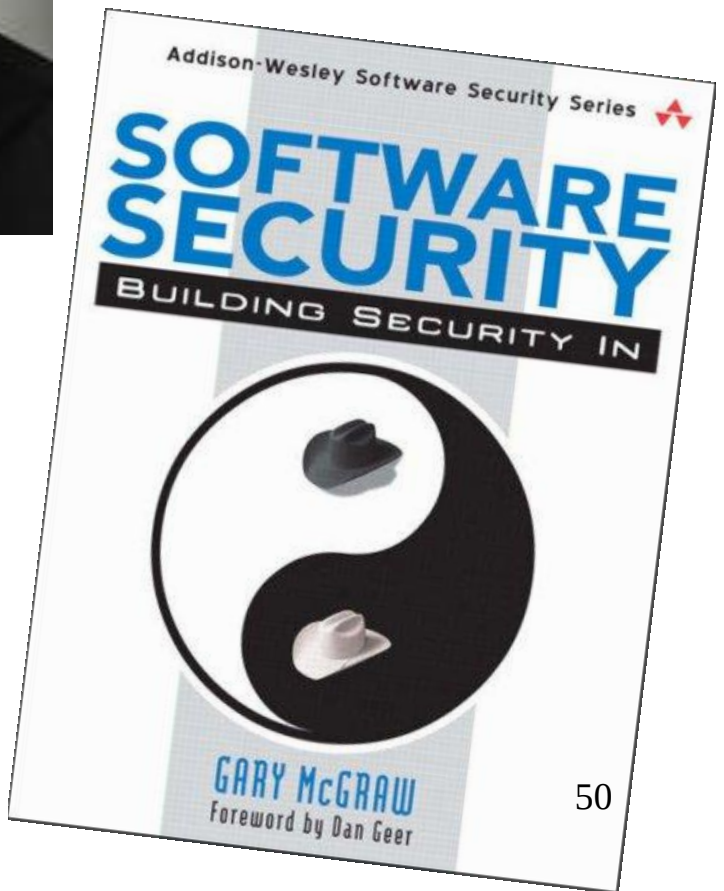


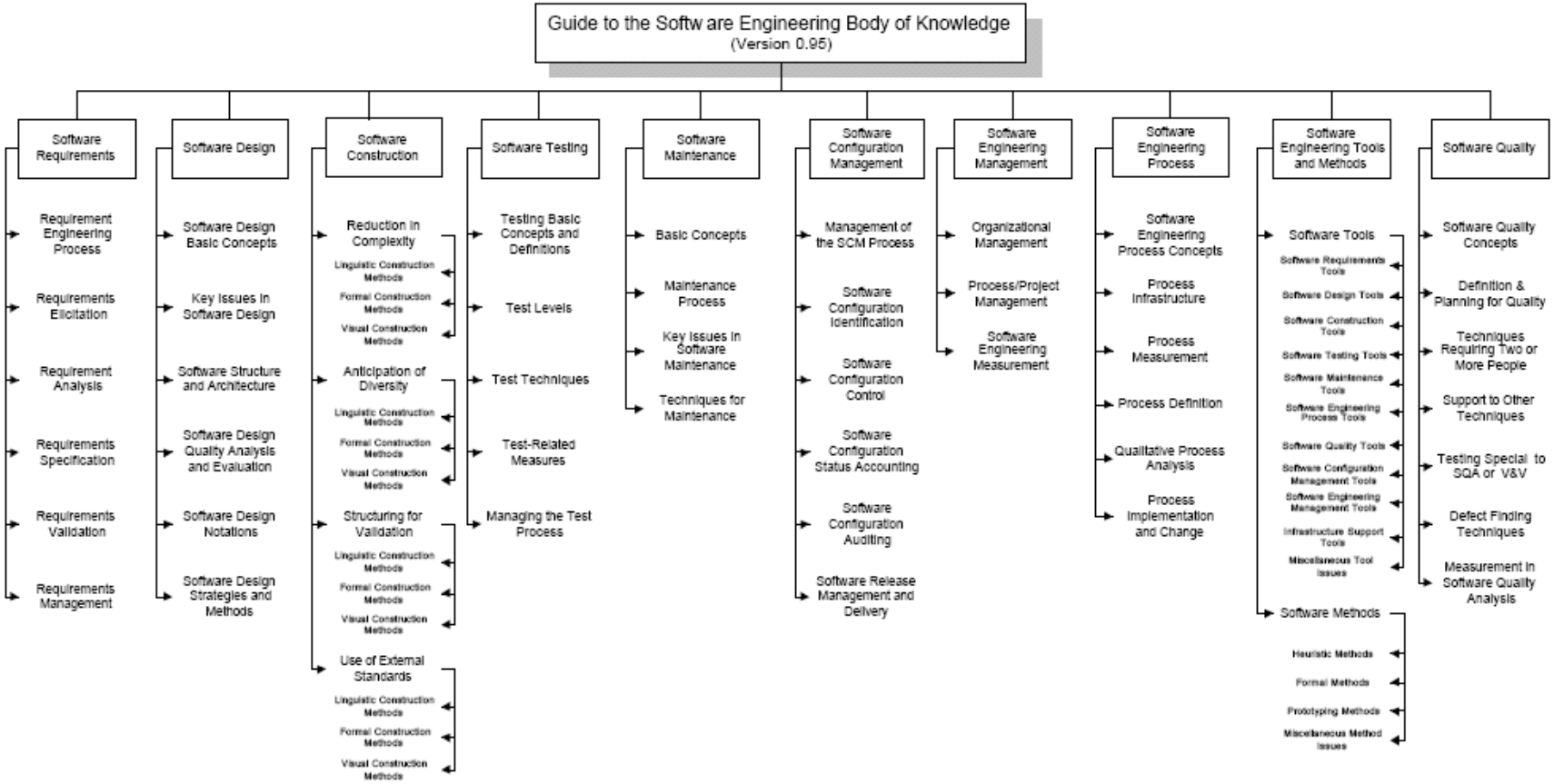

Software Reliability Engineering: More Reliable Software Faster and Cheaper
2nd. Edition
John D. Musa

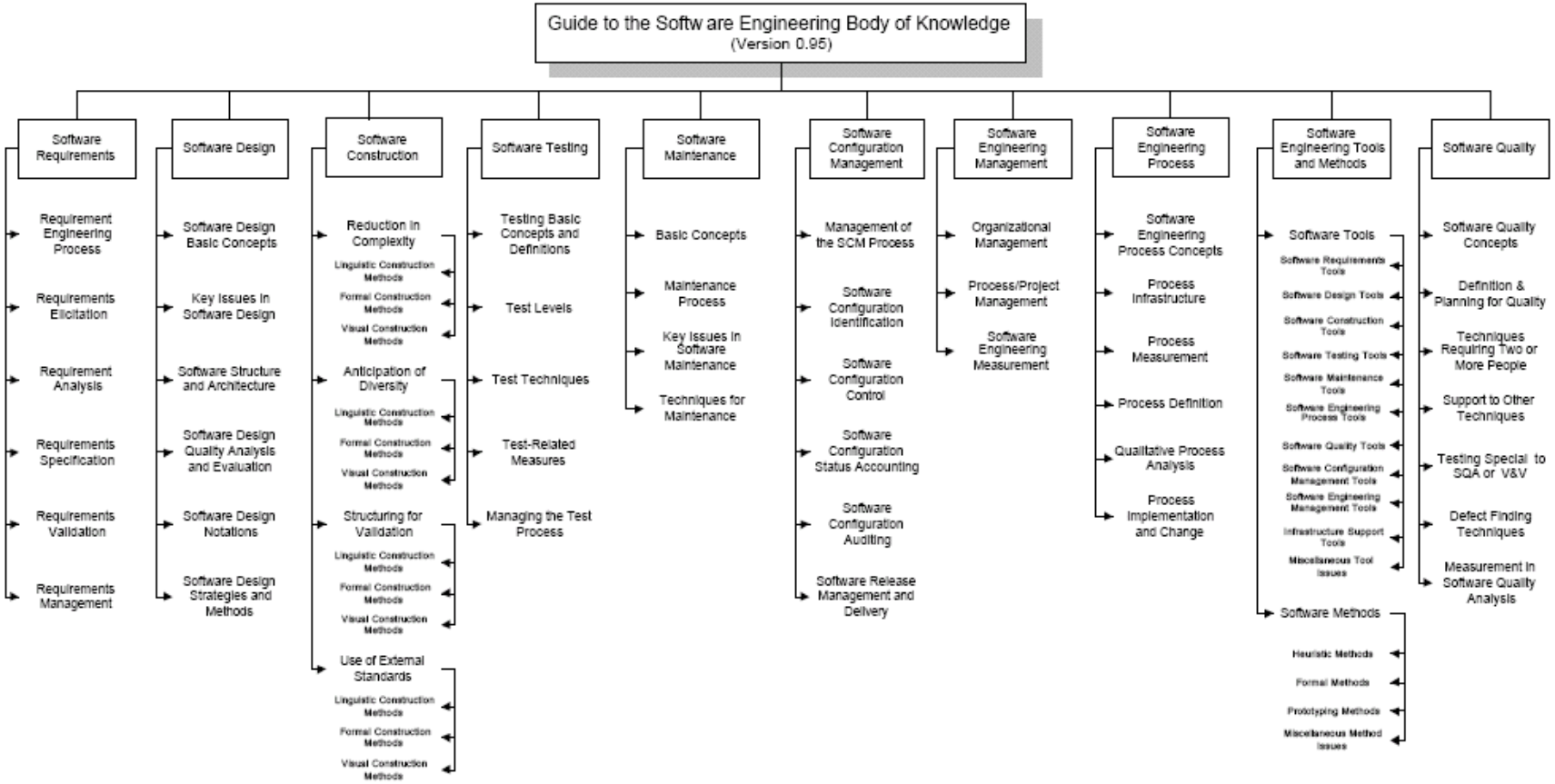

SEI SERIES • A CERT BOOK
THE CERT® ORACLE® SECURE CODING STANDARD FOR JAVA™
FRED LONG | DHRUV MOHINDRA | ROBERT C. SEACORD
DEAN F. SUTHERLAND | DAVID SVOBODA

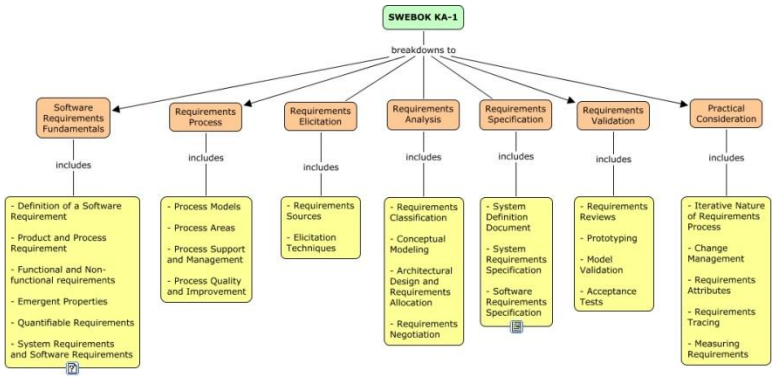

Secure Coding in C and C++
Robert C. Seacord
Foreword by Richard D. Pethia
CERT Director


Building
Secure Software
How to Avoid Security
Problems the Right Way
John Viega
Gary McGraw
Foreword by Bruce Schneier
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES


Addison-Wesley Software Security Series
SOFTWARE
SECURITY
BUILDING SECURITY IN
GARY McGRAW
Foreword by Dan Geer


Guide to the Softw are Engineering Body of Knowledge
(Version 0.95)
Software Requirements
Requirement Engineering Process
Requirements Elicitation
Requirement Analysis
Requirements Specification
Requirements Validation
Requirements Management
Software Design
Software Design Basic Concepts
Key Issues in Software Design
Software Structure and Architecture
Software Design Quality Analysis and Evaluation
Software Design Notations
Software Design Strategies and Methods
Software Construction
Reduction in Complexity
Linguistic Construction Methods
Formal Construction Methods
Visual Construction Methods
Anticipation of Diversity
Linguistic Construction Methods
Formal Construction Methods
Visual Construction Methods
Structuring for Validation
Linguistic Construction Methods
Formal Construction Methods
Visual Construction Methods
Use of External Standards
Linguistic Construction Methods
Formal Construction Methods
Visual Construction Methods
Software Testing
Testing Basic Concepts and Definitions
Test Levels
Test Techniques
Test-Related Measures
Managing the Test Process
Software Maintenance
Basic Concepts
Maintenance Process
Key Issues in Software Maintenance
Techniques for Maintenance
Software Configuration Management
Management of the SCM Process
Software Configuration Identification
Software Configuration Control
Software Configuration Status Accounting
Software Configuration Auditing
Software Release Management and Delivery
Software Engineering Management
Organizational Management
Process/Project Management
Software Engineering Measurement
Software Engineering Process
Software Engineering Process Concepts
Process Infrastructure
Process Measurement
Process Definition
Qualitative Process Analysis
Process Implementation and Change
Software Engineering Tools and Methods
Software Tools
Software Requirements Tools
Software Design Tools
Software Construction Tools
Software Testing Tools
Software Maintenance Tools
Software Engineering Process Tools
Software Quality Tools
Software Configuration Management Tools
Software Engineering Management Tools
Infrastructure Support Tools
Miscellaneous Tool Issues
Software Methods
Heuristic Methods
Formal Methods
Prototyping Methods
Miscellaneous Method Issues
Software Quality
Software Quality Concepts
Definition & Planning for Quality
Techniques Requiring Two or More People
Support to Other Techniques
Testing Special to SQA or V&V
Defect Finding Techniques
Measurement in Software Quality Analysis


Guide to the Softw are Engineering Body of Knowledge
(Version 0.95)
Software Requirements
Requirement Engineering Process
Requirements Elicitation
Requirement Analysis
Requirements Specification
Requirements Validation
Requirements Management
Software Design
Software Design Basic Concepts
Key Issues in Software Design
Software Structure and Architecture
Software Design Quality Analysis and Evaluation
Software Design Notations
Software Design Strategies and Methods
Software Construction
Reduction in Complexity
Linguistic Construction Methods
Formal Construction Methods
Visual Construction Methods
Anticipation of Diversity
Linguistic Construction Methods
Formal Construction Methods
Visual Construction Methods
Structuring for Validation
Linguistic Construction Methods
Formal Construction Methods
Visual Construction Methods
Use of External Standards
Linguistic Construction Methods
Formal Construction Methods
Visual Construction Methods
Software Testing
Testing Basic Concepts and Definitions
Test Levels
Test Techniques
Test-Related Measures
Managing the Test Process
Software Maintenance
Basic Concepts
Maintenance Process
Key Issues in Software Maintenance
Techniques for Maintenance
Software Configuration Management
Management of the SCM Process
Software Configuration Identification
Software Configuration Control
Software Configuration Status Accounting
Software Configuration Auditing
Software Release Management and Delivery
Software Engineering Management
Organizational Management
Process/Project Management
Software Engineering Measurement
Software Engineering Process
Software Engineering Process Concepts
Process Infrastructure
Process Measurement
Process Definition
Qualitative Process Analysis
Process Implementation and Change
Software Engineering Tools and Methods
Software Tools
Software Requirements Tools
Software Design Tools
Software Construction Tools
Software Testing Tools
Software Maintenance Tools
Software Engineering Process Tools
Software Quality Tools
Software Configuration Management Tools
Software Engineering Management Tools
Infrastructure Support Tools
Miscellaneous Tool Issues
Software Methods
Heuristic Methods
Formal Methods
Prototyping Methods
Miscellaneous Method Issues
Software Quality
Software Quality Concepts
Definition & Planning for Quality
Techniques Requiring Two or More People
Support to Other Techniques
Testing Special to SQA or V&V
Defect Finding Techniques
Measurement in Software Quality Analysis


SWEBOK KA-1
breakdowns to
Software Requirements Fundamentals
Requirements Process
Requirements Elicitation
Requirements Analysis
Requirements Specification
Requirements Validation
Practical Consideration
includes
- Definition of a Software Requirement
- Product and Process Requirement
- Functional and Non-functional requirements
- Emergent Properties
- Quantifiable Requirements
- System Requirements and Software Requirements
- Process Models
- Process Areas
- Process Support and Management
- Process Quality and Improvement
- Requirements Sources
- Elicitation Techniques
- Requirements Classification
- Conceptual Modeling
- Architectural Design and Requirements Allocation
- Requirements Negotiation
- System Definition Document
- System Requirements Specification
- Software Requirements Specification
- Requirements Reviews
- Prototyping
- Model Validation
- Acceptance Tests
- Iterative Nature of Requirements Process
- Change Management
- Requirements Attributes
- Requirements Tracing
- Measuring Requirements

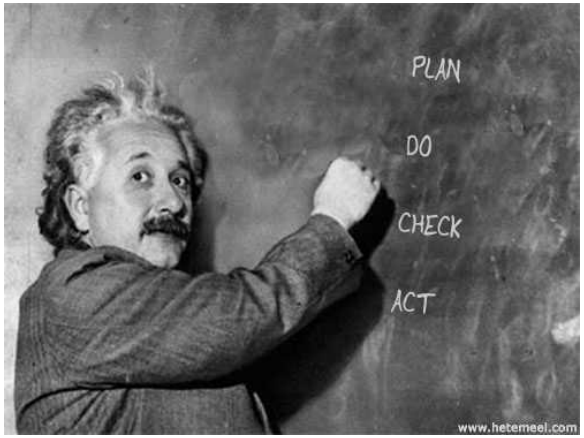
ICSQ
International Conference
on Software Quality

Institute
for Software
Excellence

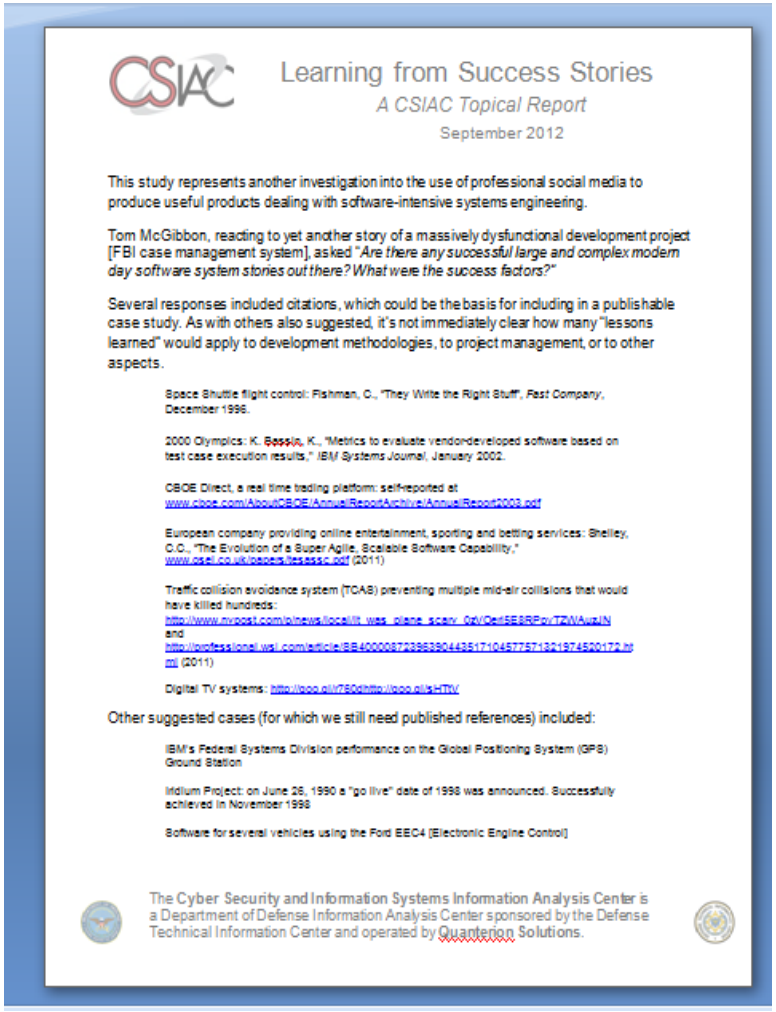
*C*yber *S*ecurity and *I*nformation Systems **I**nformation **A**nalysis **C**enter

Community of Practice → Practical Products

discussion



SMEs



CSIAC

Learning from Success Stories

A CSIAC Topical Report

September 2012

This study represents another investigation into the use of professional social media to produce useful products dealing with software-intensive systems engineering.

Tom McGibbon, reacting to yet another story of a massively dysfunctional development project [FBI case management system], asked "*Are there any successful large and complex modern day software system stories out there? What were the success factors?*"

Several responses included citations, which could be the basis for including in a publishable case study. As with others also suggested, it's not immediately clear how many "lessons learned" would apply to development methodologies, to project management, or to other aspects.

Space Shuttle flight control: Fishman, C., "They Write the Right Stuff", *Fast Company*, December 1996.

2000 Olympics: K. Bassin, K., "Metrics to evaluate vendor-developed software based on test case execution results," *IBM Systems Journal*, January 2002.

CBOE Direct, a real time trading platform: self-reported at www.cboe.com/AboutCBOE/AnnualReportArchive/AnnualReport2003.pdf

European company providing online entertainment, sporting and betting services: Shelley, C.C., "The Evolution of a Super Agile, Scalable Software Capability," www.osel.co.uk/papers/Sessssc.pdf (2011)

Traffic collision avoidance system (TCAS) preventing multiple mid-air collisions that would have killed hundreds: http://www.nypost.com/p/news/local/it_was_plane_scary_QkVQerf5E8RPpvTZWAuxJN and http://professional.wsj.com/article/SB40000872396390443517104577571321974520172.html (2011)

Digital TV systems: http://goo.gl/i760d http://goo.gl/sHTtV

Other suggested cases (for which we still need published references) included:

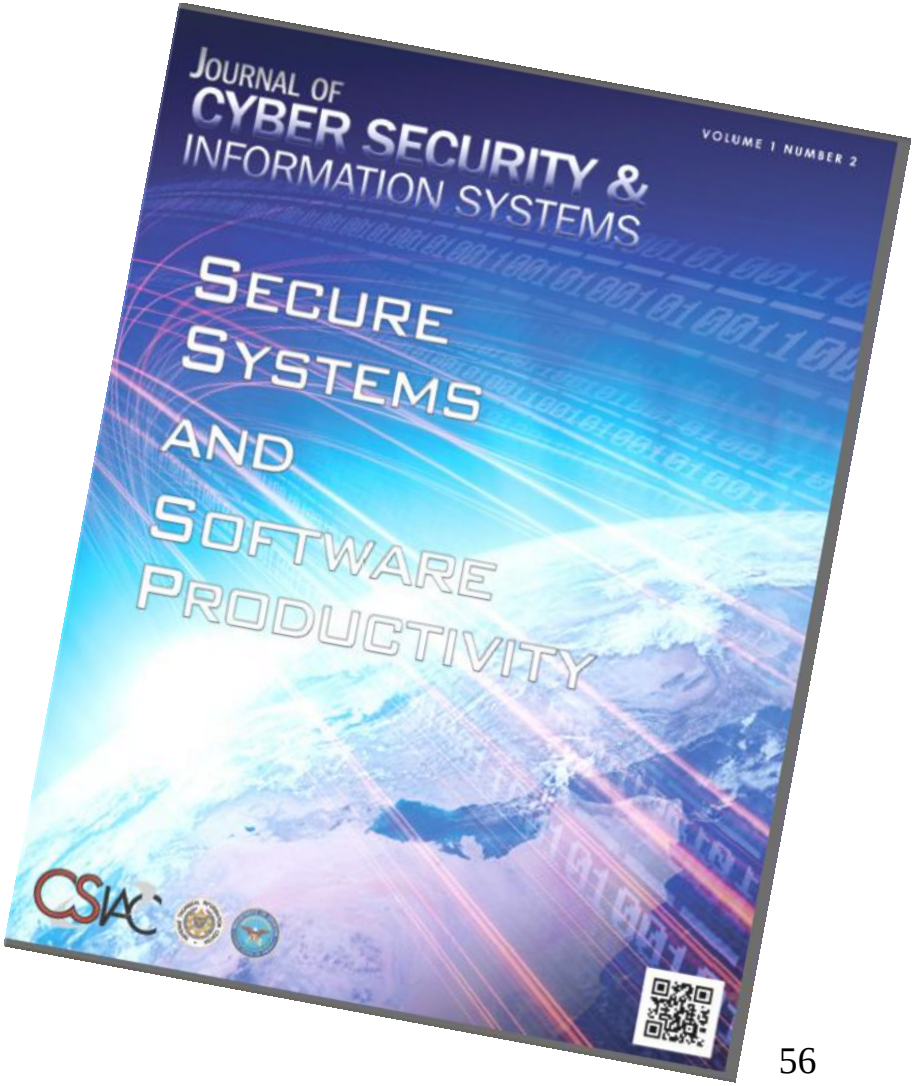
IBM's Federal Systems Division performance on the Global Positioning System (GPS) Ground Station

Iridium Project: on June 26, 1990 a "go live" date of 1998 was announced. Successfully achieved in November 1998

Software for several vehicles using the Ford EEC4 [Electronic Engine Control]

The Cyber Security and Information Systems Information Analysis Center is a Department of Defense Information Analysis Center sponsored by the Defense Technical Information Center and operated by Quanterion Solutions.


VOLUME 1 NUMBER 1
JOURNAL OF
CYBER SECURITY &
INFORMATION SYSTEMS
Welcome to the
CSIAC


JOURNAL OF
CYBER SECURITY &
INFORMATION SYSTEMS
VOLUME 1 NUMBER 2
SECURE
SYSTEMS
AND
SOFTWARE
PRODUCTIVITY


SOAR
Software Security Assurance
IATAC
DACS
CSIAC
Cyber Security & Information Systems Information Analysis Center
Design, Develop, Integrate
Develop Design
Acquire/ Build
Test & Integrate Solution
Determine Needs
Assurance Case
Planning & Requirements
Field Incremental Capability
Understand Problem
Operation & Maintenance
Assess Operational Security
O&M
Building Software Security Assurance into Systems: Recent Advances
A CSIAC State of the Art Report

follow-up [virtual] sessions

initial class session

management-sponsored project

Taz Daughtrey

hdaughtrey@quanterion.com

Cyber Security & Information Systems
Information Analysis Center

434 841 5444