

Predicting Software Quality Early in the Software Development Lifecycle and Produce Secure Software

Girish Seshagiri

Atlanta SPIN
February 16, 2010

Trademarks And Service Marks

The following are service marks of Carnegie Mellon University.

CMMISM

Team Software ProcessSM

TSPSM

Personal Software ProcessSM

PSPSM

The following are registered trademarks of Carnegie Mellon University.

Capability Maturity Model[®]

CMM[®]

Capability Maturity Model[®] Integration

CMMI[®]

CERT[®]

The following are registered service marks of Advanced Information Services Inc.

No Surprise Project ManagementSM

The following are registered trade marks of Advanced Information Services Inc.

CPIW[®]

SOLONsys[®]

ais

Preamble

Don't think of business as a life without greatness
Unless the distant goals of meaning, greatness, and
destiny are addressed, we can't make an
intelligent decision about what to do tomorrow
morning – much less set the long-term strategy of
the company

First decision must be to commit to an ethical world, a
civilized existence, a moral order

Nothing is more practical than for people to deepen
themselves

- *Peter Koestenbaum (pkipeter@ix.netcom.com)*

Managing the Software Work

State of the Practice - 1

In spite of increased adoption of PMP, Six Sigma, Lean, Agile, CMMI, ITIL etc, industry performance outcomes for cost, schedule and quality continue to be mixed

Changing managers, procurement regulations, acquisition procedures, and contracting provisions including oversight reporting have not resolved the cost, schedule, and quality problems of most software projects

Even with all of the emphasis on writing software with security in mind, most software applications remain
ais riddled with security holes

Managing the Software Work State of the Practice - 2

Current software quality methods rely on testing, and are expensive, unpredictable and ineffective

Heroic efforts rescue troubled projects; heroes are in short supply

High-performance teams are the exceptions and not the norm

Why Are We Here? - 1

The adverse impact of software vulnerabilities caused by defective software is far-reaching

The defects that escape testing are exploited by hackers to launch cyber attacks

The current method of dealing with the increasing number of cyber attacks is reactive

Why Are We Here? - 2

We need a rapid transformation of the U.S. software industry from the current “Deliver now, fix later” culture to one capable of delivering substantially defect free code within predictable cost and schedule

This is a national high-priority need

If we continue with current methods, the U.S. taxpayer will pay billions of dollars for fixing defects in delivered products

Why are We Here? - 3

First step is to make quality the number one priority and recognize that in order to manage the software work we must learn to manage quality

By adopting proven principles of managing knowledge work, software development quality and productivity can be increased by orders of magnitude

The leadership challenge is to build a cohesive and rewarding team environment where most people can do much better work than they are currently doing and produce truly amazing results

Common Misconceptions About The Software Work - 1

We must start with firm requirements

If it passes test, it must be OK

Software quality can't be measured

The problems are technical

We need better people

Software management is different

Source: *Managing the Software Process*, Watts Humphrey

Common Misconceptions About The Software Work – 2

Maturity level 3 is all that is needed

Higher maturity levels add to cost

Higher maturity levels are needed only for safety critical or business mission critical systems

If it is “agile” or “lean”, it is good

What we need are lean processes

Maturity levels guarantee results

Maturity level 5 is the end

Contracts and Management Systems - 1

Too often, our contracts and management systems are based on the common misconceptions

Time and Materials and Cost Plus Fee contracts end up rewarding poor cost, schedule and quality performance

CMMI levels 2 or 3 specified as pass/fail criteria

Emphasis on product technical reviews which are important, but do not motivate change and are not helpful to improve the process or track status

Contracts and Management Systems - 2

Not trusting teams to estimate the job, prepare aggressive and realistic plans, and negotiate responsible commitments

Lack of precision and timeliness in reporting contract performance and job status

Multi-contractor teams lack sense of purpose, goals and mission and do not act as a real team

The Top 15 U.S. Software Cost Drivers in Rank Order Circa 2009

1. **The cost of finding and fixing bugs**
2. The cost of cancelled projects
3. The cost of producing paper documents and English words
4. The cost of recovery from security flaws and attacks
5. The cost of requirements changes during development
6. The cost of programming or coding
7. The cost of customer support
8. The cost of meetings and communication
9. The cost of project management
10. The cost of application renovation
11. **The cost of innovation and new kinds of features**
12. The cost of litigation for cancelled projects
13. The cost of training and learning software applications
14. The cost of avoiding security flaws
15. **The cost of acquiring reusable components**

Schedule Compression and Quality

Schedule/Quality Trade-off				
	Default	10% Compression	20% Compression	10% Extension
Duration Mths	25.9	23.3	20.7	28.5
Defect Count	1,033	1,316	1,715	849
% Change		27.4%	66.0%	-17.8%

Source: Donald M. Beckett and Douglas T. Putnam, STN 13-1 April 2010: Software Quality, Reliability, and Error Prediction

Adding Staff and Quality

Staff/Quality Trade-off			
	Peak Staff 16	Peak Staff 32	% Change
Duration Mths	26	22.6	-13.1%
Defect Count	1,043	1,411	35.3%
Effort Months	225	392.0	74.2%

Source: Donald M. Beckett and Douglas T. Putnam, STN 13-1 April 2010: Software Quality, Reliability, and Error Prediction

Defect Injection Rate and Removal Percent - Sample

Phase	Injection Rate – Defects Per Hour	Percent Defects Removed - Yield
Requirements	0.5	
High Level Design	0.5	
Detailed Design	1.5	
Code	2.0	
Unit Test		50%
Integration Test		55%
System Test		35%

Your organization/projects have such data?

Defects Per Function Point

Defect Origins	Defect Potentials	Removal Efficiency	Delivered Defects
Requirements	1.00	77%	0.23
Design	1.25	85%	0.19
Coding	1.75	95%	0.09
Document	0.60	80%	0.12
Bad Fixes	0.40	70%	0.12
<i>Total</i>	<i>5.00</i>	<i>85%</i>	<i>0.75</i>

Source: *Capers Jones, STN 13-1 April 2010: Software Quality, and Software Economics*

Managing Software Quality - 1

To meet schedule and cost commitments consistently, you must manage software quality

Quality without numbers is just talk

The common ways to manage software quality are with testing and reuse

Testing is now relied upon and is not sufficient

For reuse, the parts must be initially of high quality or the quality problems will be worse

ais

Managing Software Quality - 2

Software-intensive products typically have many defects

The three defect removal strategies

Test, test, test

Inspect and test

Review, inspect and test

Time to find and fix test defects can vary from a few hours to few weeks

Managing Software Quality - 3

Quality work is more predictable

If you do not manage quality, your schedule problems will end up as quality disasters

Software professionals must be trained to make plans and negotiate commitments

Deliver now, fix later - 1

Why do competent software professionals agree to delivery dates when they have no idea how to meet them?

Why do rational managers accept schedule commitments when engineers offer no evidence that they can meet the commitments?

Deliver now, fix later - 2

If it doesn't have to work any body can deliver on time

If you want the product in the worst way, that's how you will get it

If the situation looks truly impossible, it probably is

Schedule is what must happen; quality determines what will happen

Negotiated Commitments - Developers

When pressed for early deliveries, the responsible team members say

“I understand your requirements, I will do my utmost to meet it, but until I make a plan, I can not responsibly commit to a date”

Negotiated Commitments - Managers

When pressed for early deliveries, the responsible managers say

“I trust you to create an aggressive and realistic plan, I will review the plan, but I will not commit you to a date that you can not meet”

Managing the Software Work - 1

Managing the software work is difficult, because of the way it is currently done

Poor quality is the principal cause of cost and schedule problems

To manage the software work, you need to manage software quality

Managing the Software Work - 2

Software and systems development is knowledge work

The first rule for knowledge work is that managers can't manage it - the workers must manage themselves

Managing the Software work - 3

The second rule is that developers and their teams
Must know how to manage themselves
Negotiate their commitments with management
Manage with data
Own their own work

The third rule is that management must trust the
development teams to plan and manage their
own work

Source: Acquiring Quality Software, Watts Humphrey

Current Methods

Command and control management system not suitable for managing knowledge work

Workers and managers have different objectives and different views of project success

Lack of trust between management and teams

Teams unable to negotiate schedule and cost commitment

Arbitrary, unrealistic schedules creating culture of “Deliver now, fix later”

Lack of precise project status measures

Management Principles for Knowledge Work

Trust the knowledge workers

Build trustworthy teams

Rely on facts and data

Manage quality

Provide leadership to support knowledge workers to manage themselves

Source: "Why Can't We Manage Large Projects", Watts Humphrey, Crosstalk, July/August 2010

Goals and Measurement

Myths and Facts - 1

Dr. Deming

Numerical goals accomplish nothing

Extrinsic motivation leads to the destruction of the individual

Rewards motivate people to work for the rewards

Various components should work together for optimization of profit and joy in work

System must create something of value, in other words, results

Life is variation

Goals and Measurement

Myths and Facts - 2

Watts Humphrey

Undisciplined or unmotivated people can not do timely or predictable intellectual work

Disciplined and motivated people need aggressive goals; support goals with specific programs and plans

You can't easily tell the quality of a program, but you can ask if it was properly developed

If measures can not detect one-day slip, you can not anticipate problems and prevent them

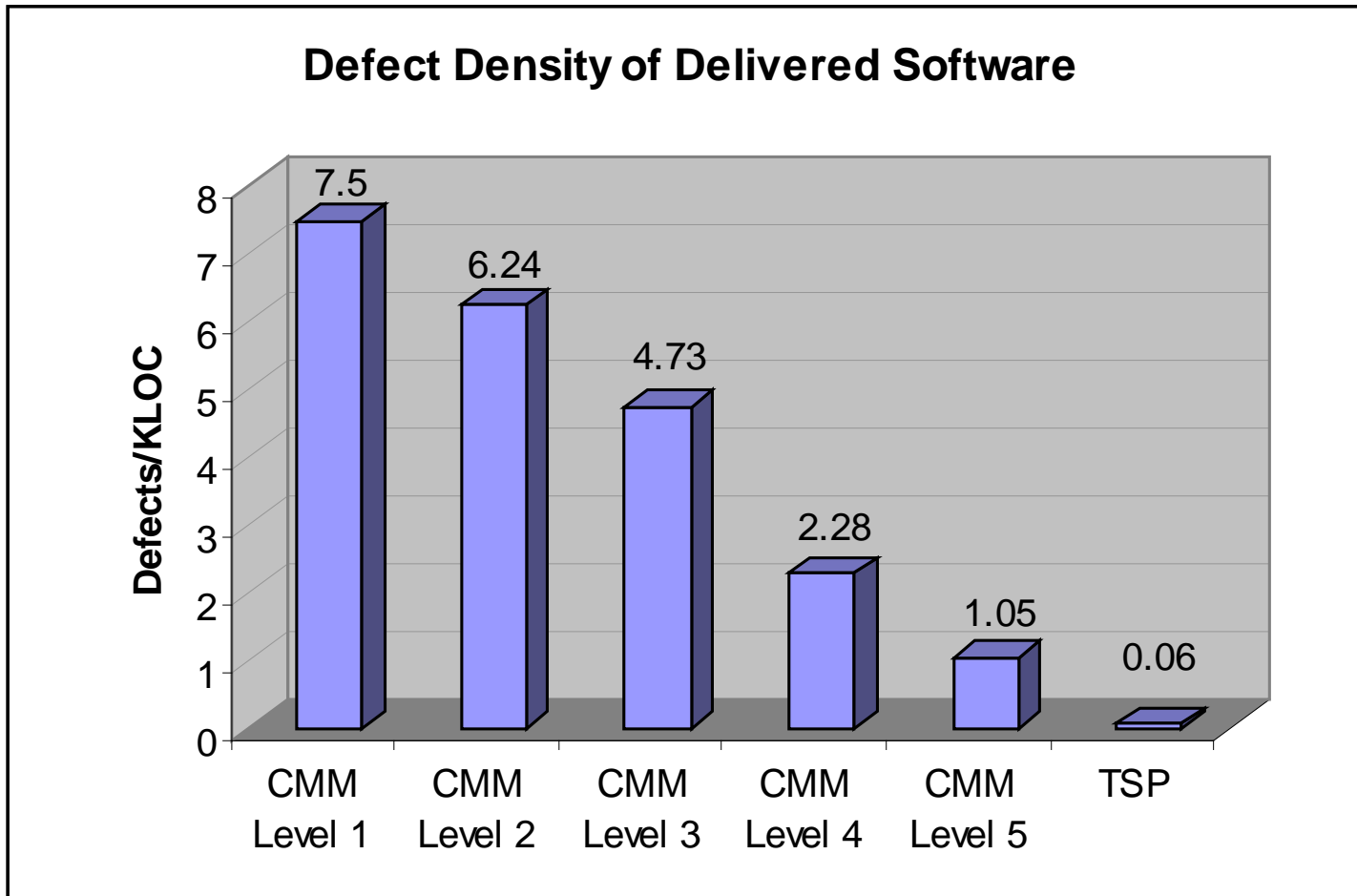
Defining measures is not always easy, but it is almost always possible

Planning is the first step for everything you do

Business is prediction

Performance Metrics That Matter

Post-delivery Defects



Source: Davis, N., Mullaney, J. The Team Software Process (TSP) in Practice: A Summary of Recent Results. CMU/SEI-2003-TR-014. September 2003

Performance Metrics That Matter

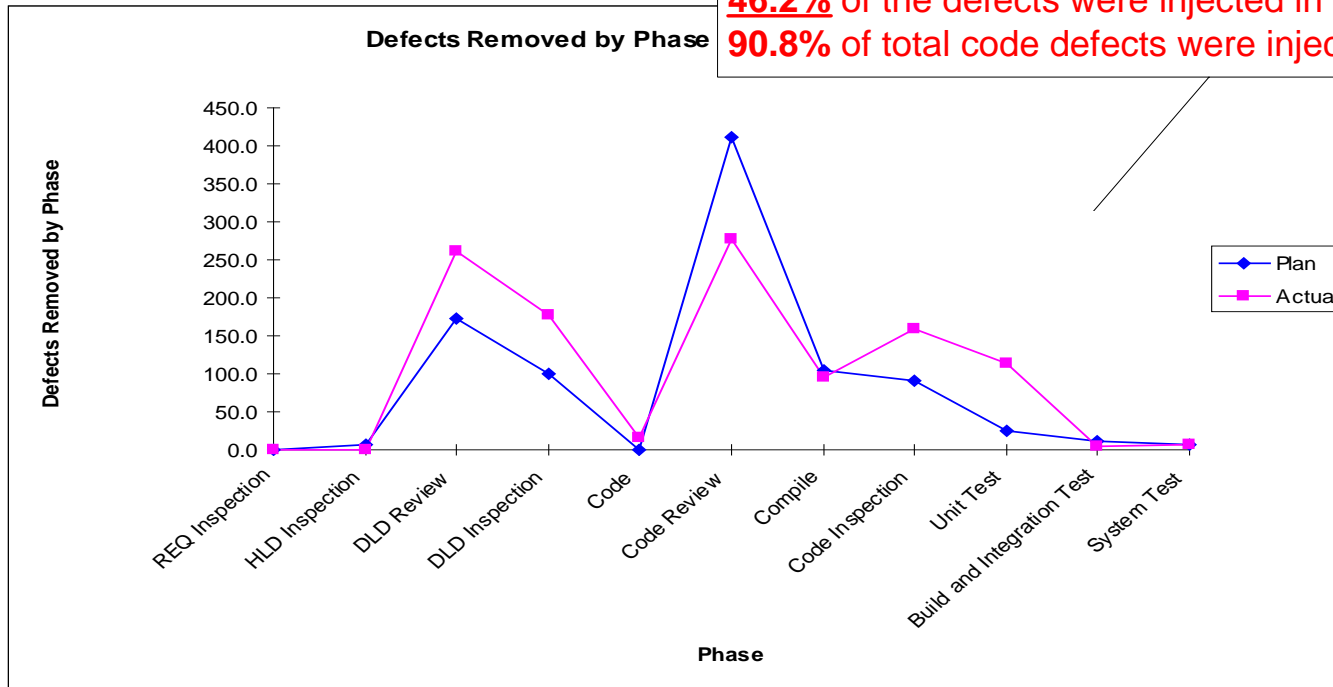
Defects Removed by Phase

The project Injected and Removed a total of 1112 code defects throughout the project lifecycle

44.6% of the defects were injected in Detailed Design

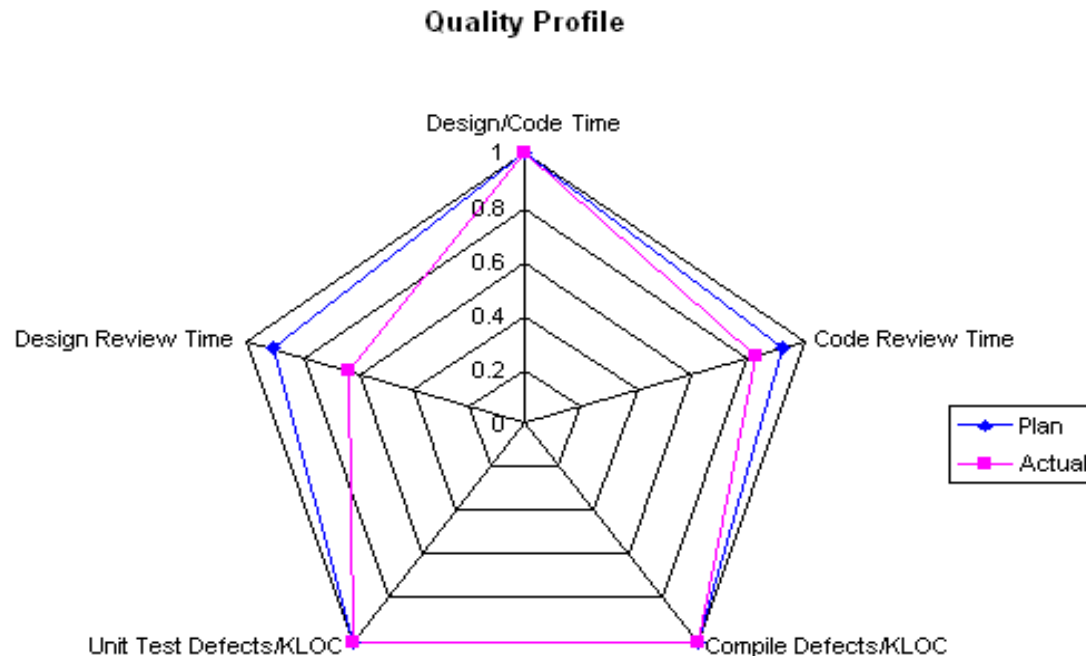
46.2% of the defects were injected in Code

90.8% of total code defects were injected Design & Code



Performance Metrics That Matter

Quality Profile



The above chart shows our 5 measures to achieve our quality goals:

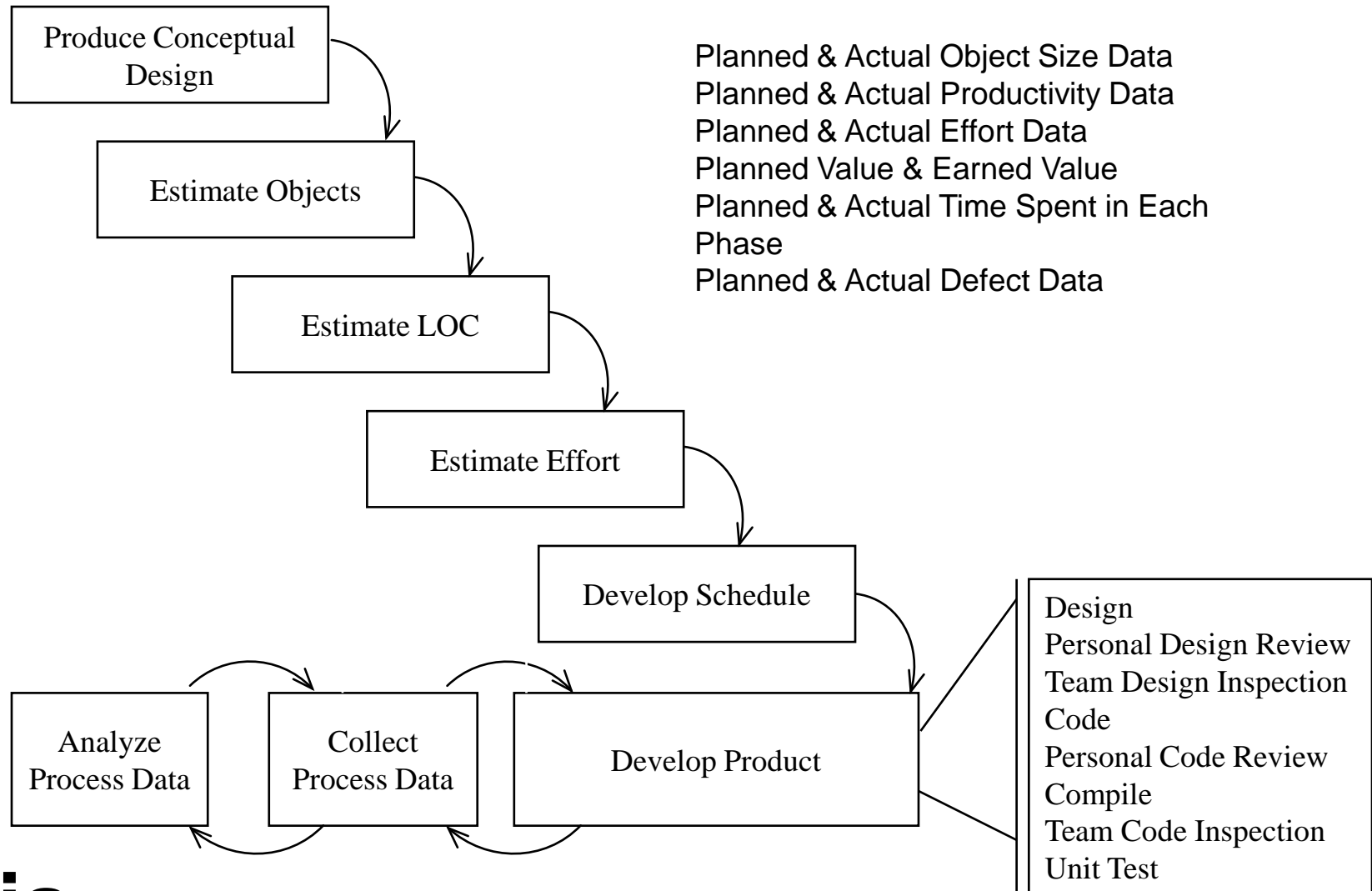
- 1) Design time to Code time comparison
- 2) Design Review Time as a % of design time (should be 50% or greater)
- 3) Code Review Time as a % of code time (should be 50% or greater)
- 4) Compile Defects per KLOC (Compile should find less than 10 defects per KLOC)
- 5) Unit Test Defects per KLOC (UT should find less than 5 defects per KLOC)

Performance Metrics That Matter

Benchmarking

	Industry Average	AIS Average
Schedule deviation	>50%	<10%
No. of defects in delivered product 100,000 LOC	>100	<15
% of design and code inspected	<100	100
Time to accept 100,000 LOC product	10 Months	5 Weeks
% of defects removed prior to system test	<60%	>85%
% of development time fixing system test defects	>33%	<10%
Cost of quality	>50%	<35%
Warranty on products	?	Lifetime

Component Development Process



Performance Metrics That Matter

Early Defect Removal

Developers measure and manage the quality of their individual components and remove defects early through personal reviews and team inspections

Include OWASP Top Ten and CWE/SANS Top 25 Most Dangerous Programming Errors in review and inspection checklists

Developers strive to get the highest quality product
ais into test

Performance Metrics That Matter

Team Goals for Phase Yields

Phase Yields (% defect removed)	Goal
Requirements Inspections	70%
Design reviews and inspections	70%
Code reviews and inspections	70%
Compiling	50%
Unit test at 5 or less defects/KLOC	90%
Integration and system test at <1.0 defects/KLOC	80%
Before compile	>75%
Before unit test	>85%
Before integration test	>97.5%
Before system test	>99%

Software Components - 1

To manage software product quality, we need to manage quality of the components

Components are what developers build

Error-prone components account for a disproportionate number of defects found in integration, system and user acceptance tests

Software Components - 2

20% of the modules in a system typically account for 80% of the defects

It is extremely useful to know which components are likely to be error-prone in later testing so that we can take corrective actions pro-actively

Performance Metrics That Matter

Process Quality Index

PQI is a leading indicator of overall product quality

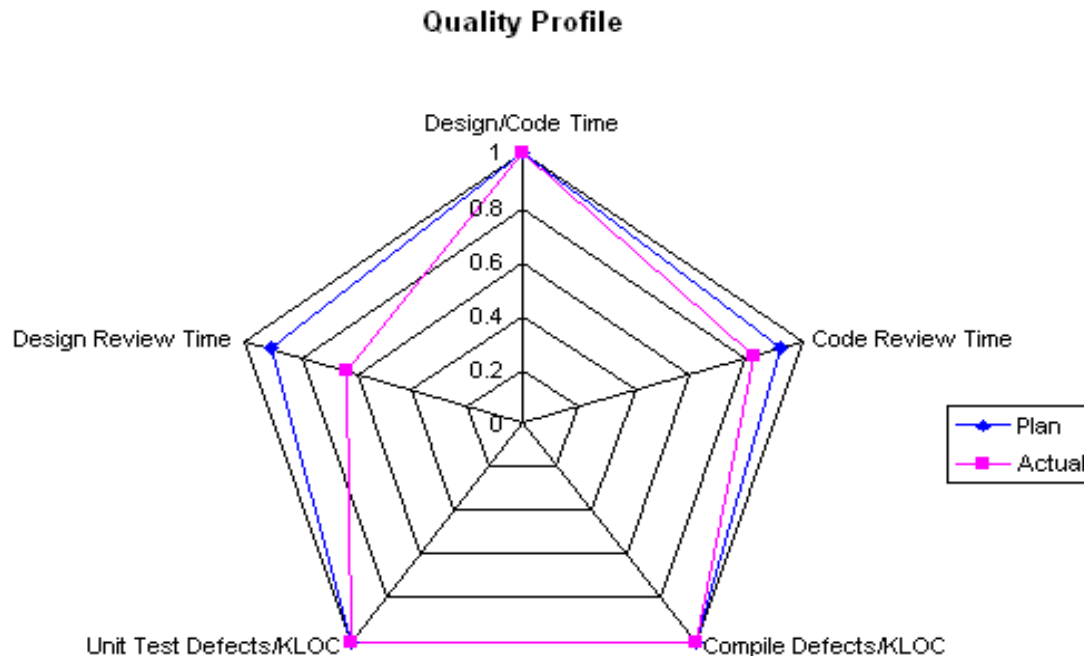
PQI gives ability to predict whether components that have been unit tested will have down-stream defects in integration, system, and user acceptance testing

Teams can take corrective action and reduce test and rework time

PQI Calculation

1. Design time to Code time comparison
2. Design Review Time as a % of design time (should be 50% or greater)
3. Code Review Time as a % of code time (should be 50% or greater)
4. Compile Defects per KLOC (Compile should find less than 10 defects per KLOC)
5. Unit Test Defects per KLOC (UT should find less than 5 defects per KLOC)

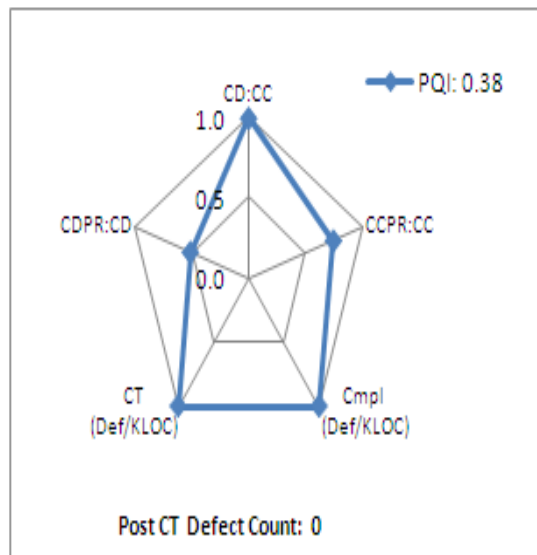
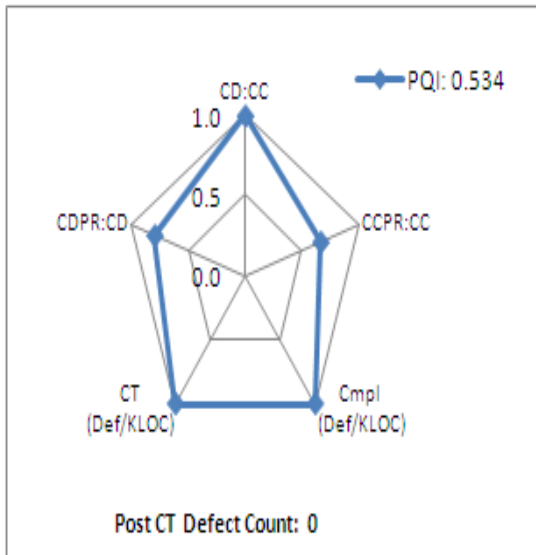
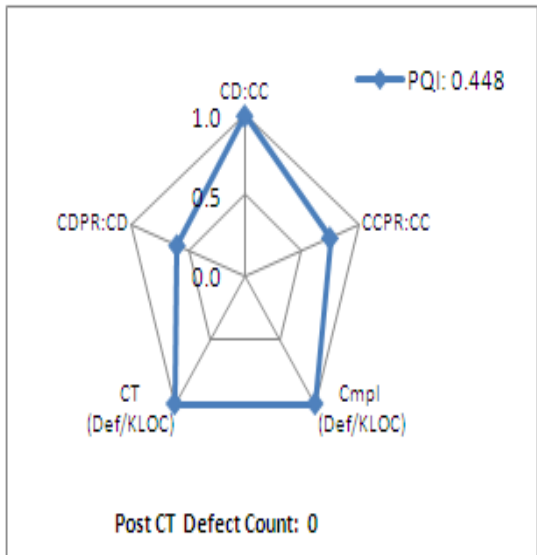
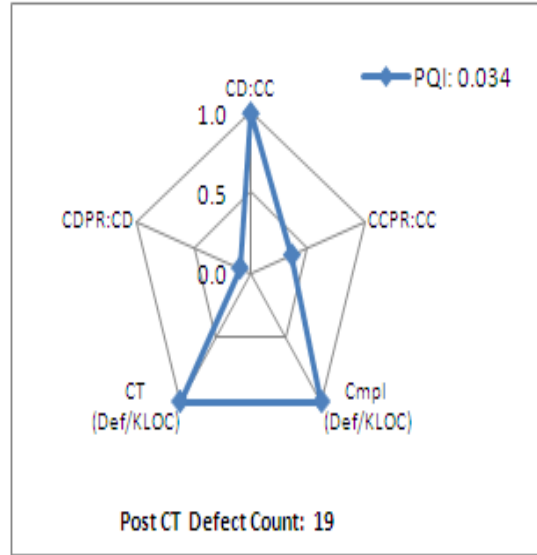
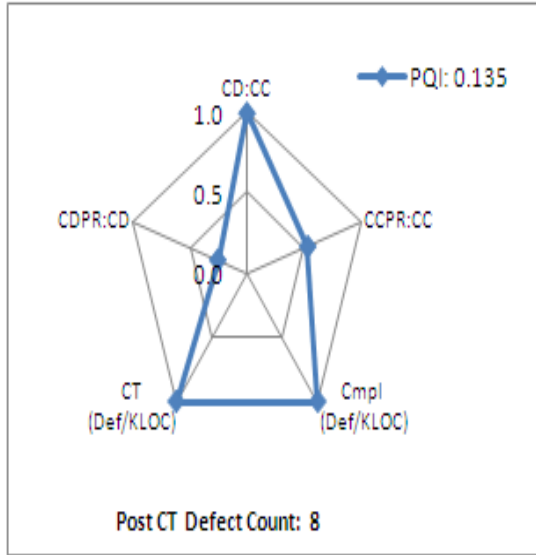
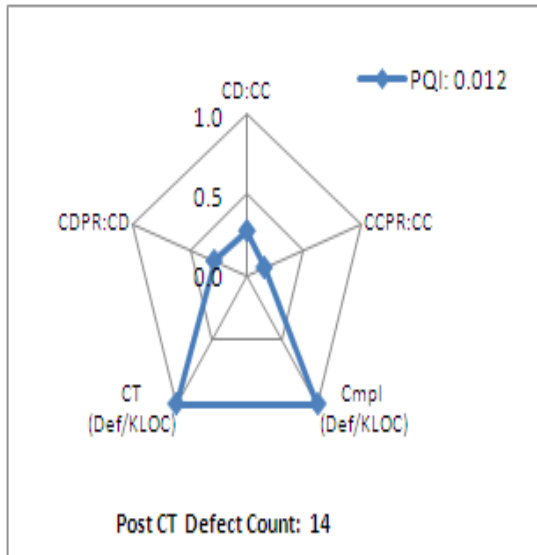
Component Quality Profile



By plotting the five PQI values on radar charts, a component's quality profile can be visually identified in time to take corrective action



AIS Federal Project



Performance Metrics That Matter

Percent Defect Free Components

When knowledge workers have been trained and know how to manage themselves, they are capable of giving early warning to management when problems arise

A key metric is the percent of components in the product that are defect free

Management can motivate the technical teams to strive for 100% defect free components when both parties view project success the same way



Predicting Quality - 1

Developers gather, analyze and report size, time, and defect data

To manage software quality, developers need to manage quality of the software components built by them

Developers practice early defect removal to put the highest quality product into test

Predicting Quality -2

Process Quality Index (PQI) is a metric that is used to predict whether a component is likely to have downstream defects in integration, system, and acceptance tests

Teams should strive for 100 percent defect free components in the products they build

Microsoft Fixed Schedule Project - 1

Teams negotiated features and resources to fit the schedule

Team members were empowered to stop a project from going into Code, SIT, UAT or Production

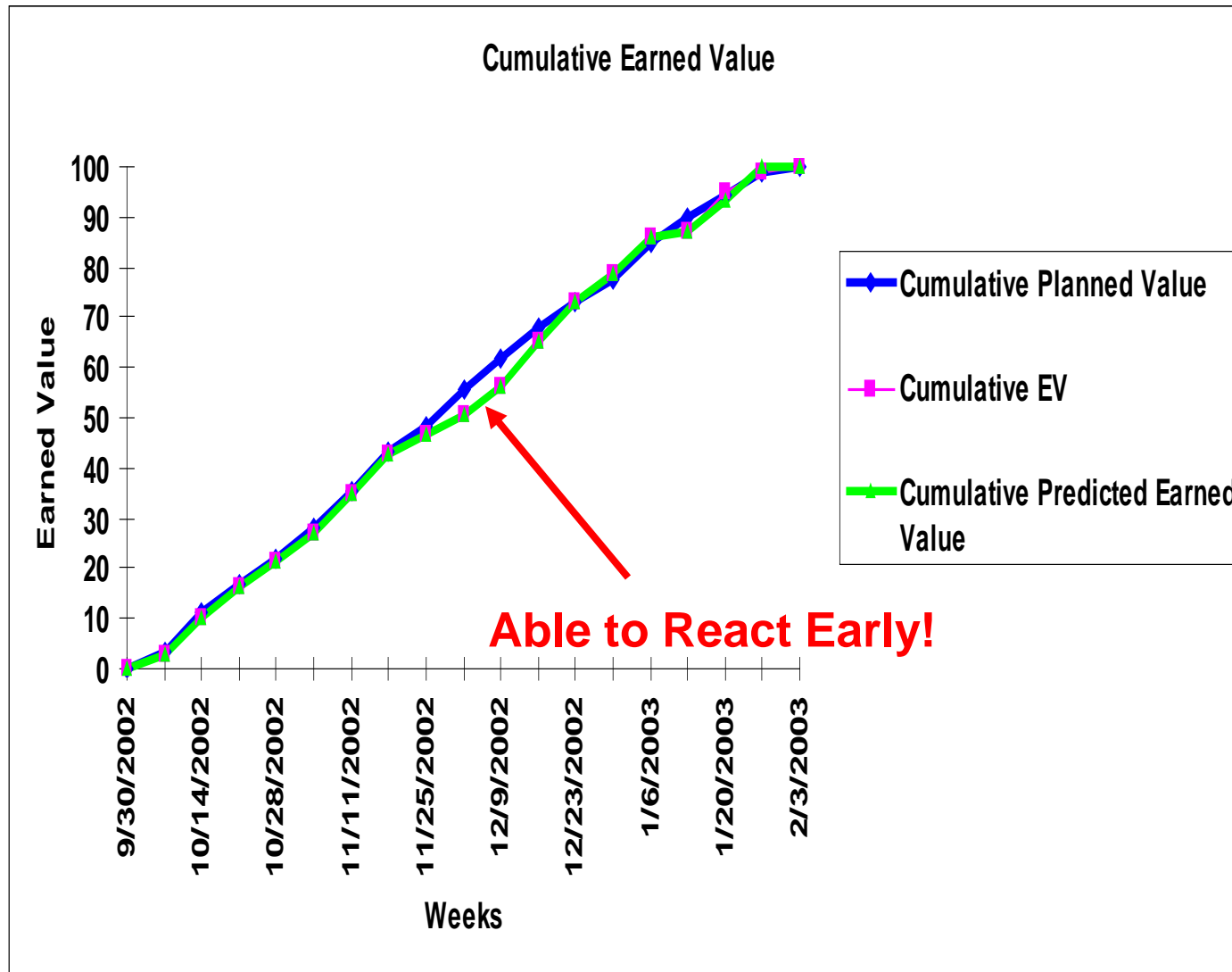
Predicted software quality early using metrics

Achieved work life balance

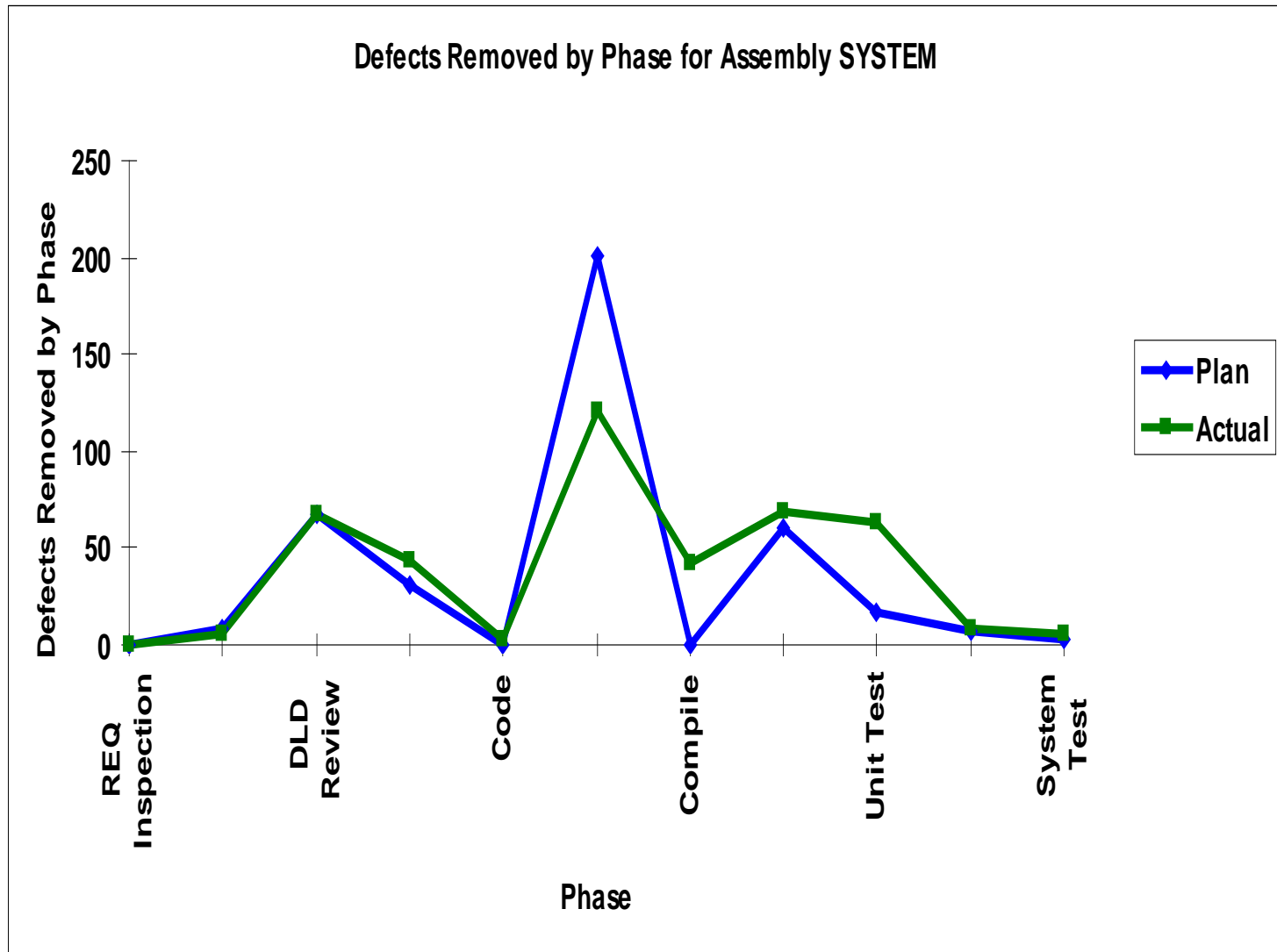
Developers released early from projects

Low budget for post production defects

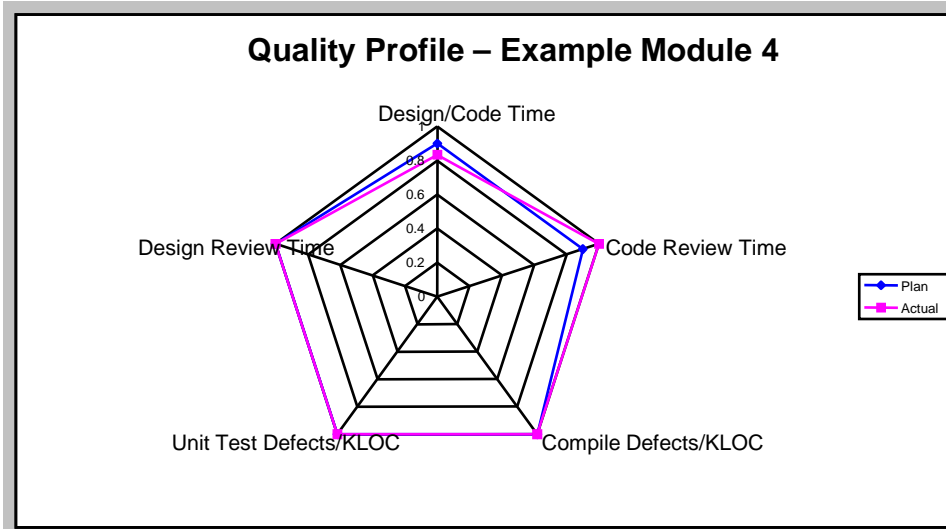
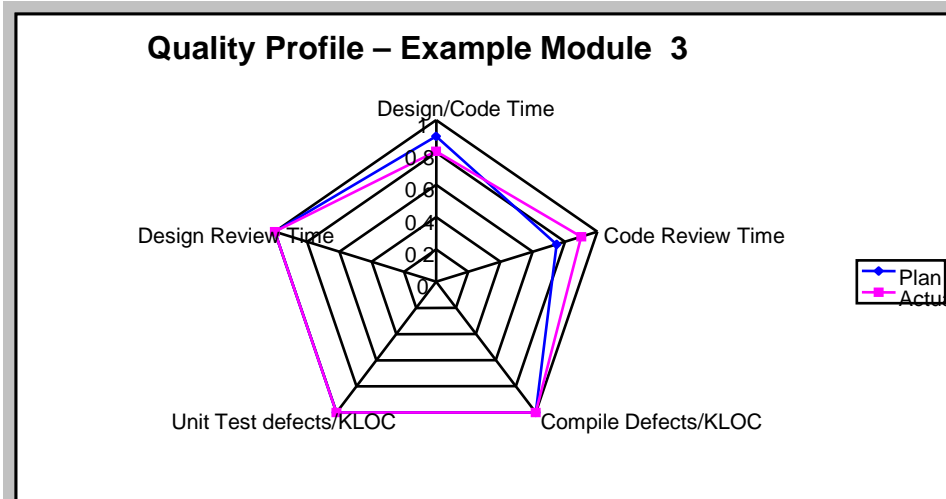
Microsoft Fixed Schedule Project - 2



Microsoft Fixed Schedule Project - 3



Microsoft Fixed Schedule Project - 4



- Planned correctly
- Executed correctly
- Planned & Actual PQI > 0.7
- Zero system test defects

Intuit

Reduced project risk by finding 86% of defects before System Testing even started

Engineers doubled available time for developing features by reducing rework and testing time to <10%

Improved quality and time-to-market, while eliminating work that adds no value at Intuit

“Engineers love it... Once they adopt it they can’t imagine going back”

Source: Jim Sartain, Director, Intuit

Adobe

Teams spent four times less rework than typical Adobe software projects

Teams removed 90% of total bugs prior to systems test

Teams plan, track, measure, and improve

Measurement has encouraged other improvements

- Improved design practices

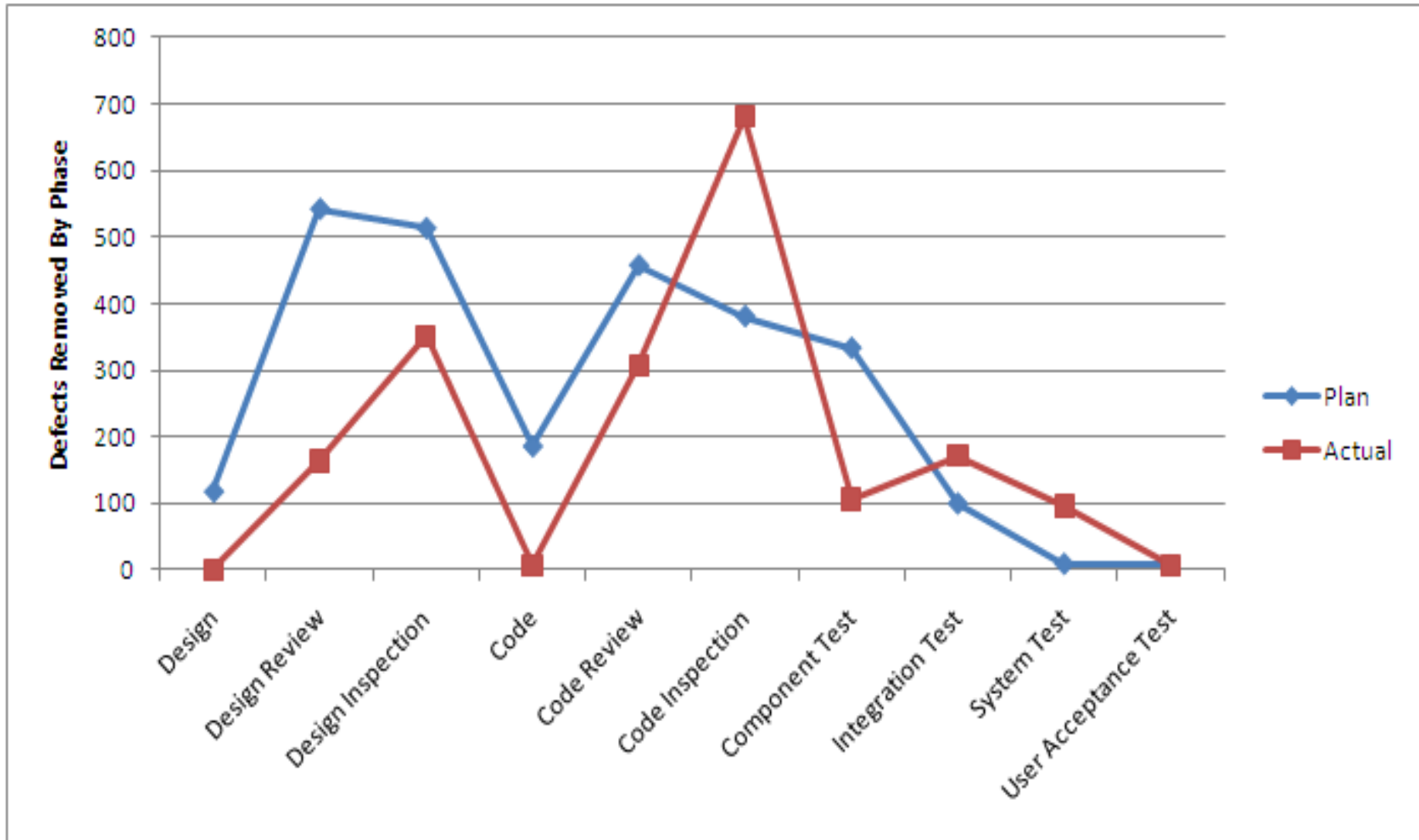
- Defect analysis to determine common causes and implement improvement methods

- Better estimation methods

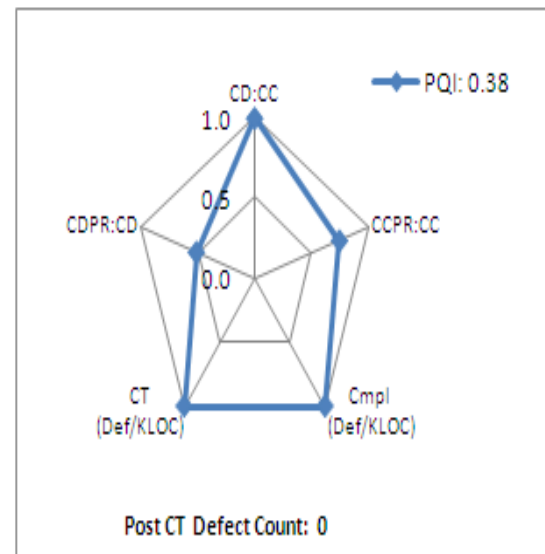
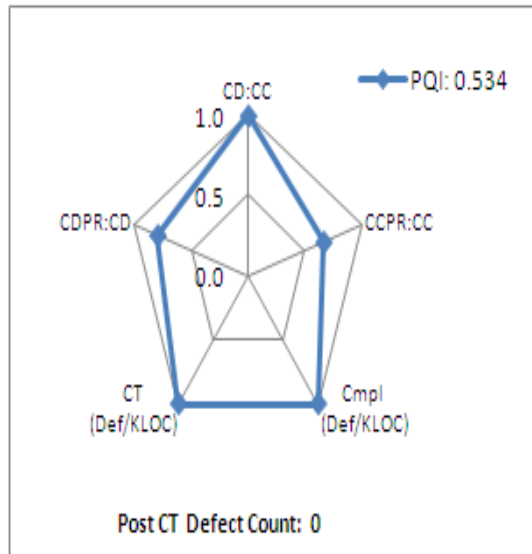
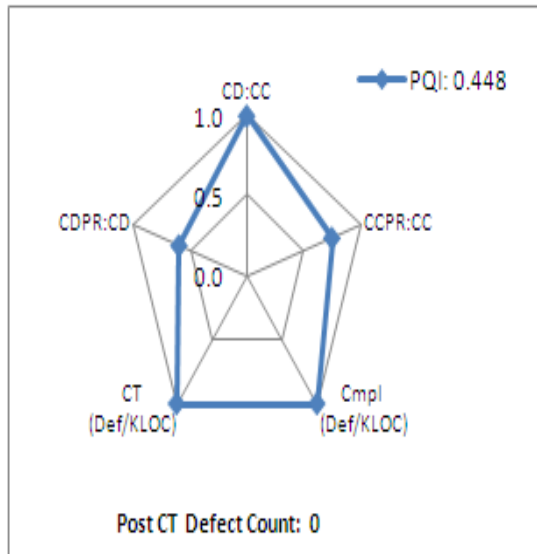
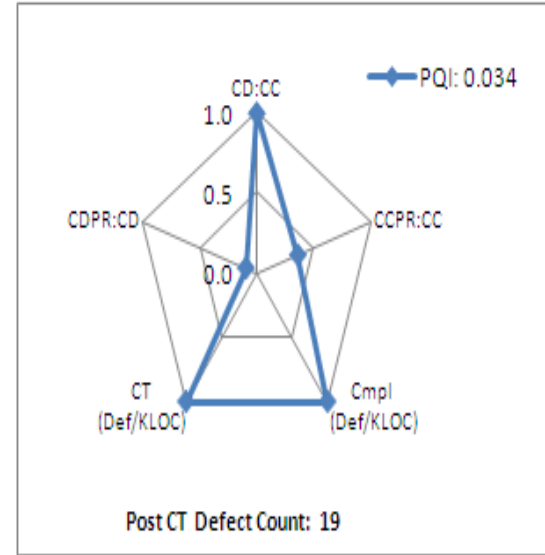
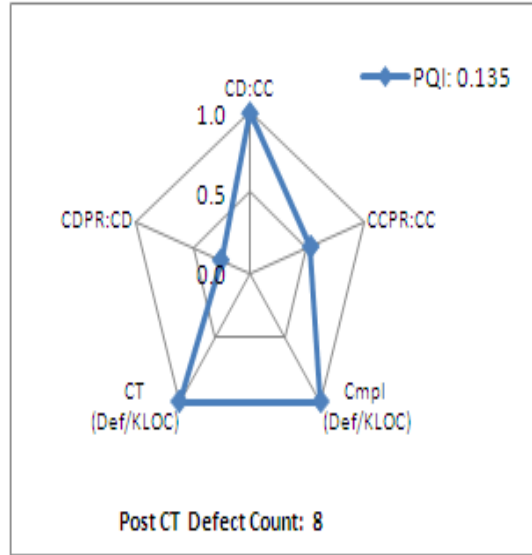
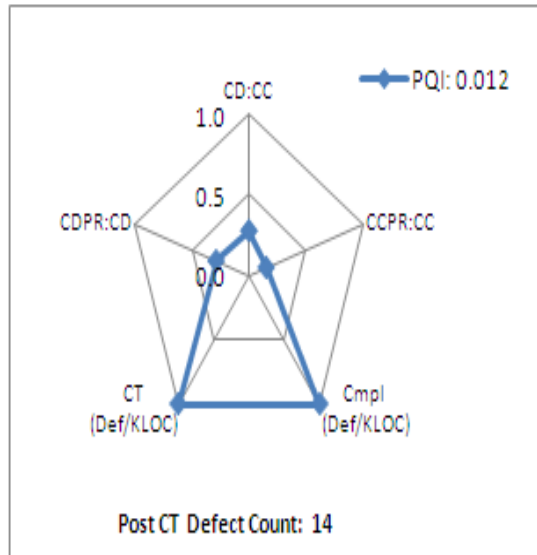


Source: The Evolution of Quality at Adobe, SEPG North America, March 20

AIS Federal IT Project Results - 1



AIS Federal IT Project Results - 2



Project Results

Some of the world's biggest software organizations have reported successful results by empowering developers and teams to successfully negotiate commitments and deliver high quality on schedule and cost

All projects practiced early defect removal and did not rely on testing to find defects

Human Motivation - 1

Unhappy people rarely do quality work

People generally get more satisfaction by doing the job the proper way

Need transformation to a new management model –
“people jointly working for themselves in democratic teams”

Human Motivation - 2

“ What motivates technical professionals is what motivates volunteers, who have to get more satisfaction from their work than paid employees, precisely because they do not get a paycheck. ”

Peter Drucker

The Executive Transformation - 1

Pre-TSP

“If we don’t meet the customer’s schedule expectations, we will lose the business”

“I will gamble on a vague promise”

“I will tell you what must happen”

TSP

“Show me a plan to meet customer’s aggressive schedule. I will not commit you to a date, you can’t meet”

“We will win with a firm commitment based on a plan”

“Let us analyze the data and determine what will happen”

The Executive Transformation - 2

Pre-TSP

“Where are you on the schedule?”

“What have you done for me lately?”

“I will authorize overtime”

TSP

“How is the quality in reviews, inspections, and test?”

“What are you learning today that will help us to meet future challenges?”

“Have fun doing this project”

Do You Agree?

“The best engineers and scientists don’t work for a company, a university, or a laboratory; they really work for themselves.”

Watts Humphrey

“Successful careers are not planned. They develop when people are prepared for opportunities because they know their strengths, their method of work, and their values.”

Peter Drucker

“Anyone that enjoys his work is a pleasure to work with.”

Edwards Deming

<p>Necessary Conditions For Self-Motivation</p>	<p>The Source Of Self-Motivation</p>	<p>Outcome of Fulfilled Motivation</p>
<p>Individual Emotional and Mental Health</p> <p>Safety and Trust</p> <p>Dignity and Self-Esteem</p> <p>Autonomy and Freedom</p> <p>Perspective</p>	<p>Calling Service Purpose Connectedness Interesting work Working with others</p> <p>MEANING</p> <p>Challenge Excelling Creating Learning Beauty Love</p>	<p>Joy</p>

Source: Abolishing Performance Appraisals, Coens and Jenkins, 2000

What does
“FUN ON THE JOB”
Mean to you?

Contact Information

Girish Seshagiri

Advanced Information Services Inc.

(703) 426 2790

Email: girish.seshagiri@advinfo.net

Website: www.advinfo.net

For More Information - 1

Visit the SEI Website for CMMI, TSP

<http://www.sei.cmu.edu/cmmi>

<http://www.sei.cmu.edu/tsp>

See the books

Winning with Software, by Watts Humphrey, Addison-Wesley, 2002

PSP: A Self-Improvement Process for Software Engineers, by Watts Humphrey, Addison-Wesley, 2005.

TSP: Leading a Development Team, by Watts Humphrey, Addison-Wesley, 2006

TSP: Coaching Development Teams, by Watts Humphrey, Addison-Wesley, 2006

For More Information - 2

Contact a TSP transition partner

<http://www.sei.cmu.edu/collaborating/partners/trans.part.psp.html>

Contact SEI customer relations

Phone, voice mail, and on-demand FAX: 412/268-5800

E-mail: customer-relations@sei.cmu.edu