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

Girish Seshagiri

Systems & Software Technology Conference

May 18, 2011

Salt Lake City

Trademarks And Service Marks

The following are service marks of Carnegie Mellon University.

- CMMI℠
- Team Software Process℠
- TSP℠
- Personal Software Process℠
- PSP℠

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 Management℠

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)

ais

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

ais

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

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

ais

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

Software Components - 1

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

Components are what developers build

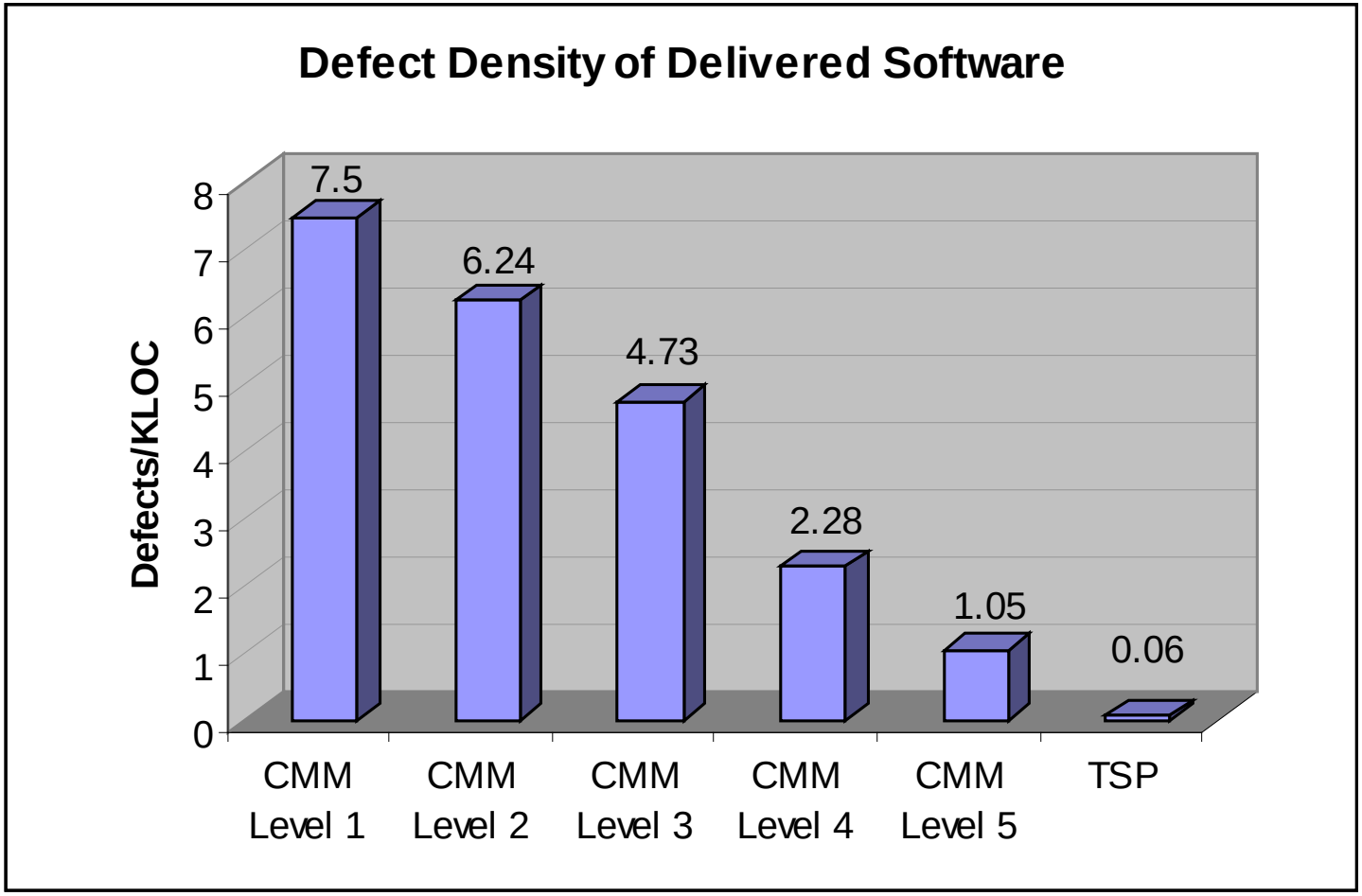
Error-prone components account for a disproportionate number of defects found in integration, system and user acceptance tests

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
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

ais

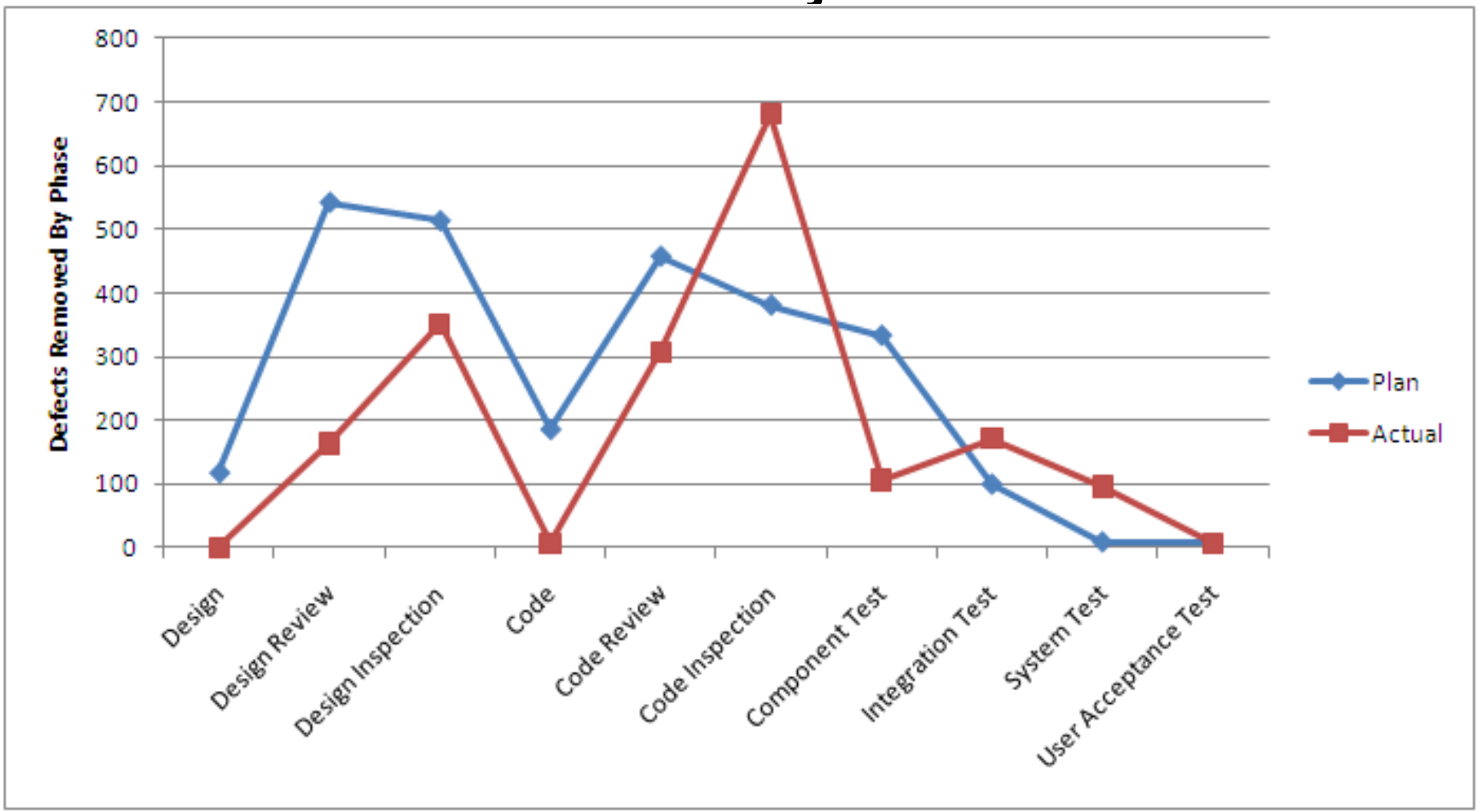
Performance Metrics That Matter
Early Defect Removal

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 into test

Performance Metrics That Matter
AIS Federal Project Results



ais

Performance Metrics That Matter
Process Quality Index

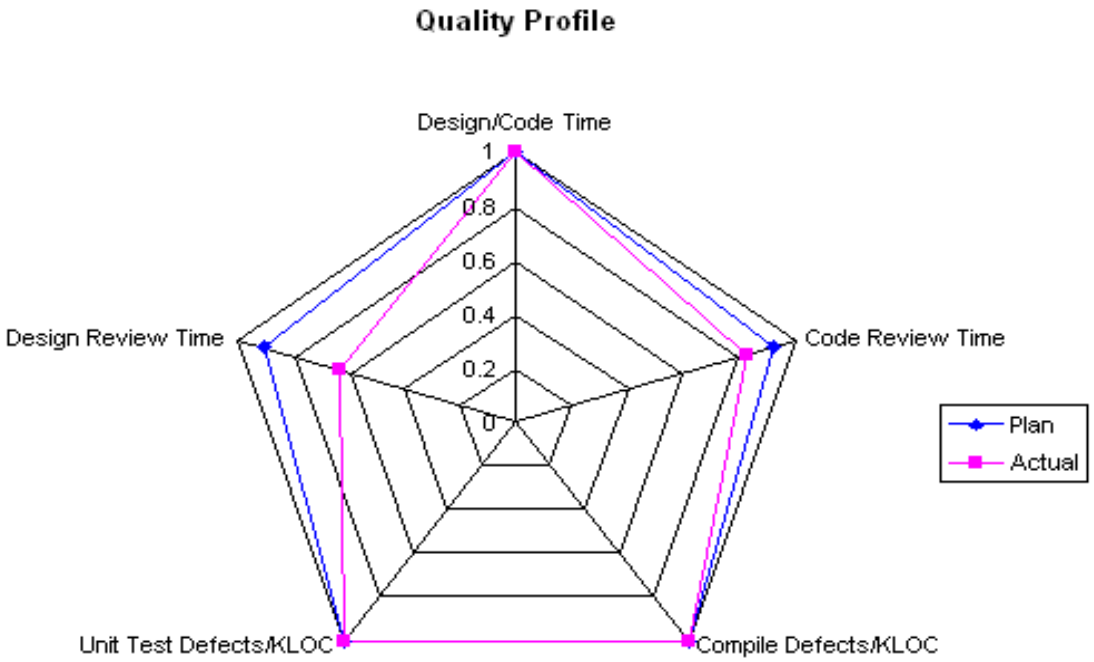
PQI is a leading indicator of overall product quality

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

ais

Performance Metrics That Matter
Component Quality Profile

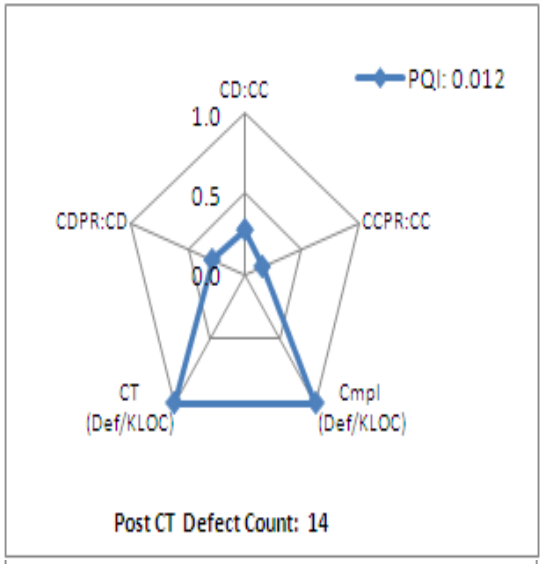


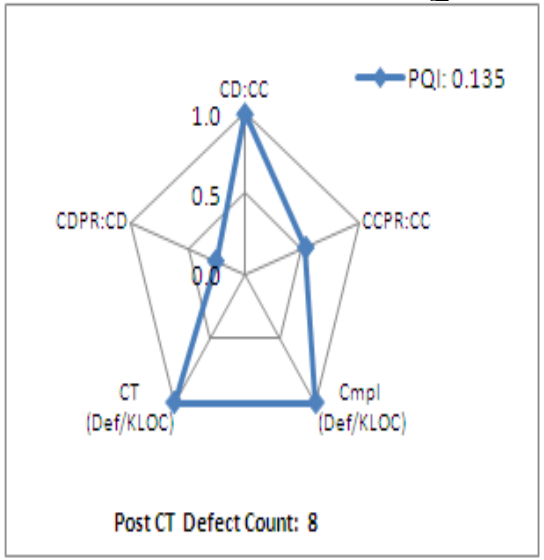
The above chart shows our 5 measures to achieve our quality goals:

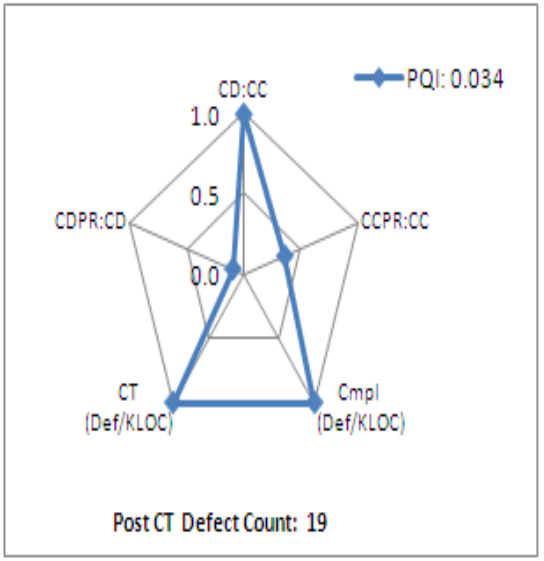
1) Design time to Code time comparison
2) Design Review Time as a % of design time (should be 50% or greater)
3) Code Review Time as a % of code time (should be 50% or greater)
4) Compile Defects per KLOC (Compile should find less than 10 defects per KLOC)
5) Unit Test Defects per KLOC (UT should find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

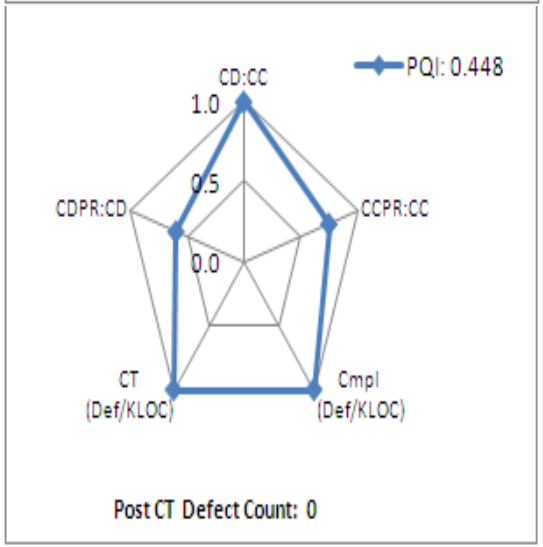
ais

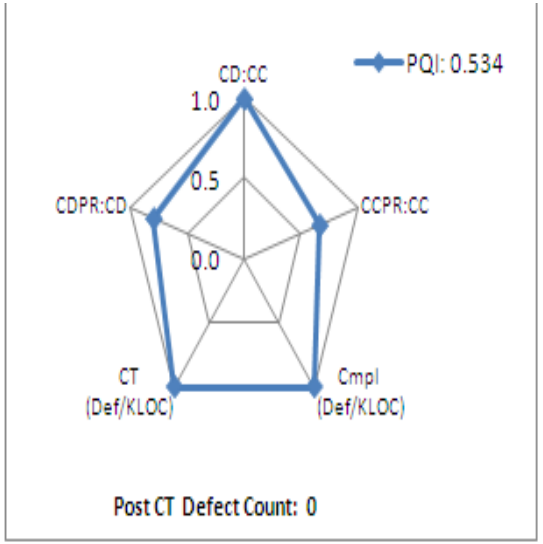
Performance Metrics That Matter
AIS Federal Project Results

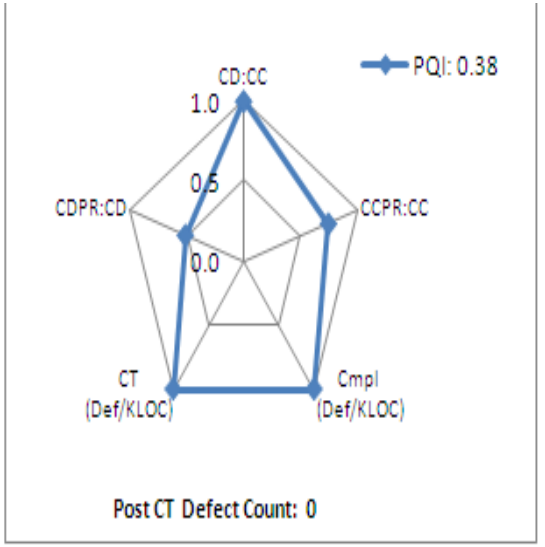












Performance Metrics That Matter
Percent Defect Free Components

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

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

Schedule Compression and Defects

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 Defects

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

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”

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