Team Software Process (TSP) In Context

Capers Jones, President

www.SPR.com
Capers.Jones3@Gmail.com

August 15, 2011

Dedications

This presentation is dedicated to:

- **Watts Humphrey, the developer of Team Software Process (TSP)**
- **Allan Albrecht, the developer of function point metrics**

Watts was an industry leader in achieving high quality software.

Al was an industry leader in developing metrics that could quantify high quality software.

Innovations Pioneered by Watts Humphrey

- **IBM process assessments** **1972**
- **IBM inspection methods** **1972**
- **SEI inspection methods** **1987**
- **Personal Software Process (PSP)** **1995**
- **Team Software Process (TSP}** **2000**

Innovations Pioneered by Allan Albrecht

- **IBM function point metrics (co inventor)** **1975**
- **IBM backfiring (LOC to function points)** **1977**
- **IBM function point training course** **1978**
- **Co-inventor of feature points** **1986**
- **First function point certification** **1987**

CALENDAR OF SOFTWARE DEVELOPMENT METHODS

• Waterfall development	1962
• Structured development	1975
• Object-Oriented development (OO)	1980
• Rapid Application Development (RAD)	1984
• Iterative development	1985
• Rational Unified Process (RUP)	1986
• Agile development	1997
• Personal Software Process (PSP)	2000
• Team Software Process (TSP)	2000

SIGNIFICANT SOFTWARE INNOVATIONS

• Software defect severity scale (IBM)	1956
• Automated change management tools	1967
• High-level programming languages	1969
• Software process assessments (IBM)	1970
• Structured coding	1971
• Design and code inspections (IBM)	1972
• Automated project management tools	1973
• Automated cost and quality estimation (IBM)	1974

SIGNIFICANT SOFTWARE INNOVATIONS

• Function point metrics (IBM)	1975
• Joint application design (JAD) (IBM)	1976
• Backfiring LOC to function points (IBM)	1977
• Software reusability	1979
• Commercial software estimating tools	1980
• Object-oriented programming	1981
• Complexity analysis tools	1985
• SEI capability maturing model (CMM/CMMI)	1985

SIGNIFICANT SOFTWARE INNOVATIONS

• Software development/maintenance workbenches	1986
• Test coverage analysis tools	1990
• Use cases for requirements	1994
• IBM Orthogonal defect classification	1995
• Commercial software benchmarks	1997
• Static analysis tools	1997
• Automated testing tools	1997
• Six-Sigma for Software	2000
• Launch of Wiki-based collaboration	2001

CALENDAR OF SIGNIFICANT SOFTWARE PROBLEMS

• Requirements < 50% complete	1966
• Requirements change > 2% per month	1975
• Requirements defects resist testing	1976
• Testing < 60% efficient in finding bugs	1978
• Bad fixes > 7% of all defect repairs	1979
• About 5% of modules contain > 50% of defects	1979
• About 35% of large projects are cancelled	1980
• Most estimates are excessively optimistic	1980
• Average defect removal <85% in U.S.	1980

SOFTWARE PROBLEMS HELPED BY TSP

• Requirements < 50% complete	1966
• Requirements change > 2% per month	1975
• Requirements defects resist testing	1976 **
• Testing < 60% efficient in finding bugs	1978 **
• Bad fixes > 7% of all defect repairs	1979 **
• About 5% of modules contain > 50% of defects	1979 **
• About 35% of large projects are cancelled	1980 **
• Most estimates are excessively optimistic	1980 **
• Average defect removal <85% in U.S.	1980 **

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)
(Code defects = 35% of total 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 or in TSP projects)

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 water fall projects > 10,000 Function Points).

DEFECT POTENTIALS AND 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 CMMI 1	5.25	80%	1.05
SEI CMMI 2	5.00	85%	0.75
SEI CMMI 3	4.75	90%	0.48
SEI CMMI 4	4.50	93%	0.32
SEI CMMI 5	4.25	96%	0.17

DEFECT POTENTIALS AND REMOVAL EFFICIENCY FOR EACH LEVEL OF SEI CMM

(Data Expressed in Terms of Defects per Function Point
For projects 10,000 function points in size)

SEI CMM Levels	Defect Potentials	Removal Efficiency	Delivered Defects
SEI CMMI 1	6.50	75%	1.63
SEI CMMI 2	6.25	82%	1.13
SEI CMMI 3	5.50	87%	0.71
SEI CMMI 4	5.25	90%	0.53
SEI CMMI 5	4.75	94%	0.29

DEFECTS AND SOFTWARE METHODOLGOIES

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

Software methods	Defect Potential	Removal Efficiency	Delivered Defects
Waterfall	5.50	80%	1.10
Iterative	4.75	87%	0.62
Object-Oriented	4.50	88%	0.54
Agile	4.00	90%	0.40
Rational Unified Process (RUP)	4.25	92%	0.34
PSP and TSP	3.50	96%	0.14
Hybrid with 85% certified reuse	1.75	99%	0.02

DEFECTS AND SOFTWARE METHODOLGOIES

(Data Expressed in Terms of Defects per Function Point
For projects nominally 10,000 function points in size)

Software methods	Defect Potential	Removal Efficiency	Delivered Defects
Waterfall	7.00	75%	1.75
Iterative	6.25	82%	1.13
Object-Oriented	5.75	85%	0.86
Agile	5.50	87%	0.72
Rational Unified Process (RUP)	5.50	90%	0.55
PSP and TSP	5.00	95%	0.25
Hybrid with 85% certified reuse	2.25	96%	0.09

QUALITY PREDICTION RULES OF THUMB

RAISE APPLICATION SIZE IN FUNCTION POINTS TO THESE POWERS TO PREDICT SOFTWARE DEFECT POTENTIALS

(Requirements, design, code, documents, and bad fixes)

Method	Power	Size	Defects	Defects per Function Point
Waterfall	1.24	1000	5,248	5.25
Agile	1.20	1000	3,981	3.98
RUP	1.21	1000	4,266	4.27
TSP	1.18	1000	3,467	3.47

MOVING TO EXCELLENCE IN SOFTWARE ENGINEERING

- **Start with an assessment and baseline to find out what is right and wrong with current practices.**
- **Commission a benchmark study to compare your performance with best practices in your industry**
- **Stop doing what is wrong.**
- **Do more of what is right.**
- **Set targets: *Best in Class* *****, Better than Average****, Better than Today***.**
- **Develop a three-year technology plan.**
- **Include: capital equipment, offices, tools, methods, education, culture, languages and return on investment (ROI).**

TECHNICAL REASONS FOR SOFTWARE FAILURES

Unsuccessful Projects	Successful Projects
Inappropriate methodologies	Optimal methodologies
No automated sizing tools	Automated sizing tools
No automated estimation tools	Automated estimation tools
No automated planning tools	Automated planning tools
No progress reporting	Accurate progress reporting
Inaccurate cost collection	Accurate cost collection
No measurement data	Substantial measurement data
Inaccurate metrics	Accurate metrics
No design reviews	Formal design reviews
No code inspections	Formal code inspections
No defect tracking	Formal defect tracking
Informal change control	Formal change control
Unstable requirements (>30%)	Stable requirements (< 10%)

SOCIAL REASONS FOR SOFTWARE FAILURES

Unsuccessful Projects

- Excessive schedule pressure
- Severe friction with clients
- Poor communications
- Divisive politics
- Naive senior executives
- Management malpractice
- Technical malpractice
- Untrained Generalists

Successful Projects

- Realistic schedule expectation
- Cooperation with clients
- Good communications
- Politics held in check
- Experienced senior executives
- Capable management
- Capable technical staff
- Trained Specialists

- Quality Assurance
- Testing
- Planning and Estimating

OTHER CORRELATIONS WITH SOFTWARE FAILURES

Intermittent Failure Factors

Geographic separation of team with inadequate communication

Multiple sub-contractors involved with inadequate communication

Extraordinary storage or timing constraints

Projects using "low bid" as sole contract criterion

Staffing build up > 15% per month

Staff attrition > 40% of project team

Abrupt introduction of new technologies

Projects by companies that are downsizing

New executives replace proven methods with latest fads

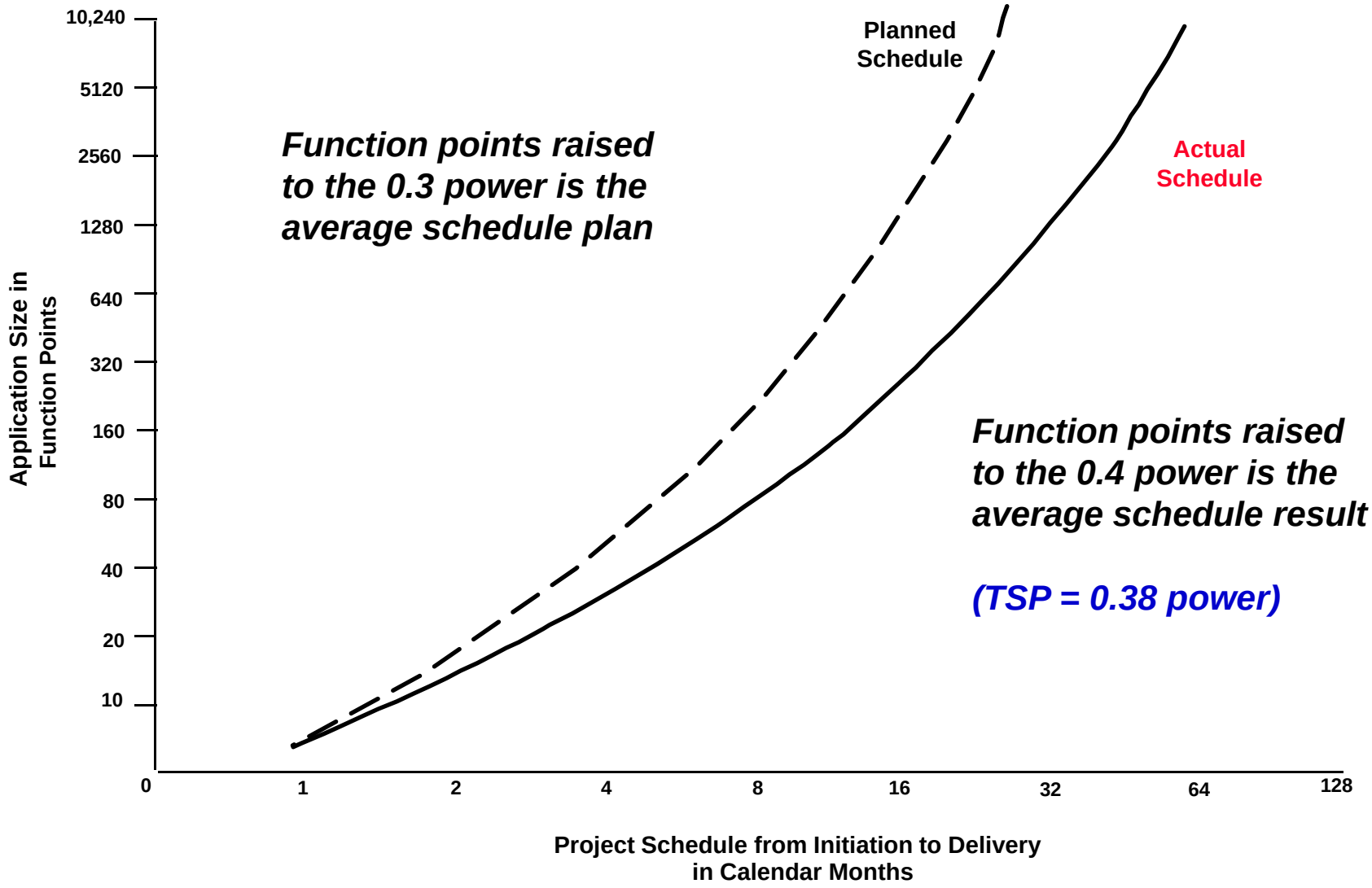
Trained personnel retire or change jobs

U.S. SOFTWARE PERFORMANCE LEVELS

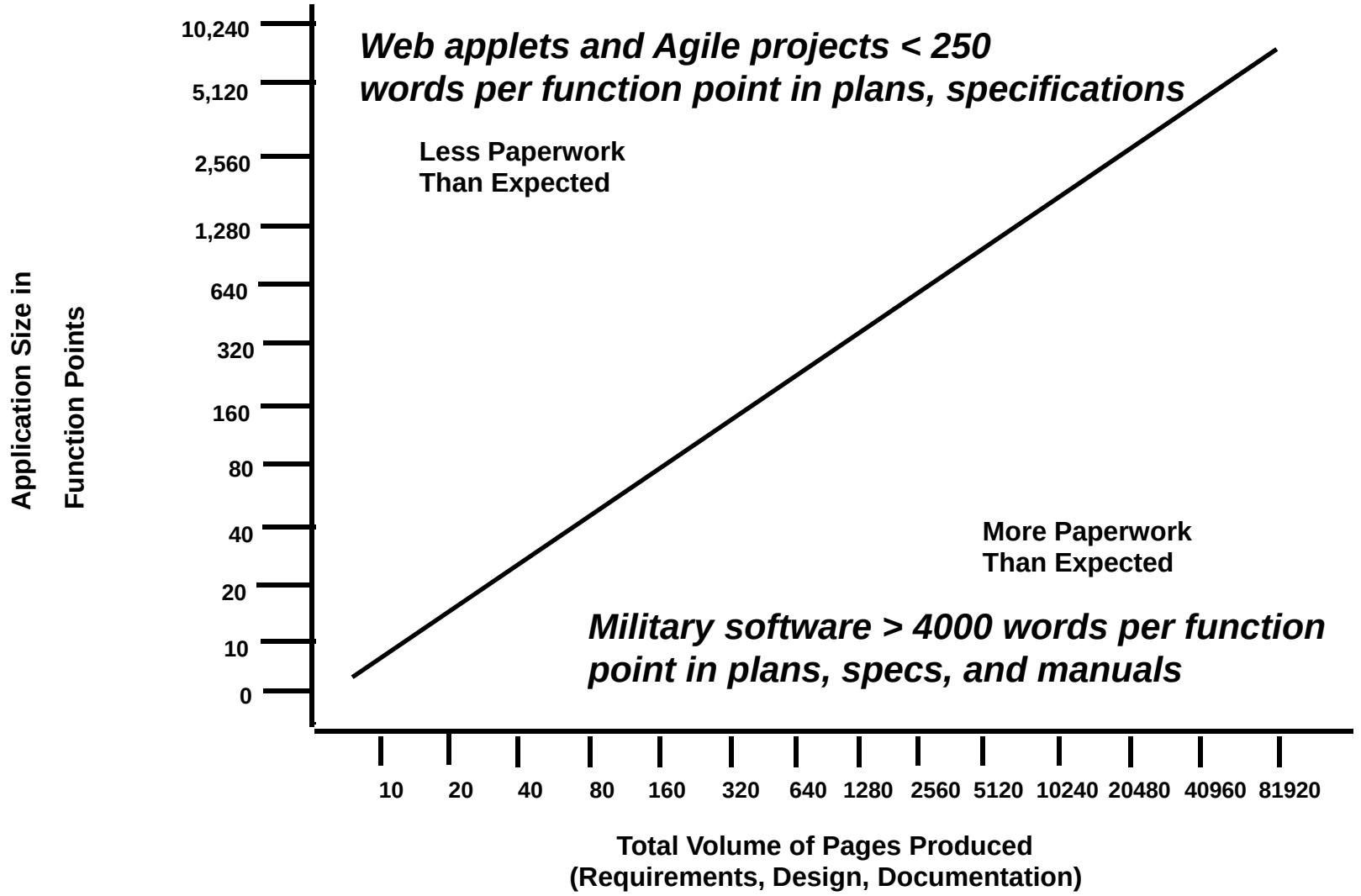
PROJECT MANAGEMENT		TECHNICAL STAFFS		SOFTWARE USERS	
Sizing	Fair	Requirements	Fair	Requirements	Poor
Estimating	Poor	Design	Good	Schedule Demands	Poor
Planning	Fair	Coding	Good	Reviews	Fair
Tracking	Poor	Reviews	Fair	Acceptance Test	Fair
Measuring	Poor	Testing	Good	Usage	Good
Overall	Poor		Good		Fair

Conclusion: **U. S. technical skills are better than U. S. management skills. Project management and quality are frequent problem areas.**

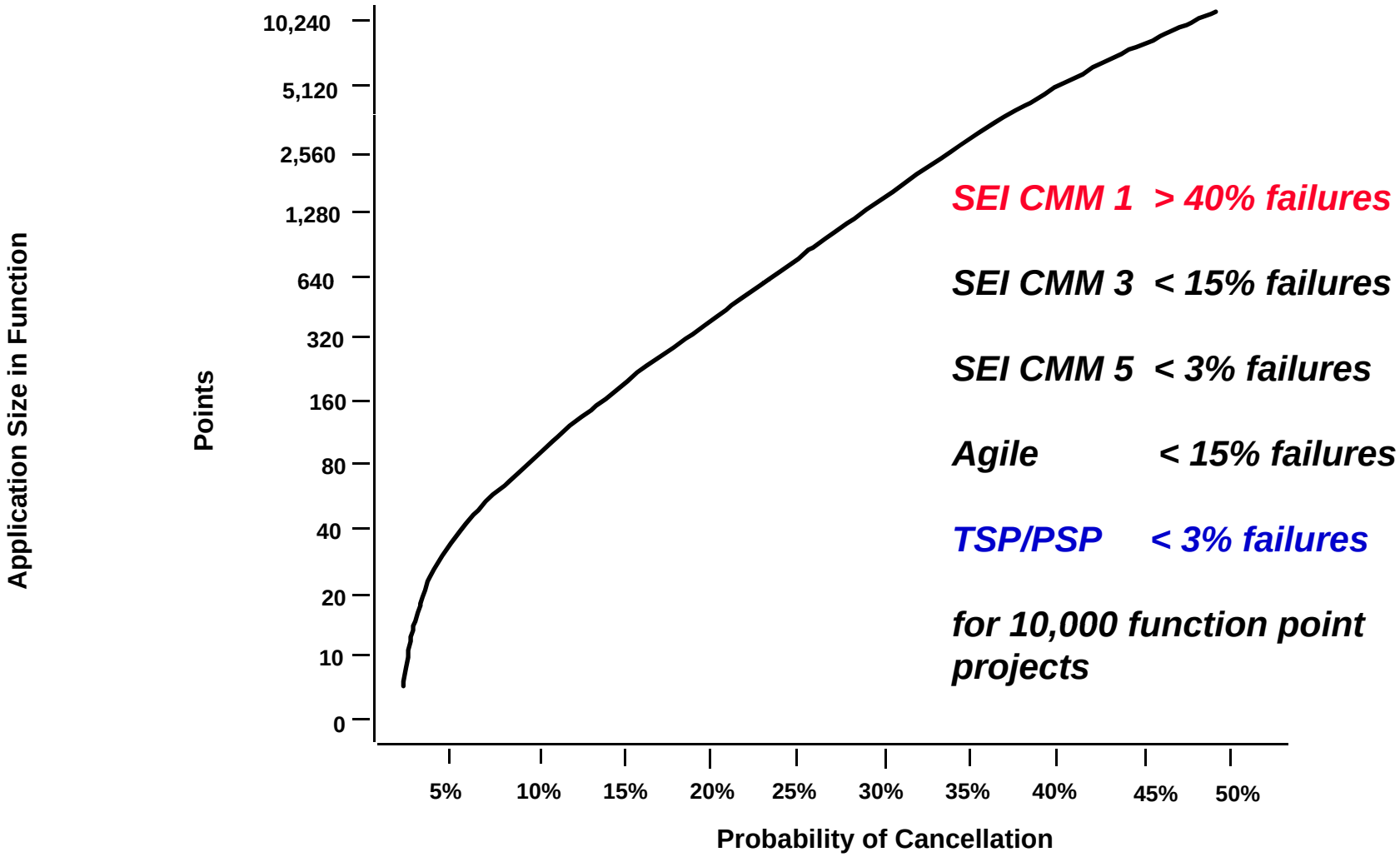
PLANNED VERSUS ACTUAL PROJECT SCHEDULES



SOFTWARE PAPERWORK



RISK OF PROJECT FAILURE

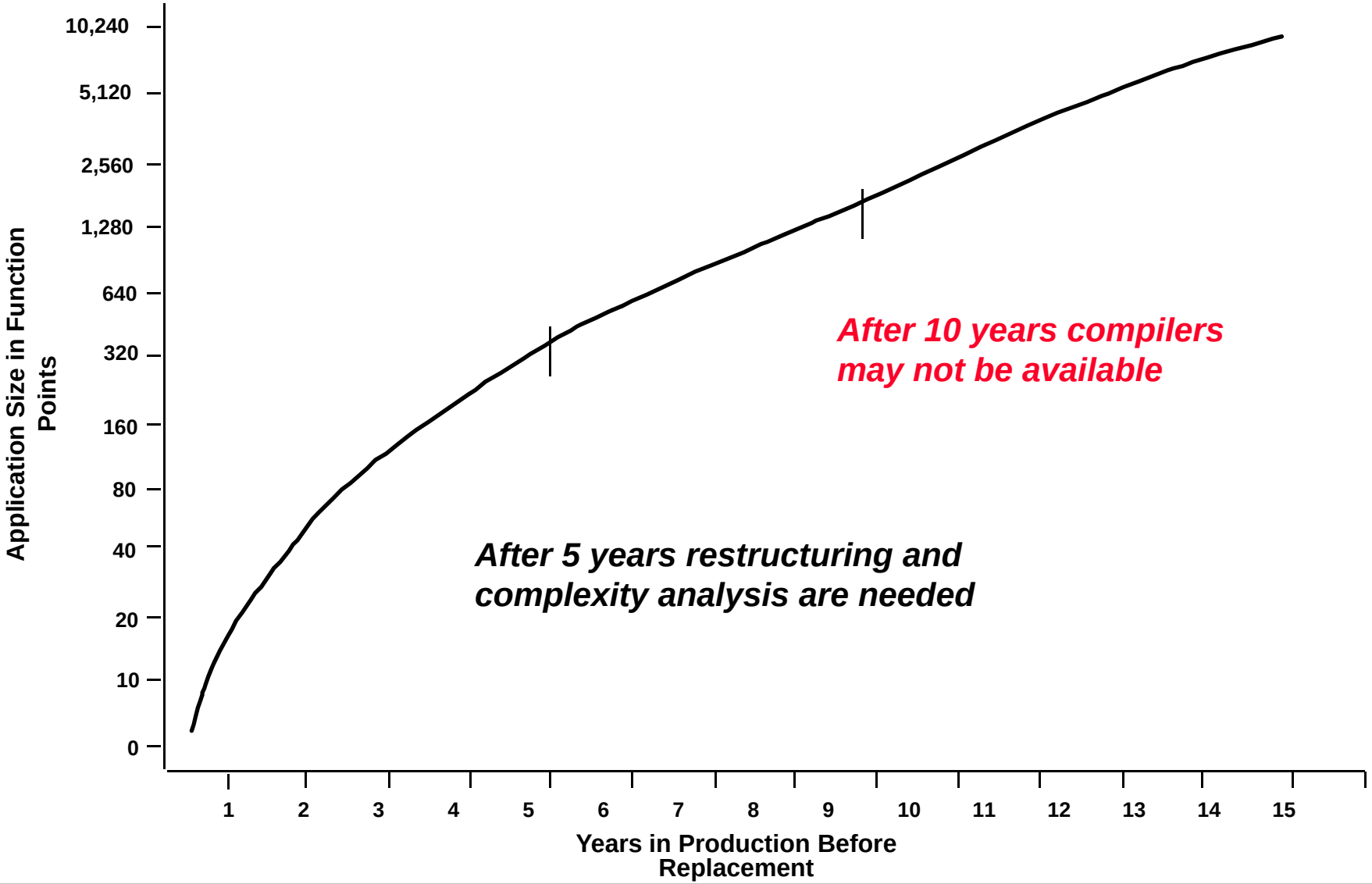


RISKS OF FAILURE OR DELAY BY CMM LEVEL

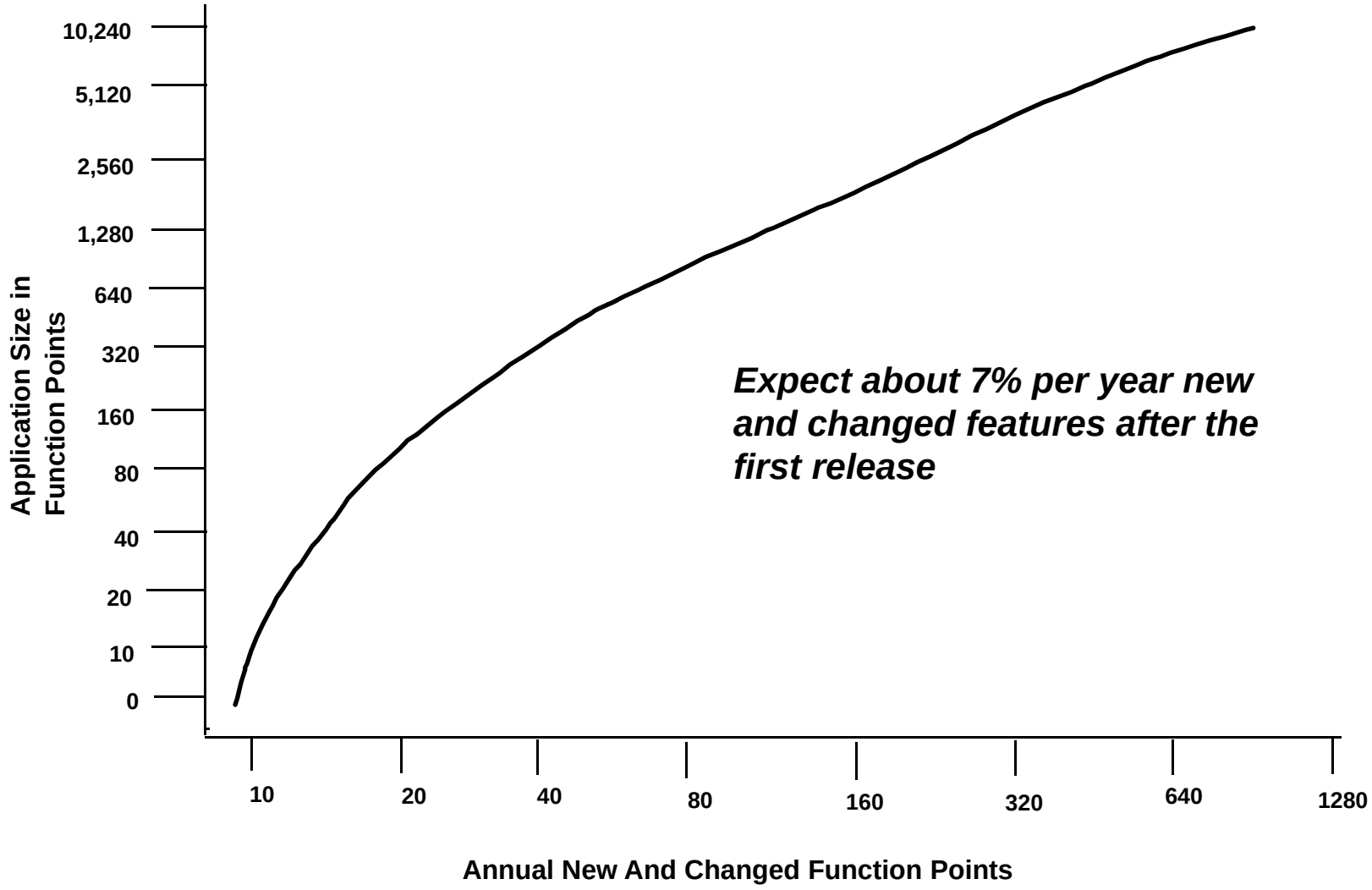
(Complex projects of 10,000 function points in size)

SEI CMM LEVEL	Delay > 1 year	Termination
SEI CMMI Level 1	35%	40%
SEI CMMI Level 2	30%	30%
SEI CMMI Level 3	20%	12%
SEI CMMI Level 4	12%	04%
SEI CMMI Level 5	08%	02%
SEI CMMI Level 5 + TSP	05%	02%

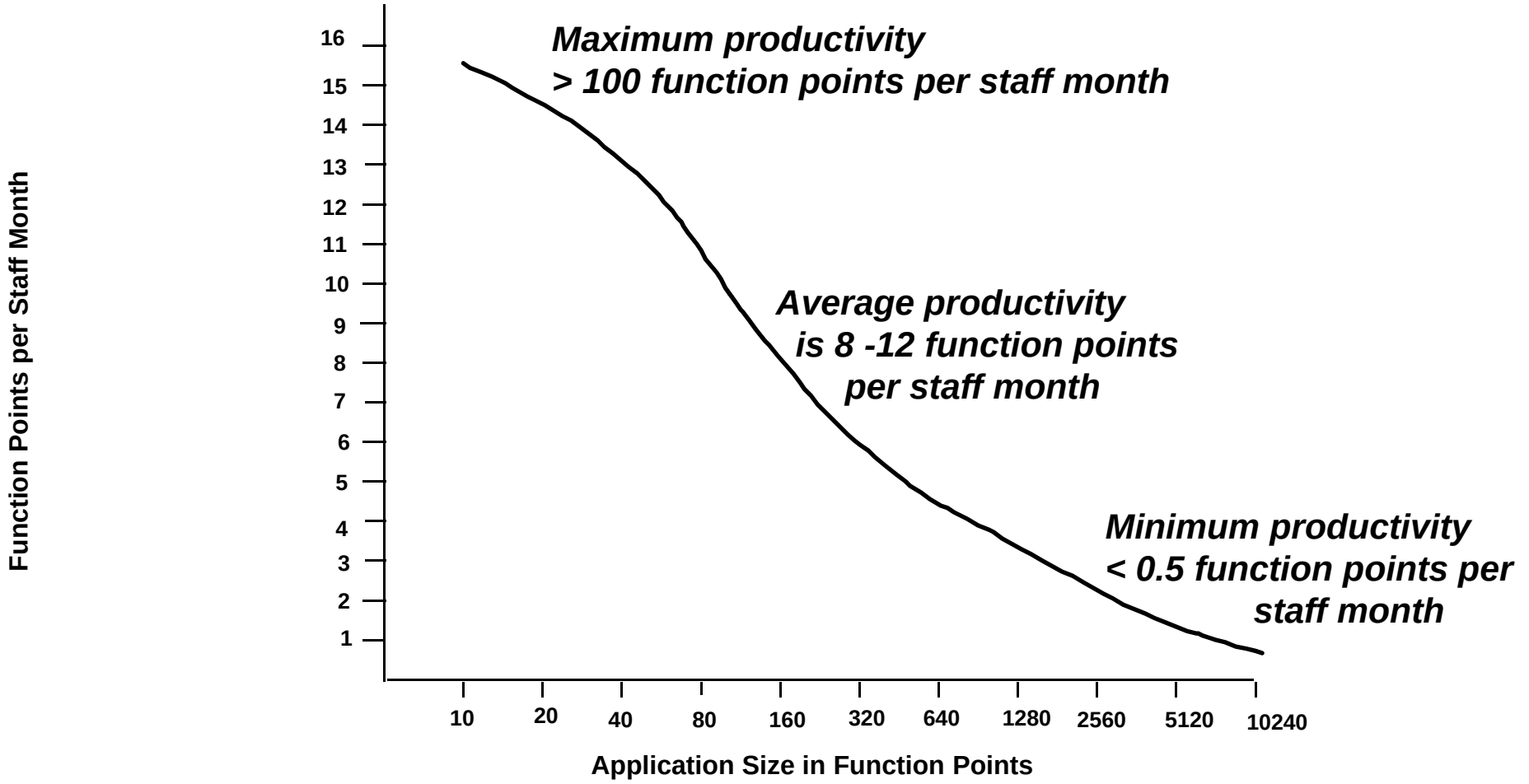
SOFTWARE LIFE EXPECTANCY



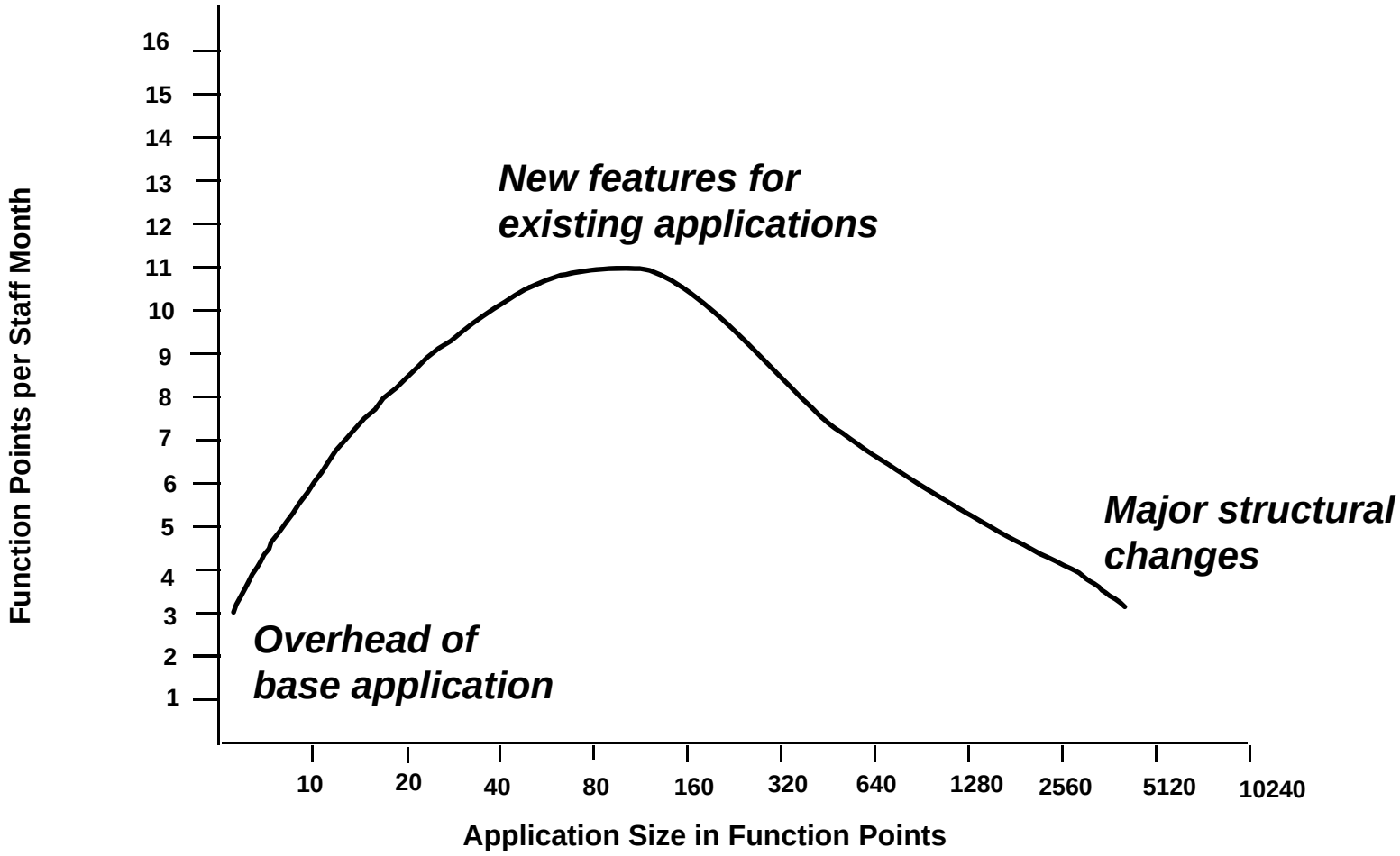
ANNUAL SOFTWARE ENHANCEMENTS



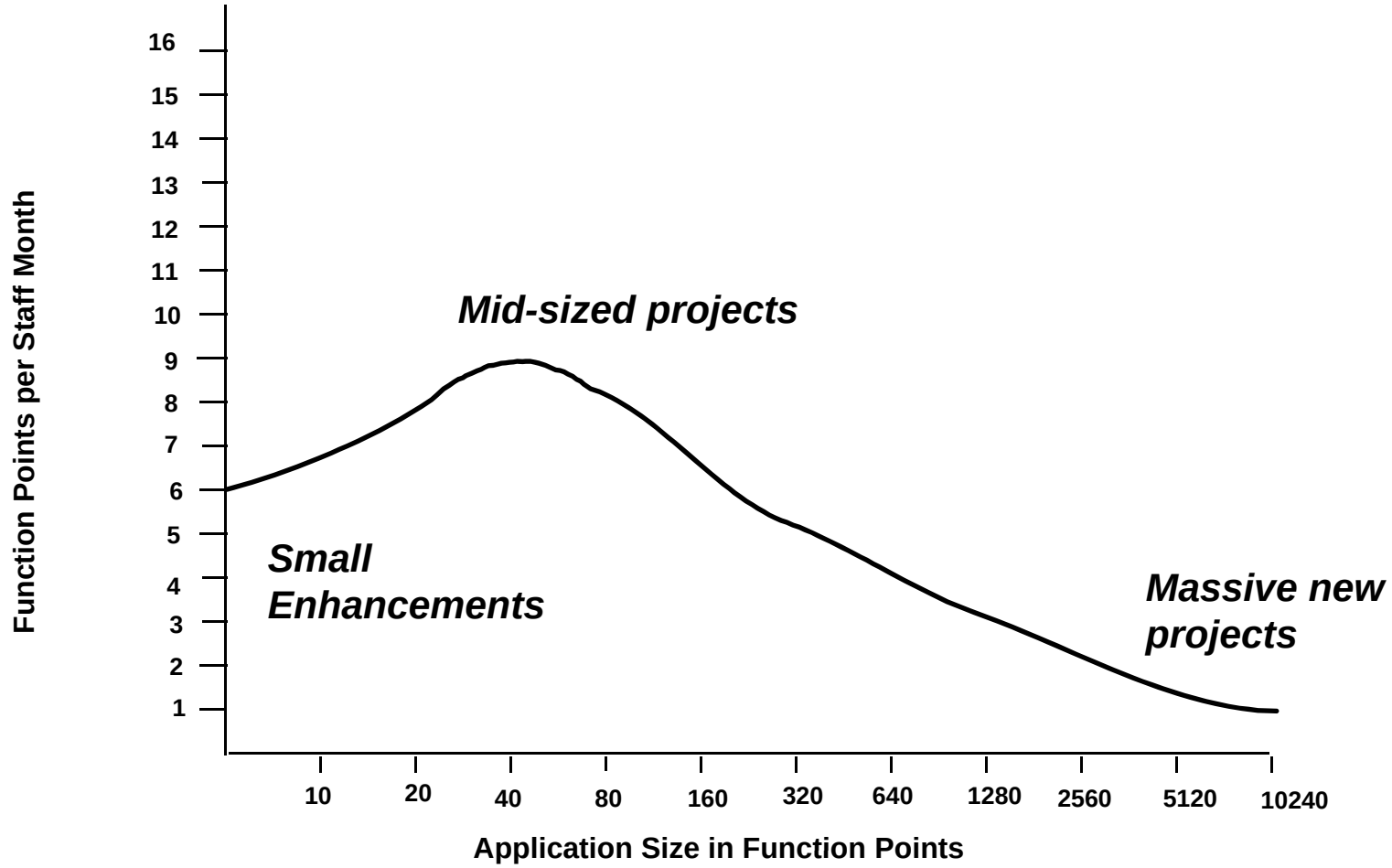
AVERAGE PRODUCTIVITY RATES (NEW PROJECTS)



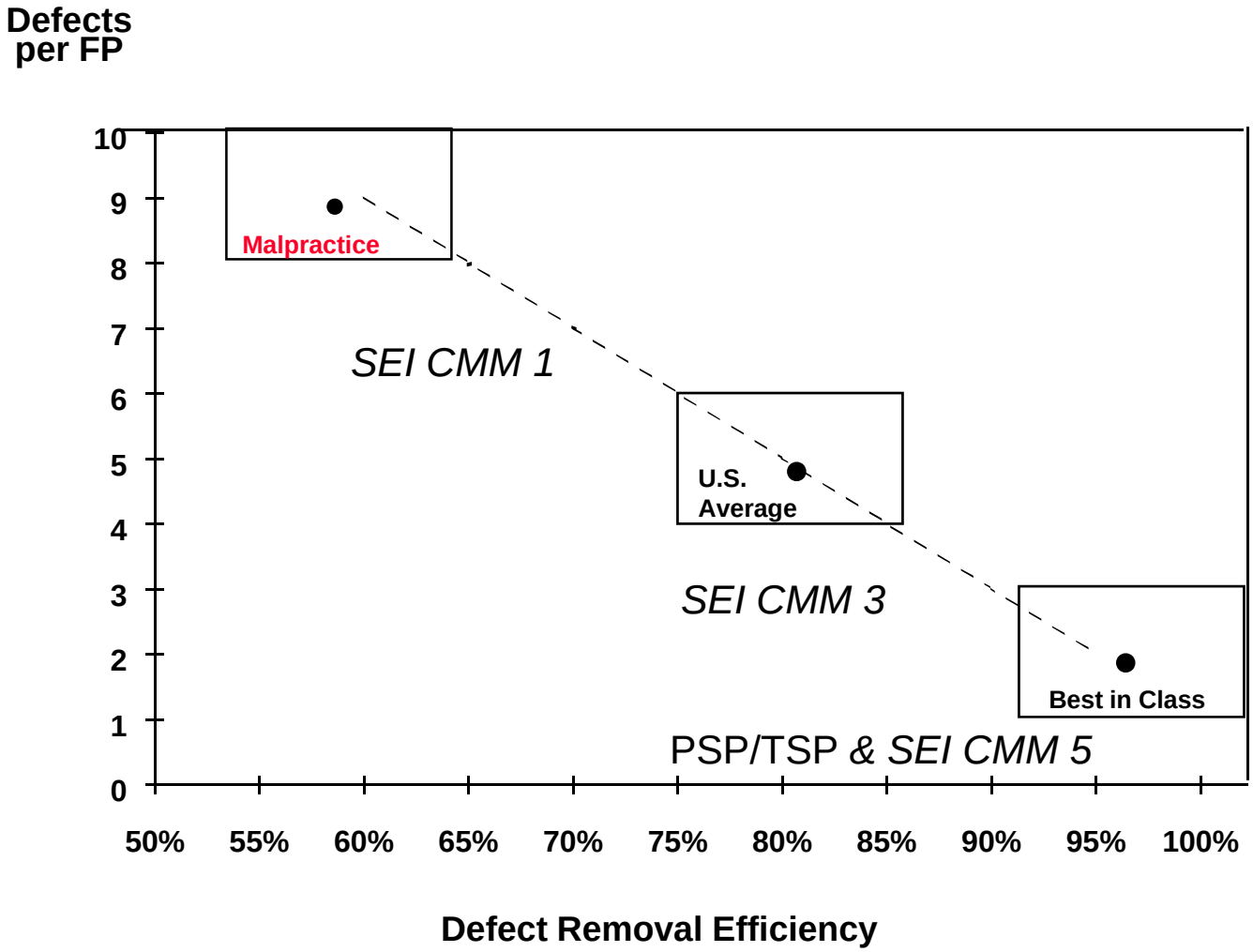
PRODUCTIVITY RATES FOR ENHANCEMENT SOFTWARE PROJECTS



PRODUCTIVITY RATES (OVERALL AVERAGE)



SOFTWARE QUALITY IMPROVEMENT



SEVEN STAGES OF SOFTWARE EXCELLENCE

Stage 0: **Assessment, Baseline, Benchmark analysis**

Stage 1: **Focus on Project Management**

Stage 2: **Focus on Development and Maintenance Methods**

Stage 3: **Focus on New Tools and Approaches**

Stage 4: **Focus on Infrastructure**

Stage 5: **Focus on Reusability**

Stage 6: **Focus on Industry Leadership**

Stage 7: **Focus on continuous improvement forever!**

TIME REQUIRED TO ADVANCE FROM STAGE TO STAGE

(Duration in Calendar Months)

	Enterprise Software Population			
	<10	**11-100**	**101-1000**	**>1000**
Stage 0 Assessment/Baseline	**1**	**2**	**3**	**4**
Stage 1 Management	**3**	**6**	**7**	**8**
Stage 2 Methods	**3**	**6**	**9**	**9**
Stage 3 Tools	**4**	**4**	**6**	**9**
Stage 4 Infrastructure	**3**	**4**	**6**	**9**
Stage 5 Reusability	**4**	**6**	**8**	**12**
Stage 6 Leadership	**6**	**8**	**9**	**9**
Total	**24**	**36**	**48**	**60**

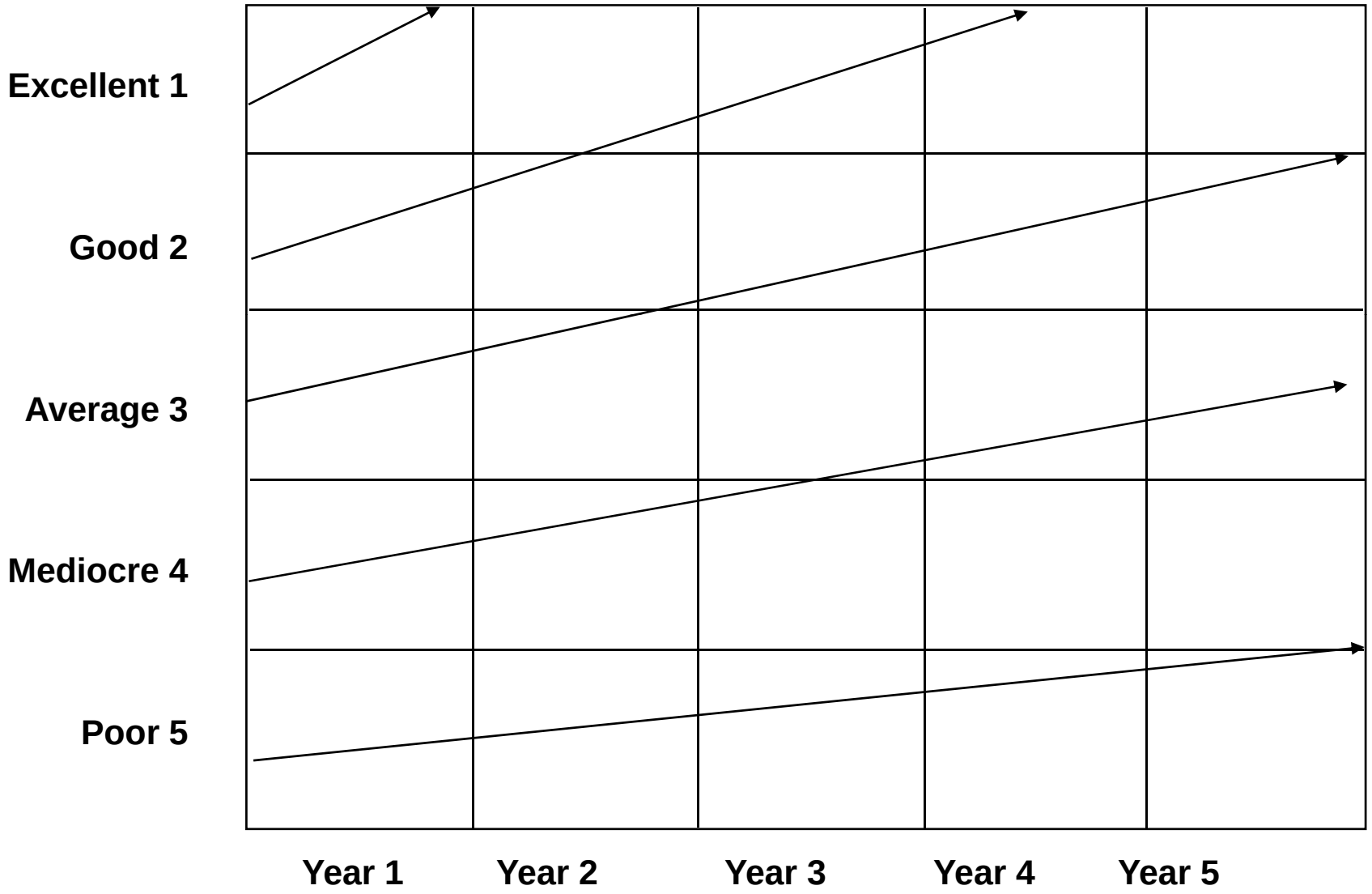
THE QUALITY AND PRODUCTIVITY BENEFITS FROM COMPLETING EACH STAGE

	Defect Reduction	Productivity Increase	Schedule Compression
Stage 0 Assessment	0	0	0
Stage 1 Management	- 10%	0	- 10%
Stage 2 Methods	- 50%	25%	- 15%
Stage 3 Tools	- 10%	35%	- 15%
Stage 4 Infrastructure	- 5%	10%	- 5%
Stage 5 Reusability	- 85%	65%	- 50%
Stage 6 Leadership	- 5%	5%	- 5%
Overall Results	- 90%	350%	- 70%

PROCESS IMPROVEMENT EXPENSES PER CAPITA

	Small < 100 staff	Medium 100-1000	Large > 1000 staff	
Stage 0 Assessment	**$125**	**$150**	**$250**	*SEI CMM 1*
Stage 1 Management	**$1000**	**$2500**	**$3000**	
Stage 2 Methods	**$1500**	**$2500**	**$3500**	*SEI CMM 2 PSP/TSP*
Stage 3 Tools	**$2500**	**$3500**	**$5000**	
Stage 4 Infrastructure	**$1500**	**$2000**	**$3000**	*SEI CMM 3*
Stage 5 Reusability	**$2000**	**$2500**	**$3500**	*SEI CMM 4*
Stage 6 Leadership	**$1000**	**$1000**	**$2000**	*SEI CMM 5*
Overall Results	**$9625**	**$14150**	**$20250**	

RATES OF PROCESS IMPROVEMENT CORRELATED TO INITIAL RANKING



BEST CASE RETURN ON INVESTMENT (ROI)

- **Assume improvement costs of about $1,000,000**
- **Value of better quality $5,000,000**
- **Value of shorter schedules $4,000,000**
- **Value of higher productivity $3,000,000**
- **Value of reduced maintenance $2,000,000**
- **Value of better customer satisfaction $5,000,000**
- **TOTAL VALUE $20,000,000 ***
- **RETURN ON INVESTMENT $20 to $1**

*** Assumes 2 years of improvements and 3 years of results**

UNSUCCESSFUL PROCESS IMPROVEMENT

- **Assume improvement costs of about $1,000,000**
- **Value of better quality $100,000**
- **Value of shorter schedules $100,000**
- **Value of higher productivity $100,000**
- **Value of reduced maintenance $100,000**
- **Value of better customer satisfaction $100,000**
- **TOTAL VALUE $500,000 ***
- **RETURN ON INVESTMENT $0.5 to $1**

*** Assumes 2 years of improvements and 3 years of results**

STAGE 0: ASSESSMENT, BASELINE, BENCHMARKS

Key Technologies

- **SEI Assessment (Levels 1 through 5)**
- **Six-Sigma Baseline, Benchmark**
- **SPR Assessment, Baseline, Benchmark**
- **ISO 9001 - 9004 Audit**
- **TickIT assessment**
- **Putnam Baseline, Benchmark**
- **Gartner Baseline, Benchmark**
- **David's Baseline, Benchmark**
- **IFPUG Baseline, Benchmark**
- **ISBSG Benchmarks (commercially available)**

STAGE 1: FOCUS ON PROJECT MANAGEMENT

Key Technologies

- **Project Sizing**
- **Project Schedule Planning**
- **Project Cost Estimating**
- **Project Quality Estimating**
- **Functional Metrics**
- **Project Measurement**
- **Project Milestone Tracking**
- **Package Acquisition**
- **Risk Analysis**
- **Value Analysis**

STAGE 2: FOCUS ON DEVELOPMENT PROCESSES

Key Technologies

- **Early sizing and risk assessment**
- **Reviews and Inspections**
- **Automated static analysis**
- **Joint Application Design (JAD)**
- **Quality Function Deployment (QFD)**
- **Six-Sigma methodology**
- **Team Software Process (TSP)**
- **Personal Software Process (PSP)**
- **Rational Unified Process (RUP)**
- **Agile or XP methodologies**
- **ISO 9001 - 9004 Certification with caution**
- **SEI maturity levels (CMM and CMMI)**
- **Geriatric technologies for legacy systems**

STAGE 3: FOCUS ON NEW TOOLS & APPROACHES

- **Key Technologies -- New Tools**
  - > **Integrated tool suites**
  - > **Web and Internet Tools**
  - > **Requirements analysis tools**
  - • **Requirements validation tools**
  - > **Static analysis; inspections and automated tools**
  - > **Automated testing tools**
  - > **Reverse Engineering and maintenance tools**

STAGE 4: FOCUS ON INFRASTRUCTURE

Key Technologies

- **Staff Specialization**
- **Formal Measurement Organization**
- **Formal Maintenance Organization**
- **Formal Quality Assurance Organization**
- **Formal Testing Organization**
- **Formal Process Improvement Organization**
- **Improved Hiring Practices**
- **Improved Compensation Plans**
- **Competitive Analysis**
- **Outsource Analysis**

STAGE 5: FOCUS ON REUSABILITY

Key Technologies

- **Reusable Architectures**
- **Reusable Requirements**
- **Reusable Designs**
- **Reusable Interfaces**
- **Reusable Source Code**
- **Reusable Plans**
- **Reusable Estimates**
- **Reusable Data**
- **Reusable Human Interfaces**
- **Reusable Test Plans**
- **Reusable Test Cases**
- **Reusable Documentation**

High quality reuse has best ROI of any technology: > $40 per $1 expended.

Low quality reuse has worst ROI of any technology: > - $15 for every $1 expended.

STAGE 6: FOCUS ON INDUSTRY LEADERSHIP

Key Technologies

- **Baldrige Award**
- **Deming Prize**
- **SEI CMMI Level 5 for major software sites**
- **Best 100 Companies to Work For**
- **Market share grows > 20% from baseline**
- **Time to market better than competitors by > 30%**
- **Acquisition of Competitors**
- **Become a Software Outsourcer**

STAGE 7: Keeping Excellence After Achieving Excellence

Key Technologies

- **Measure results of every project**
- **Produce monthly reports for managers and teams**
- **Produce annual reports for top executives**
- **Publicize results to clients and media**
- **Train new hires in best practices**
- **Inform new executives of best practices!!**
- **Insist on best practices with contractors**
- **Set targets for annual improvements every year**
- **Do not abandon success once it achieved!!**

ATTRIBUTES OF BEST IN CLASS COMPANIES

1. **Good project management**
2. **Good technical staffs**
3. **Good support staffs**
4. **Good measurements**
5. **Good organization structures**
6. **Good methodologies**
7. **Good tool suites**
8. **Good environments**

GOOD PROJECT MANAGEMENT

- **Without good project management the rest is unachievable**
- **Attributes of project good management:**
  - **Fairness to staff**
  - **Desire to be excellent**
  - **Strong customer orientation**
  - **Strong people orientation**
  - **Strong technology orientation**
  - **Understands planning and estimating tools**
  - **Can defend accurate estimates to clients and executives**
  - **Can justify investments in tools and processes**

GOOD SOFTWARE ENGINEERING TECHNICAL STAFFS

- **Without good engineering technical staffs tools are not effective**
- **Attributes of good technical staffs:**
  - **Desire to be excellent**
  - **Good knowledge of applications**
  - **Good knowledge of development processes**
  - **Good knowledge of quality and defect removal methods**
  - **Good knowledge of maintenance methods**
  - **Good knowledge of programming languages**
  - **Good knowledge of software engineering tools**
  - **Like to stay at the leading edge of software engineering**

GOOD SUPPORT STAFFS

- **Without good support technical staffs and managers are handicapped**
- **Support staffs > 30% of software personnel in leading companies**
- **Attributes of good support staffs:**
  - **Planning and estimating skills**
  - **Measurement and metric skills**
  - **Writing/communication skills**
  - **Quality assurance skills**
  - **Data base skills**
  - **Network, internet, and web skills**
  - **Graphics and web-design skills**
  - **Testing and integration skills**
  - **Configuration control and change management skills**

GOOD SOFTWARE MEASUREMENTS

- **Without good measurements progress is unlikely**
- **Attributes of good measurements:**
  - **Function point analysis of entire portfolio**
  - **Annual function point benchmarks**
  - **Life-cycle quality measures**
  - **User satisfaction measures**
  - **Development and maintenance productivity measures**
  - **Soft factor assessment measures**
  - **Hard factor measures of costs, staffing, effort, schedules**
  - **Measurements used as management tools**

GOOD ORGANIZATION STRUCTURES

- **Without good organization structures progress is unlikely**
- **Attributes of good organization structures:**
  - **Balance of line and staff functions**
  - **Balance of centralized and decentralized functions**
  - **Organizations are planned**
  - **Organizations are dynamic**
  - **Effective use of specialists for key functions**
  - **Able to integrate “virtual teams” at remote locations**
  - **Able to integrate telecommuting**

GOOD PROCESSES AND METHODOLOGIES

- **Without good processes and methodologies tools are ineffective**
- **Attributes of good methodologies:**
  - **Flexible and useful for both new projects and updates**
  - **Scalable from small projects up to major systems**
  - **Versatile and able to handle multiple kinds of software**
  - **Efficient and cost effective**
  - **Evolutionary and able to handle new kinds of projects**
  - **Unobtrusive and not viewed as bureaucratic**
  - **Transferable to new hires, contractors, consultants**

GOOD TOOL SUITES

- **Without good tool suites, management and staffs are handicapped**
- **Attributes of good tool suites:**
  - **Both project management and technical tools**
  - **Quality tools (static analysis; testing, etc. are critical)**
  - **Functionally complete**
  - **Mutually compatible**
  - **Easy to learn**
  - **Easy to use**
  - **Tolerant of user errors**
  - **Secure**

GOOD ENVIRONMENTS AND ERGONOMICS

- **Without good office environments productivity is difficult**
- **Attributes of good environments and ergonomics:**
  - **Private office space for knowledge workers (> 90 square feet; > 6 square meters)**
  - **Avoid small or crowded cubicles with 3 or more staff**
  - **Adequate conference and classroom facilities**
  - **Excellent internet and intranet communications**
  - **Excellent communication with users and clients**

MOST EFFECTIVE PROCESS IMPROVEMENT METHODS

1. **Defect removal efficiency measurements**
2. **Function point productivity and quality measurements**
3. **Automated static analysis (C, Java, COBOL, SQL etc.)**
4. **Formal design and code inspections**
5. **Early sizing and early risk assessments**
6. **Joint Application Design (JAD) for requirements**
7. **Automated project management tools**
8. **Automated cost estimating tools**
9. **Automated complexity analysis and reduction tools**
10. **Automated change control tools**
11. **CMMI, TSP and PSP, RUP**
12. **Six-Sigma for software**

BEST METHODS BY SIZE PLATEAU

Function Points	Best Methods
10	PSP, Agile, XP
100	PSP, Agile, XP
1,000	RUP, TSP, XP, Agile
10,000	TSP, RUP
100,000	TSP, RUP

MINIMUM SAFE CMMI LEVEL BY SIZE PLATEAU

Function Points	CMMI LEVEL
10	1 through 5
100	1 through 5
1,000	3 through 5
10,000	3 through 5
100,000	5 only

SOFTWARE IMPROVEMENT GUIDELINES

DO	*DON'T*
• Think long range: 3 to 5 years	• Expect immediate results
• Consider all factors: – Management – Process – Tools – Organization – Skills and training – Programming Languages – Environment	• Concentrate only on Agile methods or any other "silver bullet"
• Plan expenses of up to $15,000 per staff member	• Expect major improvements for minor expenses
• Consider your corporate culture	• Ignore resistance to change

REFERENCES TO PROCESS IMPROVEMENT

Garmus, David & Herron David, Function Point Analysis, Addison Wesley, 2001.

Humphrey, Watts; Managing the Software Process; Addison Wesley, 1989.

Jones, Capers; Assessments, Benchmarks, and Best Practices; Addison Wesley, 2000.

Jones, Capers; Applied Software Measurement, McGraw Hill, 2008.

Jones, Capers; Software Engineering Best Practices; McGraw Hill 2010.

Jones, Capers & Bonsignour, Olivier; The Economics of Software Quality; Addison Wesley, 2011.

Kan, Steve; Metrics and Models in Software Quality Engineering, Addison Wesley, 2003.

Love, Tom; Object Lessons; SIGS Books, 1993.

Pressman, Roger; Software Engineering – A Practitioners Approach; McGraw Hill, 2005.

Wiegers, Karl; Creating a Software Engineering Culture; Dorset House, 1996.

Web sites:

IFPUG.ORG ITMPI.ORG ISBSG.ORG ASQ.ORG SEI.CMU.EDU, IEEE.ORG