Software Productivity Research
an Artemis company

SOFTWARE QUALITY IN 2002: A SURVEY OF THE STATE OF THE ART

Capers Jones, Chief Scientist Emeritus

Six Lincoln Knoll Lane
Burlington, Massachusetts 01803
http://www.SPR.com

July 23, 2002

SOURCES OF SPR'S QUALITY DATA

SPR clients from 1984 through 2002

- **About 600 companies (150 clients in Fortune 500 set)**
- **About 30 government/military groups**
- **About 12,000 total projects**
- **New data = about 75 projects per month**
- **Data collected from 24 countries**
- **Observations during more than a dozen lawsuits**

NEW LESSONS FOR SOFTWARE QUALITY IN 2002

QUALITY LESSONS FROM THE INTERNET ERA

Businesses are tightly coupled in "supply chains."

Poor quality in one company can affect scores of companies.

Poor quality drives away clients and loses business.

Poor quality can lead to expensive litigation.

Quality and security are becoming intertwined.

Web-based "content" is a special case (i.e. graphics, sounds)

BASIC DEFINITIONS

SOFTWARE QUALITY	**Software that combines the characteristics of low defect rates and high user satisfaction**
USER SATISFACTION	**Clients who are pleased with a vendor's products, quality levels, ease of use, and support**
DEFECT PREVENTION	**Technologies that minimize the risk of making errors in software deliverables**
DEFECT REMOVAL	**Activities that find and correct defects in software deliverables**
BAD FIXES	**Secondary defects injected as a byproduct of defect repairs**

FUNDAMENTAL SOFTWARE QUALITY METRICS

- **Defect Potentials**
  - requirements errors, design errors, code errors, document errors, bad fix errors, test plan errors, and test case errors
- **Defects Removed**
  - by origin of defects
  - before testing
  - during testing
  - during deployment
- **Defect Removal Efficiency**
  - ratio of development defects to customer defects
- **Defect Severity Levels (Valid defects)**
  - fatal, serious, minor, cosmetic

FUNDAMENTAL SOFTWARE QUALITY METRICS (cont.)

- **Duplicate Defects**
- **Invalid Defects**
- **Defect Removal Effort and Costs**
  - preparation
  - execution
  - repairs and rework
  - effort on duplicates and invalids
- **Supplemental Quality Metrics**
  - complexity
  - test case volumes
  - test case coverage
  - IBM's orthogonal defect categories

FUNDAMENTAL SOFTWARE QUALITY METRICS (cont.)

- **Standard Cost of Quality**
  - Prevention
  - Appraisal
  - Failures
- **Revised Software Cost of Quality**
  - Defect Prevention
  - Non-Test Defect Removal
  - Testing Defect Removal
  - Post-Release Defect Removal
- **Error-Prone Module Effort**
  - Identification
  - Removal or redevelopment
  - Repairs and rework

HAZARDOUS QUALITY DEFINITIONS

Should *quality* mean "conformance to requirements?"

Requirements contain > 15% of software errors.

Requirements grow at > 2% per month.

Do you conform to requirements errors?

Do you conform to totally new requirements?

Whose requirements are you trying to satisfy?

HAZARDOUS QUALITY METRICS

Cost per Defect

- Approaches infinity as defects near zero
- Conceals real economic value of quality

COST PER DEFECT PENALIZES QUALITY

	Ⓐ Poor Quality	Ⓑ Good Quality	Ⓒ Excellent Quality	Ⓓ Zero Defects
Function Points	100	100	100	100
Bugs Discovered	500	50	5	0
Preparation	$5,000	$5,000	$5,000	$5,000
Removal	$5,000	$2,500	$1,000	$ 0
Repairs	$25,000	$5,000	$1,000	$ 0
Total	$35,000	$12,500	$7,000	$5,000
Cost per Defect Removed	$70	$250	$1,400	∞
Cost per Function Point	$350	$125	$70	$50

HAZARDS OF "DEFECTS PER KLOC" METRICS

Defects per KLOC

When defects are found in multiple deliverables, it is invalid to assign all defects to a single item.

Software defects are found in:

- Requirements
- Design
- Source code
- User documents
- Bad fixes (secondary defects)

Requirements and design defects outnumber code defects.

Defects per KLOC metrics make major sources of software defects invisible.

FOUR LANGUAGE COMPARISON OF SOFTWARE DEFECT POTENTIALS

Defect Origin	Assembly	Ada	C ++	C++ and Reuse
Function points	100	100	100	100
KLOC	30	7.5	5.5	2.5
Requirements	20	20	20	20
Design	50	50	35	15
Code	150	45	35	15
Documents	25	25	25	25
Bad Fixes	20	10	7	4
TOTAL DEFECTS	265	150	122	79
Defects per KLOC	10.6	20.0	22.2	31.6
Defects/Function Point	3.0	2.0	1.22	0.79

Defect per KLOC may be considered to be professional malpractice.

U.S. AVERAGES FOR SOFTWARE QUALITY

(Data expressed in terms of defects per function point)

Defect Origins	Defect Potential	Removal Efficiency	Delivered Defects
Requirements	1.00	77%	0.23
Design	1.25	85%	0.19
Coding	1.75	95%	0.09
Documents	0.60	80%	0.12
Bad Fixes	0.40	70%	0.12
TOTAL	5.00	85%	0.75

(Function points show all defect sources - not just coding defects)

BEST IN CLASS SOFTWARE QUALITY

(Data expressed in terms of defects per function point)

Defect Origins	Defect Potential	Removal Efficiency	Delivered Defects
Requirements	0.40	85%	0.08
Design	0.60	97%	0.02
Coding	1.00	99%	0.01
Documents	0.40	98%	0.01
Bad Fixes	0.10	95%	0.01
TOTAL	2.50	96%	0.13

OBSERVATIONS

Most often found in systems software > SEI CMM Level 3

POOR SOFTWARE QUALITY - MALPRACTICE

(Data expressed in terms of defects per function point)

Defect Origins	Defect Potential	Removal Efficiency	Delivered Defects
Requirements	1.50	50%	0.75
Design	2.20	50%	1.10
Coding	2.50	80%	0.50
Documents	1.00	70%	0.30
Bad Fixes	0.80	50%	0.40
TOTAL	8.00	62%	3.05

OBSERVATIONS

Most often found in large client-server projects (> 5000 FP).

GOOD QUALITY RESULTS > 90% SUCCESS RATE

- Formal Inspections (Requirements, Design, and Code)
- Joint Application Design (JAD)
- Quality Function Deployment (QFD)
- Quality Metrics using function points
- Quality Metrics using IBM's Orthogonal classification
- Defect Removal Efficiency Measurements
- Automated Defect tracking tools
- Active Quality Assurance (> 5% SQA staff)
- Formal change controls
- User Satisfaction Surveys
- Formal Test Plans for Major Projects
- Quality Estimation Tools
- Automated Test Support Tools
- Testing Specialists
- Root-Cause Analysis

MIXED QUALITY RESULTS: < 50% SUCCESS RATE

- Total Quality Management (TQM)
- Independent Verification & Validation (IV & V)
- Independent quality audits
- Six-Sigma quality programs
- Baldrige Awards
- IEEE Quality Standards
- Testing only by Developers
- DOD 2167A and DOD 498
- Reliability Models
- Quality circles
- Clean-room methods
- Cost of quality without software modifications

POOR QUALITY RESULTS: < 25% SUCCESS RATE

- ISO 9000 - 9004 Quality Standards
- Informal Testing
- Manual Testing
- Passive Quality Assurance (< 3% QA staff)
- Token Quality Assurance (< 1% QA staff)
- LOC Metrics for quality
- Cost per defect metric
- Rapid Application Development (RAD)

A PRACTICAL DEFINITION OF SOFTWARE QUALITY (PREDICTABLE AND MEASURABLE)

- Low Defect Potentials (< 2.5 per Function Point)
- High Defect Removal Efficiency (> 95%)
- Unambiguous, Stable Requirements (< 2.5% change)
- Explicit Requirements Achieved (> 97.5% achieved)
- High User Satisfaction Ratings (> 90% “excellent”)
  - Installation
  - Ease of learning
  - Ease of use
  - Functionality
  - Compatibility
  - Error handling
  - User information (screens, manuals, tutorials)
  - Customer support
  - Defect repairs

SOFTWARE QUALITY OBSERVATIONS

Quality Measurements Have Found:

- Individual programmers -- Less than 50% efficient in finding bugs in their own software
- Normal test steps -- often less than 70% efficient (1 of 3 bugs remain)
- Design Reviews and Code Inspections -- often more than 65% efficient; have topped 85%
- Reviews or inspections plus formal testing -- are often more than 96% efficient; have hit 99%
- Reviews and Inspections -- lower costs and schedules by as much as 30%

SOFTWARE DEFECT ORIGINS

1) Requirements:	Hardest to prevent and repair
2) Design:	Most severe and pervasive
3) Code:	Most numerous; easiest to fix
4) Documentation:	Can be serious if ignored
5) Bad Fixes:	Very difficult to find
6) Bad Test Cases:	Common and troublesome
7) Data quality:	Common but hard to measure
8) Web content:	Unmeasured circa 2002

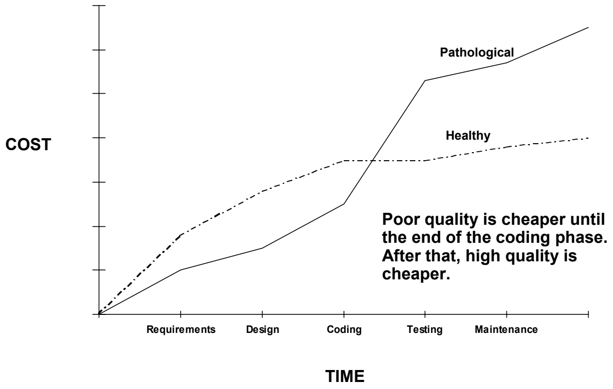
SOFTWARE DEFECT SEVERITY CATEGORIES

Severity 1:	TOTAL FAILURES	1% at release
Severity 2:	MAJOR PROBLEMS	20% at release
Severity 3:	MINOR PROBLEMS	35% at release
Severity 4:	COSMETIC ERRORS	44% at release
INVALID	USER OR SYSTEM ERRORS	15% of reports
DUPLICATE	MULTIPLE REPORTS	30% of reports
ABEYANT	CAN'T RECREATE ERROR	5% of reports

PERCENTAGE OF SOFTWARE EFFORT BY TASK

Size in Function Points	Mgt./ Support	Defect Removal	Paperwork	Coding	Total
10,240	18%	36%	34%	12%	100%
5,120	17%	33%	32%	18%	100%
2,580	16%	31%	29%	24%	100%
1,280	15%	29%	26%	30%	100%
640	14%	27%	23%	36%	100%
320	13%	25%	20%	42%	100%
160	12%	23%	17%	48%	100%
80	11%	21%	14%	54%	100%
40	10%	19%	11%	60%	100%
20	9%	17%	8%	66%	100%
10	8%	15%	5%	72%	100%

HOW QUALITY AFFECTS SOFTWARE COSTS



Poor quality is cheaper until the end of the coding phase. After that, high quality is cheaper.

U. S. SOFTWARE QUALITY AVERAGES CIRCA 2002

(Defects per Function Point)

	System Software	Commercial Software	Information Software	Military Software	Outsource Software
Defect Potentials	6.0	5.0	4.5	7.0	5.2
Defect Removal Efficiency	94%	90%	73%	96%	92%
Delivered Defects	0.4	0.5	1.2	0.3	0.4
First Year Discovery Rate	65%	70%	30%	75%	60%
First Year Reported Defects	0.26	0.35	0.36	0.23	0.30

U. S. SOFTWARE QUALITY AVERAGES CIRCA 2002

(Defects per Function Point)

	Web Software	Embedded Software	SEI-CMM 3 Software	SEI-CMM 1 Software	Overall Average
Defect Potentials	4.0	5.5	3.0	5.5	5.1
Defect Removal Efficiency	72%	95%	95%	73%	86.7%
Delivered Defects	1.1	0.3	0.15	1.5	0.68
First Year Discovery Rate	95%	90%	60%	35%	64.4%
First Year Reported Defects	1.0	0.27	0.09	0.52	0.43

SOFTWARE SIZE VS DEFECT REMOVAL EFFICIENCY

(Data Expressed in terms of Defects per Function Point)

Size	Defect Potential	Defect Removal Efficiency	Delivered Defects	1st Year Discovery Rate	1st Year Reported Defects
1	1.85	95.00%	0.09	90.00%	0.08
10	2.45	92.00%	0.20	80.00%	0.16
100	3.68	90.00%	0.37	70.00%	0.26
1000	5.00	85.00%	0.75	50.00%	0.38
10000	7.60	78.00%	1.67	40.00%	0.67
100000	9.55	75.00%	2.39	30.00%	0.72
AVERAGE	*5.02*	*85.83%*	*0.91*	*60.00%*	*0.38*

SOFTWARE DEFECT POTENTIALS AND DEFECT REMOVAL EFFICIENCY FOR EACH LEVEL OF SEI CMM

(Data Expressed in Terms of Defects per Function Point
For projects nominally 1000 function points in size)

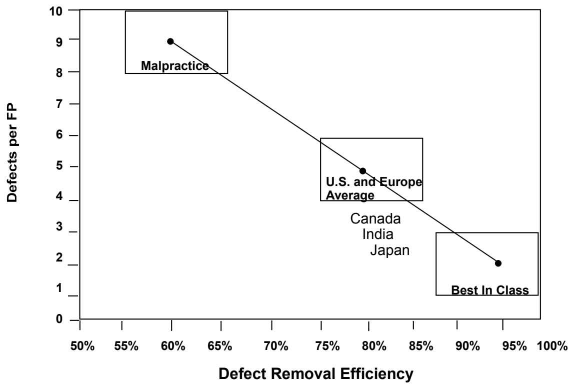
SEI CMM Levels	Defect Potentials	Removal Efficiency	Delivered Defects
SEI CMM 1	5.00	80%	1.00
SEI CMM 2	4.00	90%	0.40
SEI CMM 3	3.00	95%	0.15
SEI CMM 4	2.00	97%	0.08
SEI CMM 5	1.00	99%	0.01

SOFTWARE DEFECT POTENTIALS AND DEFECT REMOVAL EFFICIENCY FOR EACH LEVEL OF SEI CMM

(Data Expressed in Terms of Defects per Function Point
For projects > 5000 function points in size)

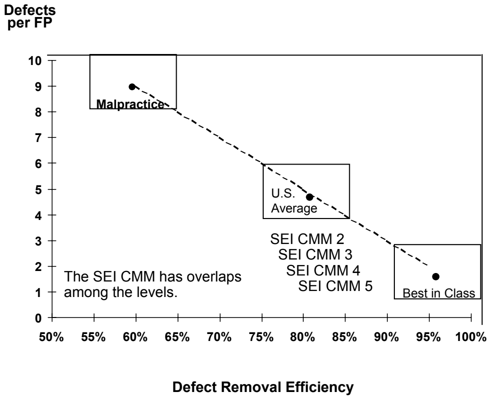
SEI CMM Levels	Defect Potentials	Removal Efficiency	Delivered Defects
SEI CMM 1	5.50	73%	1.48
SEI CMM 2	4.00	90%	0.40
SEI CMM 3	3.00	95%	0.15
SEI CMM 4	2.50	97%	0.08
SEI CMM 5	2.25	98%	0.05

MAJOR SOFTWARE QUALITY ZONES



Canada
India
Japan

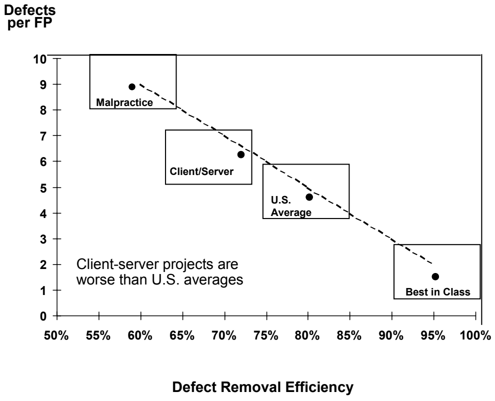
MAJOR SOFTWARE QUALITY ZONES



SEI CMM 2
SEI CMM 3
SEI CMM 4
SEI CMM 5

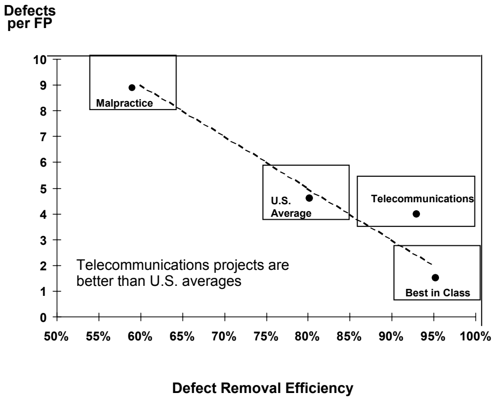
The SEI CMM has overlaps among the levels.

MAJOR SOFTWARE QUALITY ZONES



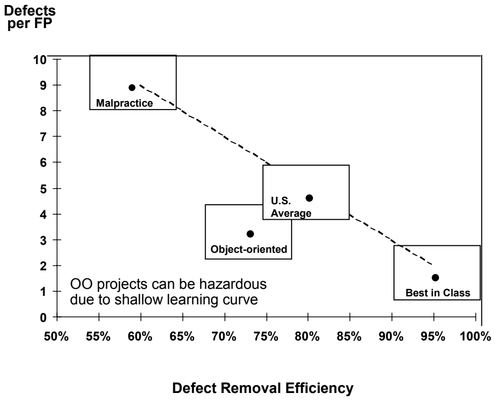
Client-server projects are worse than U.S. averages

SOFTWARE QUALITY IMPROVEMENT (cont.)



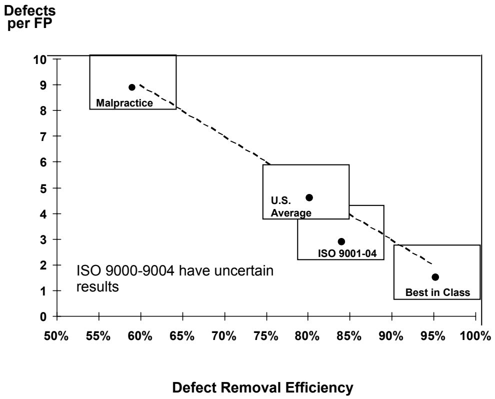
Telecommunications projects are better than U.S. averages

SOFTWARE QUALITY IMPROVEMENT (cont.)



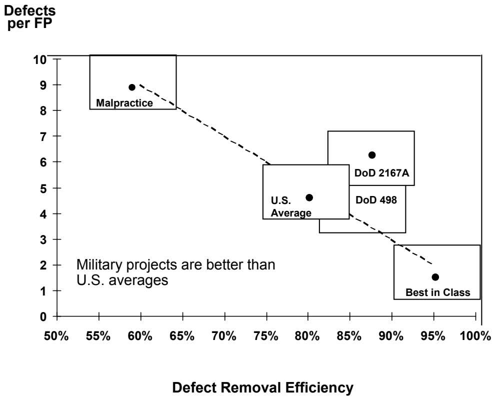
OO projects can be hazardous due to shallow learning curve

SOFTWARE QUALITY IMPROVEMENT (cont.)



ISO 9000-9004 have uncertain results

SOFTWARE QUALITY IMPROVEMENT (cont.)



Military projects are better than U.S. averages

INDUSTRY-WIDE DEFECT CAUSES

Ranked in order of effort required to fix the defects:

1. **Requirements problems (omissions; changes)**
2. **Design problems**
3. **Interface problems between modules**
4. **Logic, branching, and structural problems**
5. **Memory allocation problems**
6. **Testing omissions and poor coverage**
7. **Test case errors**
8. **Stress/performance problems**
9. **Bad fixes/Regressions**
10. **Documentation errors**

SOFTWARE QUALITY UNKNOWNS

SOFTWARE QUALITY TOPICS NEEDING RESEARCH:

- **ERRORS IN SOFTWARE TEST PLANS AND TEST CASES**
- **ERRORS IN WEB "CONTENT" (I.E. GRAPHICS, SOUNDS)**
- **MASS-UPDATE TESTING**
- **SUPPLY-CHAIN TESTING (MULTI-NATIONAL)**
- **ERRORS IN DATA BASES AND DATA WAREHOUSES**
- **CAUSES OF BAD FIX INJECTION RATES**
- **IMPACT OF COMPLEXITY ON QUALITY**
- **IMPACT OF CREEPING REQUIREMENTS**

DEFECT REMOVAL AND TESTING STAGES NOTED DURING LITIGATION FOR POOR QUALITY

	Reliable Software	Software Involved in Litigation for Poor Quality
Formal design inspections	Used	Not used
Formal code inspections	Used	Not used
Subroutine testing	Used	Used
Unit testing	Used	Used
New function testing	Used	Rushed or omitted
Regression testing	Used	Rushed or omitted
Integration testing	Used	Used
System testing	Used	Rushed or omitted
Performance testing	Used	Rushed or omitted
Capacity testing	Used	Rushed or omitted

SOFTWARE QUALITY AND LITIGATION CLAIMS

PLAINTIFF CLAIMS:	DEFENDANT CLAIMS:
Schedule overrun	Requirements changes
Cost overrun	New demands by clients
Poor quality	Rushed by clients
False claims	Refusal to cooperate

PROBLEMS ON BOTH SIDES

Ambiguous clauses in contract
Informal software cost estimates
No formal quality estimates at all
No use of formal inspections
Inadequate milestone tracking
Friction and severe personal disputes
Independent audits too late to solve issues

INDEPENDENT ASSESSMENTS AND AUDITS

- **Often used for military projects**
- **Can be an effective defense for litigation**
- **Effective quality assessments are formal**
- **Effective quality assessments cover defect prevention**
- **Effective quality assessments cover defect removal**
- **Effective quality assessments cover defect measures**
- **Effective assessments should cover 100% of projects**
- **Samples or partial assessments not safe for litigation**

OPTIMIZING QUALITY AND PRODUCTIVITY

Projects that achieve 95% cumulative Defect Removal Efficiency will find:

1) Minimum schedules

2) Maximum productivity

3) High levels of user satisfaction

4) Low levels of delivered defects

5) Low levels of maintenance costs

6) Low risk of litigation

ORIGINS OF SOFTWARE DEFECTS

Because defect removal is such a major cost element, studying defect origins is a valuable undertaking.

IBM Corporation (MVS)

45%	Design errors
25%	Coding errors
20%	Bad fixes
5%	Documentation errors
5%	Administrative errors
100%	

SPR Corporation (client studies)

20%	Requirements errors
30%	Design errors
35%	Coding errors
10%	Bad fixes
5%	Documentation errors
100%	

TRW Corporation

60%	Design errors
40%	Coding errors
100%	

Mitre Corporation

64%	Design errors
36%	Coding errors
100%	

Nippon Electric Corp.

60%	Design errors
40%	Coding errors
100%	

FUNCTION POINTS AND DEFECT POTENTIALS

Function points raised to the 1.15 power can predict the probable number of software defects. The range is from 1.1 to 1.25 power.

(Defects in requirements, design, code, documents, and bad fix categories.)

FUNCTION POINTS	POTENTIAL DEFECTS
1	1
10	14
100	200
1,000	2,818
10,000	39,811
100,000	316,228

SOFTWARE QUALITY AND PRODUCTIVITY

- **The most effective way of improving software productivity and shortening project schedules is to reduce defect levels.**
- **Defect reduction can occur through:**

1. Defect <u>prevention</u> technologies
  Structured design and JAD
  Structured code
  Reuse of certified components

2. Defect <u>removal</u> technologies
  Design inspections
  Code inspections
  Formal Testing

DEFECT PREVENTION METHODS

DEFECT PREVENTION

- **Joint Application Design (JAD)**
- **Quality function deployment (QFD)**
- **Software reuse (high-quality components)**
- **Root cause analysis**
- **Six-Sigma quality programs**
- **ISO 9000-9004 audits**
- **Climbing > Level 2 on the SEI CMM**
- **IBM “clean room” methods**

DEFECT PREVENTION - Continued

DEFECT PREVENTION

- **SEI CMM assessments**
- **SPR assessments**
- **TickIT assessments**
- **SPICE assessments**
- **Kaizen methodology**
- **Quality circles**
- **Independent Verification & Validation (IV&V)**

DEFECT PREVENTION - Continued

DEFECT PREVENTION

- **Total quality management (TQM)**
- **Quality measurements**
- **Orthogonal defect analysis**
- **Defect tracking tools**
- **Formal design inspections**
- **Formal code inspections**

DEFECT REMOVAL METHODS

DEFECT REMOVAL

- **Requirements inspections**
- **Design inspections**
- **Test plan inspections**
- **Test case inspections**
- **Code inspections**
- **User manual inspections**
- **Data quality inspections**

DEFECT REMOVAL - Continued

DEFECT REMOVAL

- **Independent audits**
- **Testing: normal forms**
- **Testing: special forms**
- **Testing: user-based forms**
- **Testing: independent**
- **Testing: clean-room**

DEFECT PREVENTION MATRIX

	Requirements Defects	Design Defects	Code Defects	Document Defects	Performance Defects
JAD's	Excellent	Good	Not Applicable	Fair	Poor
Prototypes	Excellent	Excellent	Fair	Not Applicable	Excellent
Structured Methods	Fair	Good	Excellent	Fair	Fair
ISO 9000-9004	Fair	Good	Fair	Fair	Fair
Blueprints & Reusable Code	Excellent	Excellent	Excellent	Excellent	Good
QFD	Good	Excellent	Fair	Poor	Good

DEFECT REMOVAL MATRIX

	Requirements Defects	Design Defects	Code Defects	Document Defects	Performance Defects
Reviews/ Inspections	Fair	Excellent	Excellent	Good	Fair
Prototypes	Good	Fair	Fair	Not Applicable	Good
Testing (all forms)	Poor	Poor	Good	Fair	Excellent
Correctness Proofs	Poor	Poor	Fair	Poor	Poor

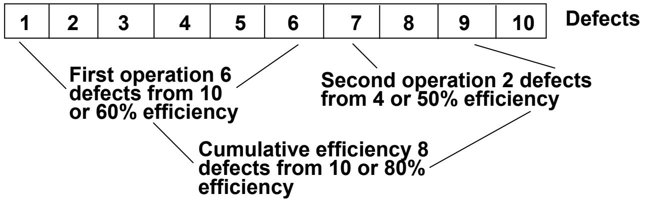
QUALITY MEASUREMENT EXCELLENCE

	Defect Estimation	Defect Tracking	Usability Measures	Complexity Measures	Test Coverage Measures	Removal Measures	Maintenance Measures
1. Excellent	Yes	Yes	Yes	Yes	Yes	Yes	Yes
2. Good	Yes	Yes	Yes	No	Yes	No	Yes
3. Average	No	Yes	Yes	No	Yes	No	Yes
4. Marginal	No	No	Yes	No	Yes	No	Yes
5. Poor	No	No	No	No	No	No	No

DEFECT REMOVAL EFFICIENCY

- Defect removal efficiency is a key quality measure
- Removal efficiency = $\dfrac{\text{Defects found}}{\text{Defects present}}$
- “Defects present” is the critical parameter

DEFECT REMOVAL EFFICIENCY - continued



1	2	3	4	5	6	7	8	9	10	Defects

First operation 6 defects from 10 or 60% efficiency

Second operation 2 defects from 4 or 50% efficiency

Cumulative efficiency 8 defects from 10 or 80% efficiency

Defect removal efficiency =	Percentage of defects removed by a single level of review, inspection or test
Cumulative defect removal efficiency =	Percentage of defects removed by a series of reviews, inspections or tests

---

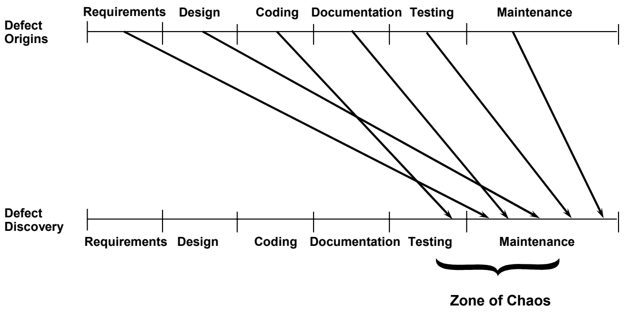
DEFECT REMOVAL EFFICIENCY EXAMPLE

DEVELOPMENT DEFECTS	
Inspections	500
Testing	400
Subtotal	900
USER-REPORTED DEFECTS IN FIRST 90 DAYS	
Valid unique defects	100
TOTAL DEFECT VOLUME	
Defect totals	1000
REMOVAL EFFICIENCY	
Dev. (900) / Total (1000) =	90%

RANGES OF DEFECT REMOVAL EFFICIENCY

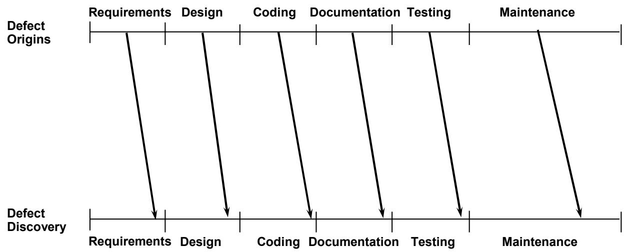
	Lowest	Median	Highest
1 Requirements review	20%	30%	50%
2 Top-level design reviews	30%	40%	60%
3 Detailed functional design reviews	30%	45%	65%
4 Detailed logic design reviews	35%	55%	75%
5 Code inspections	35%	60%	85%
6 Unit tests	10%	25%	50%
7 New Function tests	20%	35%	55%
8 Integration tests	25%	45%	60%
9 System test	25%	50%	65%
10 External Beta tests	15%	40%	75%
CUMULATIVE EFFICIENCY	75%	97%	99.99%

NORMAL DEFECT ORIGIN/DISCOVERY GAPS



Zone of Chaos

DEFECT ORIGINS/DISCOVERY WITH INSPECTIONS



SOFTWARE DEFECT REMOVAL RANGES

WORST CASE RANGE

TECHNOLOGY COMBINATIONS	DEFECT REMOVAL EFFICIENCY		
	Lowest	Median	Highest
1. No Design Inspections No Code Inspections No Quality Assurance No Formal Testing	30%	40%	50%

SOFTWARE DEFECT REMOVAL RANGES (cont.)

SINGLE TECHNOLOGY CHANGES

TECHNOLOGY COMBINATIONS	DEFECT REMOVAL EFFICIENCY Lowest	 Median	 Highest
2. No design inspections No code inspections FORMAL QUALITY ASSURANCE No formal testing	32%	45%	55%
3. No design inspections No code inspections No quality assurance FORMAL TESTING	37%	53%	60%
4. No design inspections FORMAL CODE INSPECTIONS No quality assurance No formal testing	43%	57%	65%
5. FORMAL DESIGN INSPECTIONS No code inspections No quality assurance No formal testing	45%	60%	68%

SOFTWARE DEFECT REMOVAL RANGES (cont.)

TWO TECHNOLOGY CHANGES

TECHNOLOGY COMBINATIONS	DEFECT REMOVAL EFFICIENCY Lowest	 Median	 Highest
6. No design inspections No code inspections FORMAL QUALITY ASSURANCE FORMAL TESTING	50%	65%	75%
7. No design inspections FORMAL CODE INSPECTIONS FORMAL QUALITY ASSURANCE No formal testing	53%	68%	78%
8. No design inspections FORMAL CODE INSPECTIONS No quality assurance FORMAL TESTING	55%	70%	80%

SOFTWARE DEFECT REMOVAL RANGES (cont.)

TWO TECHNOLOGY CHANGES - continued

TECHNOLOGY COMBINATIONS	DEFECT REMOVAL EFFICIENCY Lowest	 Median	 Highest
9. FORMAL DESIGN INSPECTIONS No code inspections FORMAL QUALITY ASSURANCE No formal testing	60%	75%	85%
10. FORMAL DESIGN INSPECTIONS No code inspections No quality assurance FORMAL TESTING	65%	80%	87%
11. FORMAL DESIGN INSPECTIONS FORMAL CODE INSPECTIONS No quality assurance No formal testing	70%	85%	90%

SOFTWARE DEFECT REMOVAL RANGES (cont.)

THREE TECHNOLOGY CHANGES

TECHNOLOGY COMBINATIONS	DEFECT REMOVAL EFFICIENCY Lowest	 Median	 Highest
12. No design inspections FORMAL CODE INSPECTIONS FORMAL QUALITY ASSURANCE FORMAL TESTING	75%	87%	93%
13. FORMAL DESIGN INSPECTIONS No code inspections FORMAL QUALITY ASSURANCE FORMAL TESTING	77%	90%	95%
14. FORMAL DESIGN INSPECTIONS FORMAL CODE INSPECTIONS FORMAL QUALITY ASSURANCE No formal testing	83%	95%	97%
15. FORMAL DESIGN INSPECTIONS FORMAL CODE INSPECTIONS No quality assurance FORMAL TESTING	85%	97%	99%

SOFTWARE DEFECT REMOVAL RANGES (cont.)

BEST CASE RANGE

TECHNOLOGY COMBINATIONS	DEFECT REMOVAL EFFICIENCY		
	Lowest	Median	Highest
1. FORMAL DESIGN INSPECTIONS FORMAL CODE INSPECTIONS FORMAL QUALITY ASSURANCE FORMAL TESTING	95%	99%	99.99%

DISTRIBUTION OF 1500 SOFTWARE PROJECTS BY DEFECT REMOVAL EFFICIENCY LEVEL

Defect Removal Efficiency Level (Percent)	Number of Projects	Percent of Projects
> 99	6	0.40%
95 - 99	104	6.93%
90 - 95	263	17.53%
85 - 90	559	37.26%
80 - 85	408	27.20%
< 80	161	10.73%
Total	1,500	100.00%

PATTERNS OF SOFTWARE QUALITY

SOFTWARE QUALITY METHODS VARY BY CLASS:

1) Systems software
2) Embedded software
3) Military software
4) Commercial software
5) Outsourced software
6) Information Technology (IT) software
7) End-User developed personal software
8) Web-based software

PATTERNS OF SOFTWARE QUALITY

SYSTEMS SOFTWARE QUALITY METHODS

- USUALLY > 96% DEFECT REMOVAL EFFICIENCY
- OVERALL, BEST SOFTWARE QUALITY RESULTS
- BEST QUALITY RESULTS > 10,000 FUNCTION POINTS
- FORMAL DESIGN AND CODE INSPECTIONS
- FORMAL SOFTWARE QUALITY ASSURANCE GROUPS
- FORMAL SOFTWARE QUALITY MEASUREMENTS
- FORMAL CHANGE CONTROL
- FORMAL TEST PLANS
- UNIT TEST BY DEVELOPERS
- 6 TO 10 TEST STAGES BY TEST SPECIALISTS
- USE OF SIX-SIGMA OR SEI METHODS

PATTERNS OF SOFTWARE QUALITY

EMBEDDED SOFTWARE QUALITY METHODS

- USUALLY > 94% DEFECT REMOVAL EFFICIENCY
- MOST PROJECTS < 500 FUNCTION POINTS IN SIZE
- WIDE RANGE OF SOFTWARE QUALITY RESULTS
- SHOULD USE FORMAL INSPECTIONS, BUT MAY NOT
- SHOULD USE FORMAL SQA TEAMS, BUT MAY NOT
- INFORMAL SOFTWARE QUALITY MEASUREMENTS
- SHOULD USE FORMAL CHANGE CONTROL
- SHOULD USE FORMAL TEST PLANS
- UNIT TEST BY DEVELOPERS
- 3 TO 6 TEST STAGES
- SHOULD USE TEST SPECIALISTS, BUT MAY NOT

PATTERNS OF SOFTWARE QUALITY

MILITARY SOFTWARE QUALITY METHODS

- USUALLY > 95% DEFECT REMOVAL EFFICIENCY
- OVERALL, GOOD SOFTWARE QUALITY RESULTS
- BEST QUALITY RESULTS > 100,000 FUNCTION POINTS
- FORMAL DESIGN AND CODE INSPECTIONS
- FORMAL SOFTWARE QUALITY ASSURANCE GROUPS
- FORMAL SOFTWARE QUALITY MEASUREMENTS
- FORMAL CHANGE CONTROL
- FORMAL TEST PLANS
- USE OF SEI ASSESSMENTS AND CMM APPROACHES
- 6 TO 15 TEST STAGES BY TEST SPECIALISTS
- ONLY CLASS TO USE INDEPENDENT VERIF. AND VALID.
- ONLY CLASS TO USE INDEPENDENT TESTING

PATTERNS OF SOFTWARE QUALITY

COMMERCIAL SOFTWARE QUALITY METHODS

- USUALLY > 90% DEFECT REMOVAL EFFICIENCY
- MOST PROJECTS > 5000 FUNCTION POINTS IN SIZE
- WIDE RANGE OF SOFTWARE QUALITY RESULTS
- SHOULD USE FORMAL INSPECTIONS, BUT MAY NOT
- SHOULD USE FORMAL SQA TEAMS, BUT MAY NOT
- INFORMAL SOFTWARE QUALITY MEASUREMENTS
- FORMAL CHANGE CONTROL
- FORMAL TEST PLANS
- UNIT TEST BY DEVELOPERS
- 3 TO 8 TEST STAGES
- SHOULD USE TEST SPECIALISTS, BUT MAY NOT
- OFTEN EXTENSIVE BETA TESTING BY USERS

PATTERNS OF SOFTWARE QUALITY

OUTSOURCE SOFTWARE QUALITY METHODS

- USUALLY > 94% DEFECT REMOVAL EFFICIENCY
- OVERALL, BETTER SOFTWARE QUALITY THAN CLIENTS
- GOOD QUALITY > 1000 FUNCTION POINTS
- SHOULD USE FORMAL INSPECTIONS, BUT MAY NOT
- SHOULD USE FORMAL SQA GROUPS, BUT MAY NOT
- SHOULD USE FORMAL QUALITY MEASUREMENTS
- SHOULD USE FORMAL CHANGE CONTROL
- SHOULD USE FORMAL TEST PLANS
- UNIT TEST BY DEVELOPERS
- 4 TO 8 TEST STAGES BY TEST SPECIALISTS
- ACCEPTANCE TESTING BY CLIENTS
- MANY LATE CHANGES DEMANDED BY CLIENTS

PATTERNS OF SOFTWARE QUALITY

IT SOFTWARE QUALITY METHODS

- USUALLY < 90% DEFECT REMOVAL EFFICIENCY
- OFTEN MEDIOCRE SOFTWARE QUALITY
- POOR QUALITY > 1000 FUNCTION POINTS
- SELDOM USES FORMAL DESIGN AND CODE INSPECTIONS
- SELDOM USES FORMAL SQA GROUPS
- SELDOM USES SOFTWARE QUALITY MEASUREMENTS
- FORMAL CHANGE CONTROL
- INFORMAL TEST PLANS
- UNIT TEST BY DEVELOPERS
- 2 TO 6 TEST STAGES BY DEVELOPERS
- ACCEPTANCE TESTING BY CLIENTS

PATTERNS OF SOFTWARE QUALITY

END-USER SOFTWARE QUALITY METHODS

- USUALLY < 50% DEFECT REMOVAL EFFICIENCY
- OFTEN DANGEROUSLY POOR SOFTWARE QUALITY
- ALL PROJECTS < 100 FUNCTION POINTS
- NO USE OF FORMAL DESIGN AND CODE INSPECTIONS
- NO USE OF SQA GROUPS
- NO USE OF SOFTWARE QUALITY MEASUREMENTS
- INFORMAL CHANGE CONTROL
- SELDOM ANY TEST PLANS
- UNIT TEST BY DEVELOPER MAY BE ONLY TEST STAGE

PATTERNS OF SOFTWARE QUALITY

WEB SOFTWARE QUALITY METHODS

- USUALLY < 90% DEFECT REMOVAL EFFICIENCY
- MOST PROJECTS < 1000 FUNCTION POINTS IN SIZE
- WIDE RANGE OF SOFTWARE QUALITY RESULTS
- SHOULD USE FORMAL INSPECTIONS, BUT MAY NOT
- WEB "CONTENT" IS A SPECIAL TOPIC
- INFORMAL SOFTWARE QUALITY MEASUREMENTS
- SHOULD USE FORMAL CHANGE CONTROL
- SHOULD USE FORMAL TEST PLANS
- UNIT TEST BY DEVELOPERS
- 2 TO 4 TEST STAGES
- SHOULD USE TEST SPECIALISTS, BUT MAY NOT

CONCLUSIONS ON SOFTWARE QUALITY

- No single method is adequate.
- Testing alone is insufficient.
- Formal inspections and tests combined give high efficiency, low costs and short schedules.
- Defect prevention plus inspections and tests give highest cumulative efficiency and best economics.
- Bad fix injection needs special solutions.
- Data Base errors need special solutions.
- Web "content" needs special solutions.