How To Talk To A Program Manager

10th Annual Systems Engineering Conference
October 22-25, 2007

John Mishler
Anita Carleton
Bill Nichols

© 2007 Carnegie Mellon University

You can’t always get what you want . . .

You can’t always get what you want,

But if you try sometimes, well, you just might find,

You get what you need!

(M. Jagger/K. Richards: 1968)

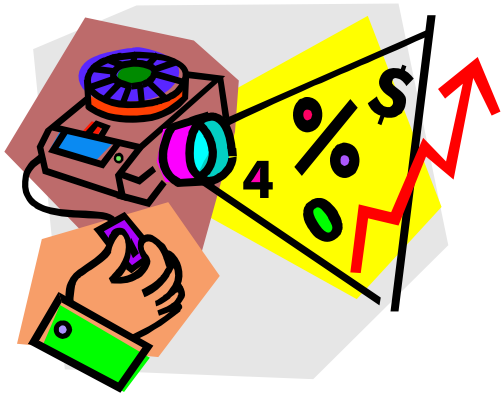
© 2007 Carnegie Mellon University

What Does a Program Manager *Want*?

A high quality product delivered within cost and schedule.

© 2007 Carnegie Mellon University

A Program Manager Also *Needs*…



Accurate, high-quality data and information to keep the program or project on-time and on-budget.

© 2007 Carnegie Mellon University

Topics

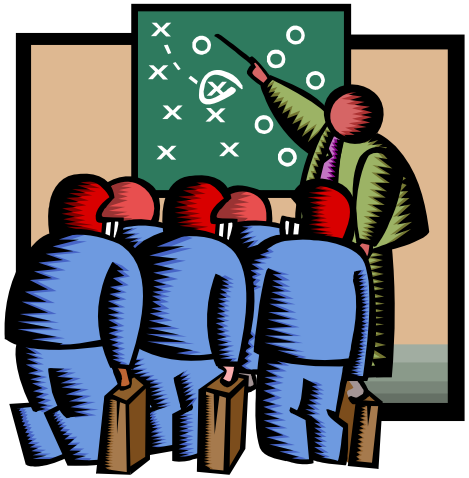
- Key Systems Development and Program Management Issues
- Metrics for Managing Cost and Schedule Issues
- Project Planning Principles and Application
- Project Tracking Principles and Application
- The Way Forward . . . *Team Process Integration*

© 2007 Carnegie Mellon University

Key Issues in Managing Projects/Programs - 1

Management may not be used to working with data . . .

Management doesn't know what to ask for . . .



© 2007 Carnegie Mellon University

Key Issues in Managing Projects/Programs - 2

Different levels of management & needs for data/information

- Frequency and type
  - Project
    - cross-project reviews
    - Quarterly
    - program/milestone
- Organizational Level
  - First level management: team level
  - Second level: middle management—whoever the team reports to; responsible for half dozen projects; matrix management issues
  - Third level: organization level, business managers, people who manage the budget and people who manage the process

© 2007 Carnegie Mellon University

Key Issues in Managing Projects/Programs - 3

Acquisition

- What did you ask for in the RFP or task order?
- Contractor and sub contractor relationships & reporting

Data analysis, summary & presentation

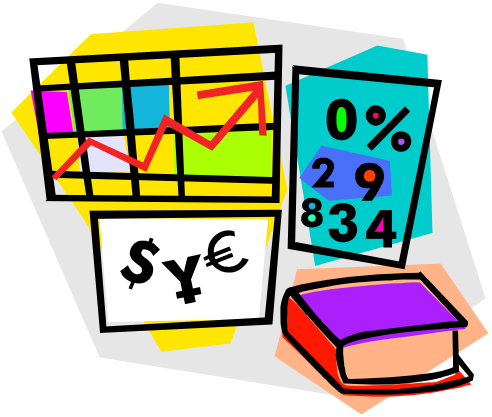
- Dashboard/scorecard
- Stoplight charts
- *Better* (e.g., PQI, PDF, defect densities, customer satisfaction . . .)
- *Faster* (e.g., test time reduction, rework time, ratio of development time to total time, …)
- *Cheaper* (e.g., cost in FTE months or $$ to produce 1,000 task hours of work, …)

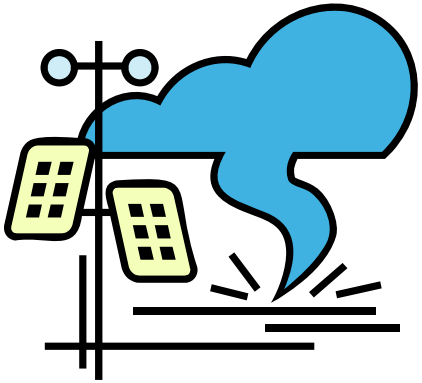
© 2007 Carnegie Mellon University

What Data Should be Collected, Analyzed, and Presented?

Primarily status/performance data and information:

- Where are you in your program/project?
- How do you know?
- Show me!
- EARLY WARNING!





Project/Program Performance

- Schedule
- Quality
- Cost
- Features
- Risks

© 2007 Carnegie Mellon University

Other Questions . . .

What is the availability of resources for a particular project?

What is the productivity?

How good is my organization?

What is the estimation accuracy?

What is the cost to develop x, y, z?

How large are our products?

How are we performing (size, effort, schedule, quality, …)?

© 2007 Carnegie Mellon University

All Projects Have Cost and Schedule Issues

- Most projects are underestimated, usually by a lot, only few are close.
- Many project plans are missing an unknown, some unknowns are disasters, and all will have impact.
- Many projects are late before they start, some know it, and some don't.

All projects have issues. The question is:

"How will you manage the trouble you're in?"

© 2007 Carnegie Mellon University

Cost and Schedule Trouble

When the program manager wants to know . . .

"What is the status of your project?"

"How do you know?"

"Show me!"

. . . How will you respond?

It's a question of how to

- identify the real issues and alternatives
- define a realistic course of action
- take it

© 2007 Carnegie Mellon University

Topics

Key Systems Development and Program Management Issues

➡ Metrics for Managing Cost and Schedule Issues

Project Planning Principles and Application

Project Tracking Principles and Application

The Way Forward . . . *Team Process Integration*

© 2007 Carnegie Mellon University

Conventional Wisdom

Advice is plentiful and full of good intent. So what's wrong with the following guidelines?

Program Management Critical Success Factors	
Cost and schedule estimates are realistic and current.	
Demonstrable Indicators	**Possible Sources of Evidence**
Does the scope of the estimate cover all activities throughout the system life?	Costed Through-Life Management Plan.
Were independent methods used to produce and check cost and schedule estimates?	Estimates available for reviewer.
Have the estimates been independently validated?	Independent Technical Review, Quality Assurance Review.
Are there mechanisms to monitor and adjust assumptions and estimates?	Process documentation / Quality System

© 2007 Carnegie Mellon University

Partial List of Answers . . .

- Lagging indicators
- Metrics disassociated from work and those performing it
- Latency in reporting
- Mechanism to adjust estimates and replan over the life of the project?
- How can we do better next time?

© 2007 Carnegie Mellon University

SEI Core Metrics

Size

Effort

Quality

Schedule

© 2007 Carnegie Mellon University

Estimated and Actual Size

What PM's Should Expect

Size is a measure of the magnitude of the deliverable

- for software: lines of code or function points
- for other intellectual work: document pages, presentation slides, computer tests or simulations, etc.

Size is estimated and reported for each component.

The actual size is measured and reported for each component.

Size data are used to

- estimate effort
- normalize other measures
- track progress

Intellectual work can be measured according to accounting categories.

- Base <measure>
- Reused <measure>
- Added <measure>
- Modified <measure>
- Deleted <measure>

© 2007 Carnegie Mellon University

Estimated and Actual Effort

What PM's Should Expect

Effort is a measure of time "on task."

The recommended effort measure is a *task hour.*

Task hours should be measured and reported by

- WBS task
- process phase
- project week

How many task hours are there in a 40 hour week?

About 15 to 20!

© 2007 Carnegie Mellon University

Estimated and Actual Schedule

What PM's Should Expect

Schedule shows the work completed or planned to be completed in a given period.

Schedule accuracy depends on granularity of tasks.

Recommended task granularity

- From 10 to 20 hours for near-term tasks
- From two to four weeks for remaining tasks.

Estimated and actual task dates are reported.

© 2007 Carnegie Mellon University

Estimated and Actual Quality

What Acquirers Should Expect

Defects are a measure of quality.

Definition: when the product must be changed to ensure proper design, implementation, test, use, or maintenance, a defect has been identified.

Estimates of the number of defects injected and removed in each phase and component should be reported.

Actual defects found in each component, and each phase, and the phase in which the defect was injected should be reported.

© 2007 Carnegie Mellon University

Derived Measures

The core metrics support many derived measures that can be used by the acquirer to evaluate plans and track program status.

Examples include

- planned value, earned value, and predicted earned value
- estimation accuracy for size, time, and quality
- time in phase distribution
- defect density
- defect removal rate
- review rates
- process and phase yield
- cost of quality
- percent defect-free
- defect removal profiles
- quality profile and process quality index

© 2007 Carnegie Mellon University

Topics

Key Systems Development and Program Management Issues

Metrics for Managing Cost and Schedule Issues

➡ Project Planning Principles and Application

Project Tracking Principles and Application

The Way Forward . . . *Team Process Integration*

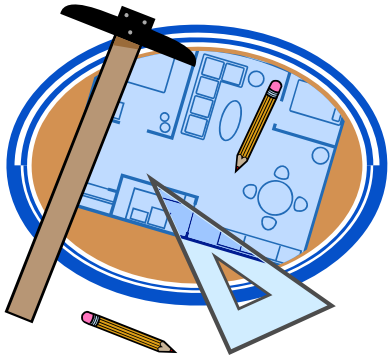
© 2007 Carnegie Mellon University

Project Plans

A Program Manager should expect to find the following:

- Most project plans are underestimated.
- Many project plans are missing the unknowns.
- Many projects are late before they start.

The problem is that many project plans ignore well-known principles of project planning and are therefore un-executable.



© 2007 Carnegie Mellon University

Project Planning Guidelines

Projects are dynamic, so are good project plans.

Accuracy improves with detail, so make detailed plans and estimates; use historical data.

Quality must be managed by the numbers, so make a quantitative quality plan.

The best plans are made by the people assigned to do the work.

© 2007 Carnegie Mellon University

Dynamic Plans

If you can't make accurate plans, plan often.

Many projects (especially software projects) are subject to considerable change.

Every change will affect the cost and schedule.

Every cycle will include new information.

Projects should produce new plans every few months.

- a detailed near-term plan
- a high-level plan to the end of the project

© 2007 Carnegie Mellon University

Detailed Plans

Detailed plans are more accurate.

They are more likely to be complete.

There is greater opportunity for over-estimates and under-estimates to cancel.

Detailed plans provide

- deeper insight into the work
- better guidance during execution
- more accurate tracking
- basis for getting out of trouble

© 2007 Carnegie Mellon University

Detailed Estimates

Like detailed plans, detailed estimates are more accurate for the same reasons.

There are many estimating methods available.

The best methods

- use historical data
- have a feedback loop
- are systematic methods or processes that reduce bias and can be improved
- are appropriate for, and calibrated to, the work

© 2007 Carnegie Mellon University

Estimating Bias

Most estimates tend to be optimistic.

The key to accurate estimates is not the accuracy of each estimate.

The key to accurate estimates is detail and consistency.

To improve consistency, the engineer must reduce estimating bias.

To reduce estimating bias the developer should

- make size estimates first
- base effort on size
- next estimate resources
- use estimated effort and resources to create a schedule

© 2007 Carnegie Mellon University

Quality Principles

Many intellectual products ignore quality until final review (or test).

You can't test in quality; if you try to you will

- increase cost and schedule by 40% to 50%
- make the plan even more unpredictable
- pay the contractor for avoidable rework
- field a defective product

To get a quality product out of test, you must put a quality product into test.

To put a quality product into test you must measure and manage quality at each step of the process.

© 2007 Carnegie Mellon University

Quantitative Quality Plans

Quality without numbers is just talk.

If you want a high-quality product you must ask for quality data.

To do this you need to know

- what quality data to ask for
- how to evaluate the quality data you're given

The core metrics are a good start.

- defects injected and removed
- by phase and component
- both estimated and actual

© 2007 Carnegie Mellon University

Quality Benchmarks (software example)

These quality benchmarks are derived from the SEI's TSP research for software projects.

They are based on results obtained from software project teams and individual developers.

They can be adapted to evaluate or track other kinds of project plans.

Sample TSP Benchmark Data	
Measure	**Benchmark**
Review rate	200 LOC/Hr.
Code injection rate	2/Hr.
Code review removal rate	4/Hr.
Code inspection removal rate	1/Hr.
Code review and inspection yield	70%
Unit test removal rate	3/Hr.
Unit test yield	50%
Integration and System Test removal rate	>10 Hr. per defect
Integration and System Test yield	< 50%

© 2007 Carnegie Mellon University

Evaluating Plans -1

The objective is to determine if the plan represents a reasonable commitment.

Is it complete, detailed, accurate, robust, and capable of producing a quality product?

Size estimate

- Was size estimated?
- How were the estimates made?
- Are the size estimates detailed?
- Was historical data used?
- Who made the size estimates?

© 2007 Carnegie Mellon University

Evaluating Plans -2

Effort estimate

- Was effort estimated?
- How were the estimates made?
- Are the efforts estimates based on size?
- Are the efforts estimates detailed?
- Was historical data used?
- Who made the effort estimates?

Quality plan

- Does the plan include estimates of defects injected and removed by phase?
- How were these estimates made?
- Are they based on either size or effort?
- Was historical data used?
- Who made the estimates?
- How is the effort distributed? (gets to quality and rework)

© 2007 Carnegie Mellon University

Probing Questions for a Plan Review

What Acquirers Should Look For

1. What are the size estimates for the principal products and how did you arrive at these estimates?

2. How do you know if you have included all of the necessary tasks?

3. What is the basis for your task estimates?

4. How did the team arrive at the schedule?

5. How did you estimate the number of weekly task hours?

6. What could management do to help maximize your weekly task hours?

7. Have you looked at alternative plans?

8. Could you accelerate the schedule, and if so, what would it cost?

9. What is your quality plan and how did you produce it?

10. What is your security plan and how did you produce it?

11. What are the key risks and your mitigation plans for them?

12. Have you considered prototyping or reuse?

13. What is the likely error in the size, time, and schedule estimates?

14. When will you have learned enough to make a more accurate plan?

© 2007 Carnegie Mellon University

Topics

Key Systems Development and Program Management Issues

Metrics for Managing Cost and Schedule Issues

Project Planning Principles and Application

➡ Project Tracking Principles and Application

The Way Forward . . . *Team Process Integration*

© 2007 Carnegie Mellon University

Commitment

Intellectual product development is a team sport.

Motivated teams are more productive and produce better products.

When teams are empowered to make their own plans they are motivated to meet their commitments.

Costs and schedules are established before the project is staffed and are difficult to change.

The question is, how early do you want to know about cost and schedule trouble?

© 2007 Carnegie Mellon University

Getting Out of Trouble

Projects don't get behind overnight, they slip day by day.

If you manage the days, the weeks, months, and years will take care of themselves.

The key is early warning.

- The review period must be short.
- Detailed weekly data are needed.

The most common causes of schedule slip

- estimation error (size, effort, hours available per week)
- quality problems

© 2007 Carnegie Mellon University

Early Warning

Projects get in trouble for many reasons.

What indicators provide early warning of cost, schedule, or quality problems?

- earned value progress tracking
- resource tracking
- software
  - defect removal profile
  - percent defect-free

© 2007 Carnegie Mellon University

Earned Value Progress Tracking

The SEI recommends the use of *earned value* for tracking project progress.

Earned value (EV) is a method for tracking progress based on the contribution that each task makes towards completing the project.

EV provides a precise way to

- measure project status
- estimate project completion dates

© 2007 Carnegie Mellon University

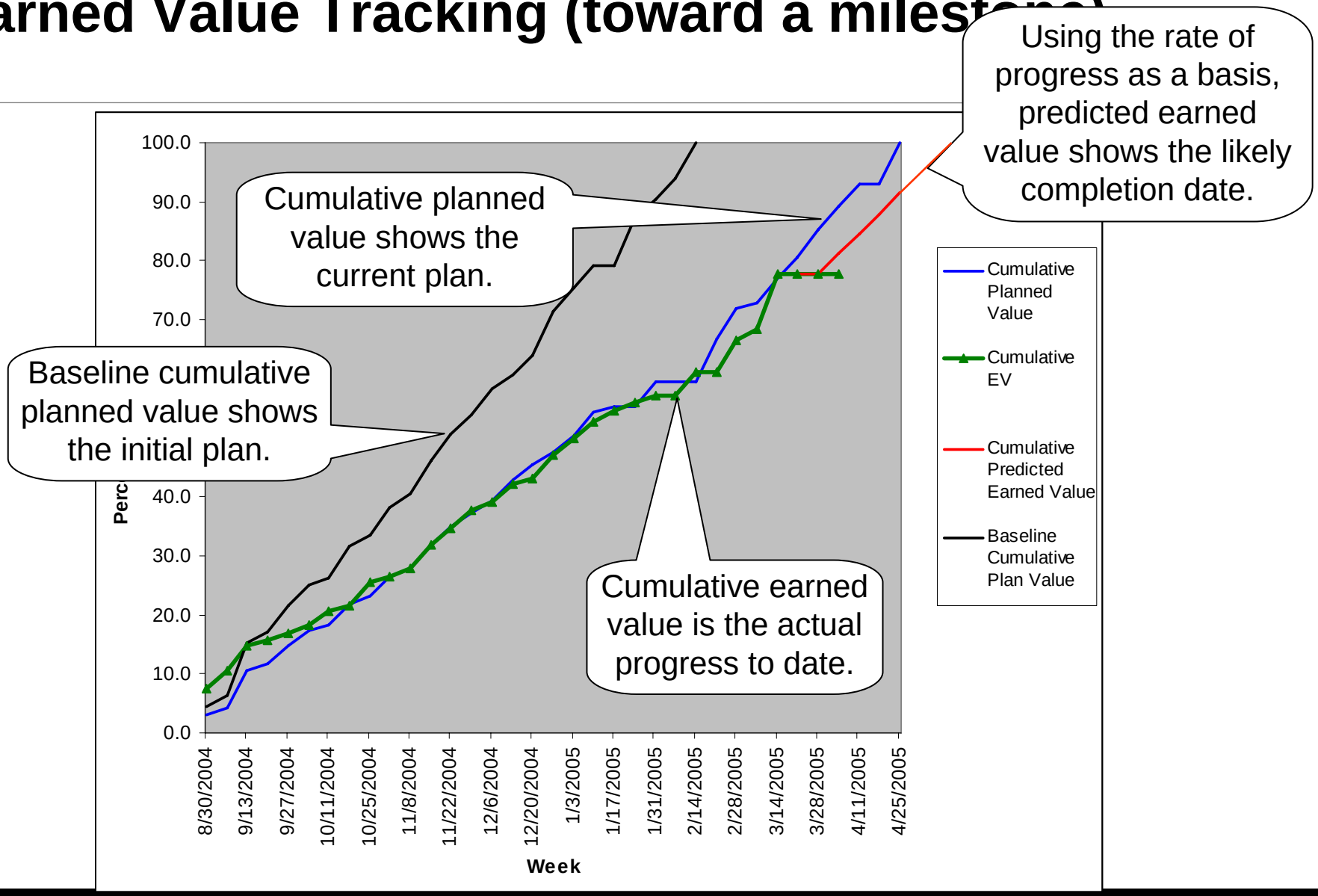
Earned Value Guidelines

The SEI recommends the use of the following earned value guidelines.

- A task's EV is the task's estimated hours divided by the total estimated hours for all tasks in the project.
- The project earns the value for a task when it is completed, there is no partial credit.
- The EV credit is the same, regardless of the time the task actually takes.

© 2007 Carnegie Mellon University

Earned Value Tracking (toward a milestone)



© 2007 Carnegie Mellon University

EV Tracking Considerations

Causes of EV slip

- difference in estimated vs. actual effort
- difference plan vs. actual resource hours
- blocked tasks

EV limitations

- assumes future is like the past
- insensitive to future estimation error or resource deltas
- poor quality in test can be a significant impact

EV must be applied consistently across projects and programs!

© 2007 Carnegie Mellon University

Status Tracking at a granular level

Here's an example of how teams can track their status weekly using a defined process and a weekly status summary in a support tool.

TSP Week Summary - Form WEEK

Name	Carol	Date	4/7/2003
Team	PSP Ghost		
Status for Week	15	Cycle	
Week Date	3/10/2003		

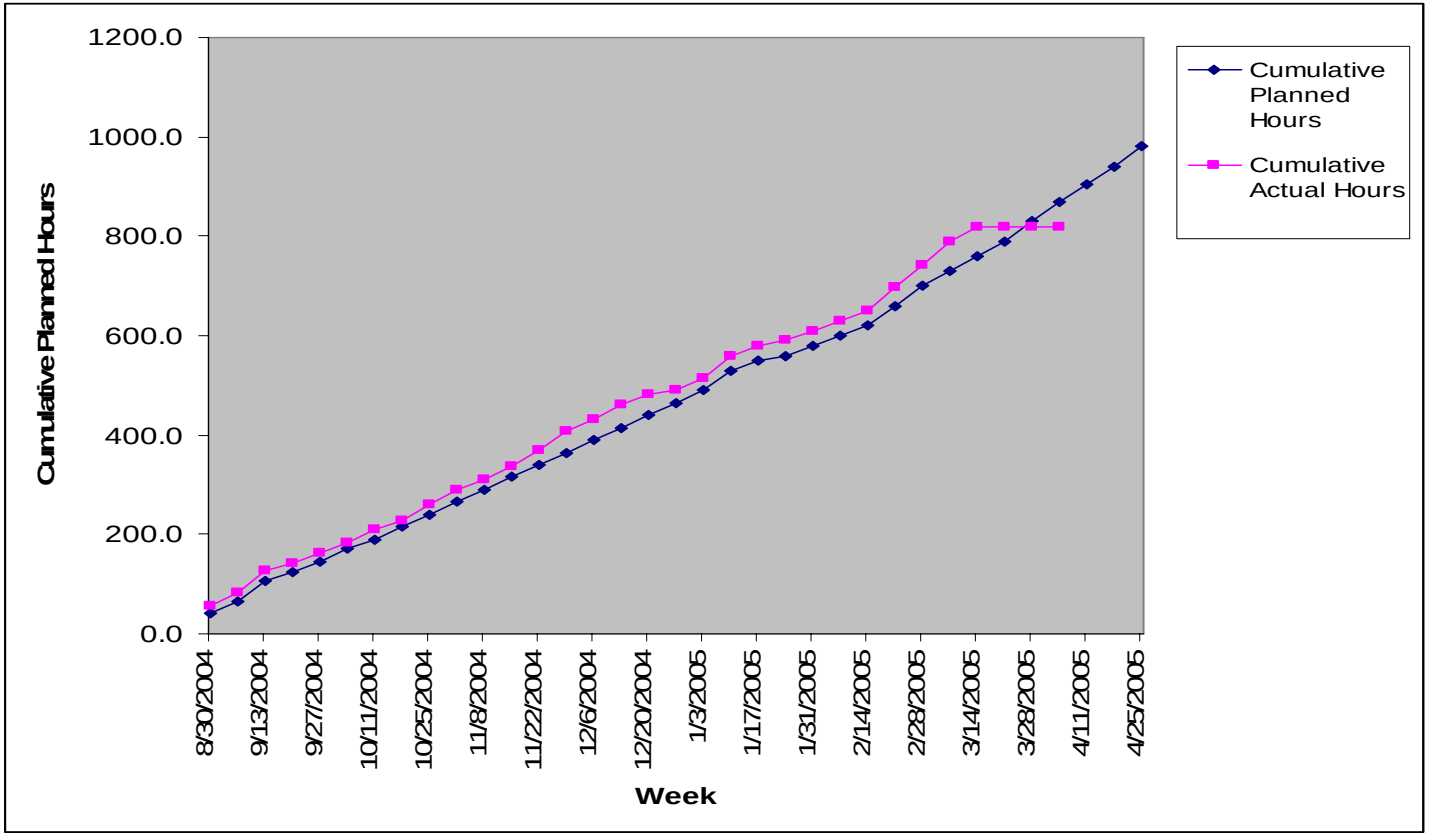
Weekly Data	Plan	Actual	Plan/ Actual
Schedule hours for this week	151.0	86.0	1.76
Schedule hours this cycle to date	1526.0	1594.8	0.96
Earned value for this week	6.9	4.2	1.64
Earned value this cycle to date	79.5	84.3	0.94
To-date hours for tasks completed	1580.7	1568.1	1.01
To-date average hours per week	101.7	106.3	0.96

Assembly	Phase	Tasks Completed or Due	Resource	Task Plan Hrs.	Task Actual Hrs.	Earned or Plan Value	Planned Week	Plan vs. Actual Hrs.
Main Form	CODEINSP	Main Form Code Inspection	SA	1.5	2.4	0.1	10	0.63
OEMMOO Delivery.aspx	UT	OEMMOO Delivery.aspx (FE-Server) U	NK	8.9	3.0	0.5	13	2.91
OEMMOO Delivery.aspx	DLDINSP	OEMMOO Delivery.aspx (FE-Client) DI	NK	0.0	0.0	0.0	13	
OEMMOO Delivery.aspx	CODE	OEMMOO Delivery.aspx (FE-Client) C	NK	7.5	5.7	0.4	14	1.32
OEMMOO Delivery.aspx	CR	OEMMOO Delivery.aspx (FE-Client) C	NK	3.8	1.7	0.2	14	2.26
OEMMOO Delivery.aspx	COMPILE	OEMMOO Delivery.aspx (FE-Client) C	NK	1.3	0.9	0.1	14	1.44
OEMMOO Delivery.aspx	CODEINSP	OEMMOO Delivery.aspx (FE-Client) C	NK	0.0	0.0	0.0	14	
OEMMOO Delivery.aspx	UT	OEMMOO Delivery.aspx (FE-Client) U	NK	5.9	6.8	0.3	14	0.87
Query Object	TD	Query Object Test Development	MB	0.0	0.0	0.0	14	
Query Object	CODEINSP	Query Object Code Inspection	MB	0.0	1.2	0.0	14	0.00
Query Object	UT	Query Object Unit Test Dialogs	MB	1.1	1.7	0.1	14	0.66

© 2007 Carnegie Mellon University

Resource Management

Here's another example of how teams can track resource utilization weekly using a defined process and a weekly status summary in a support tool.



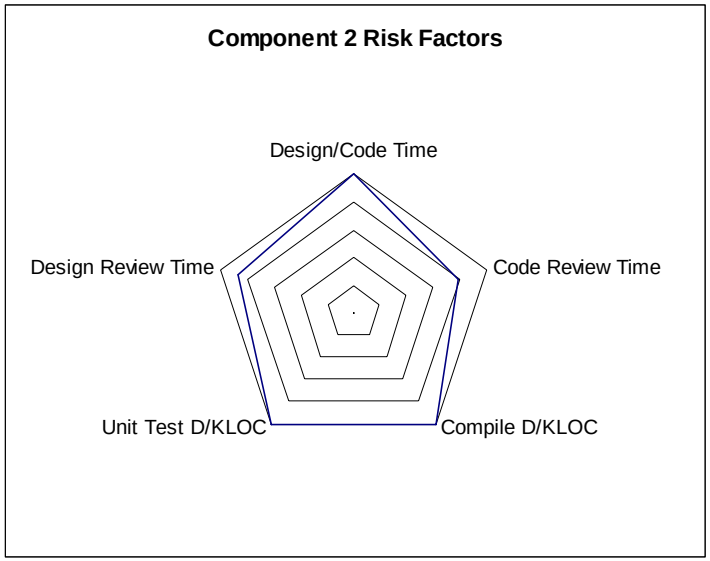
© 2007 Carnegie Mellon University

Quality Management

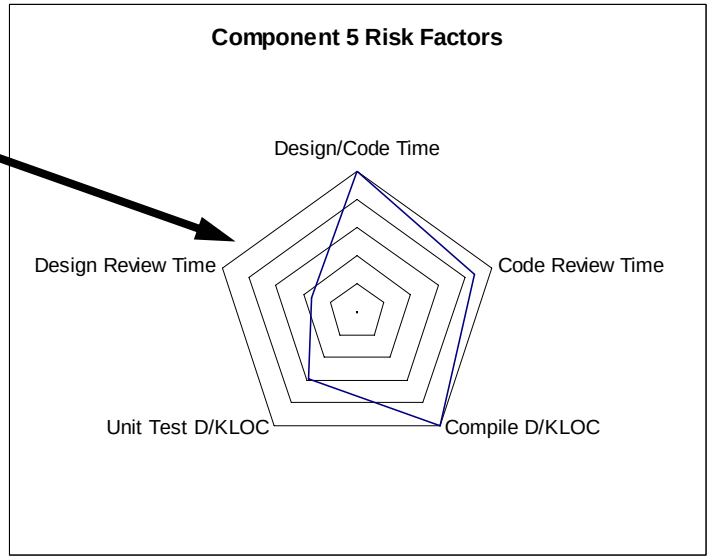
Here's another example of how Software Development teams can use a Quality Profile as an early warning indicator of post-development defects.

The quality profile uses five software quality benchmarks.

Satisfied criteria are plotted at the outside edge of the chart.



Inadequate design review time results in design defects escaping to test and production.

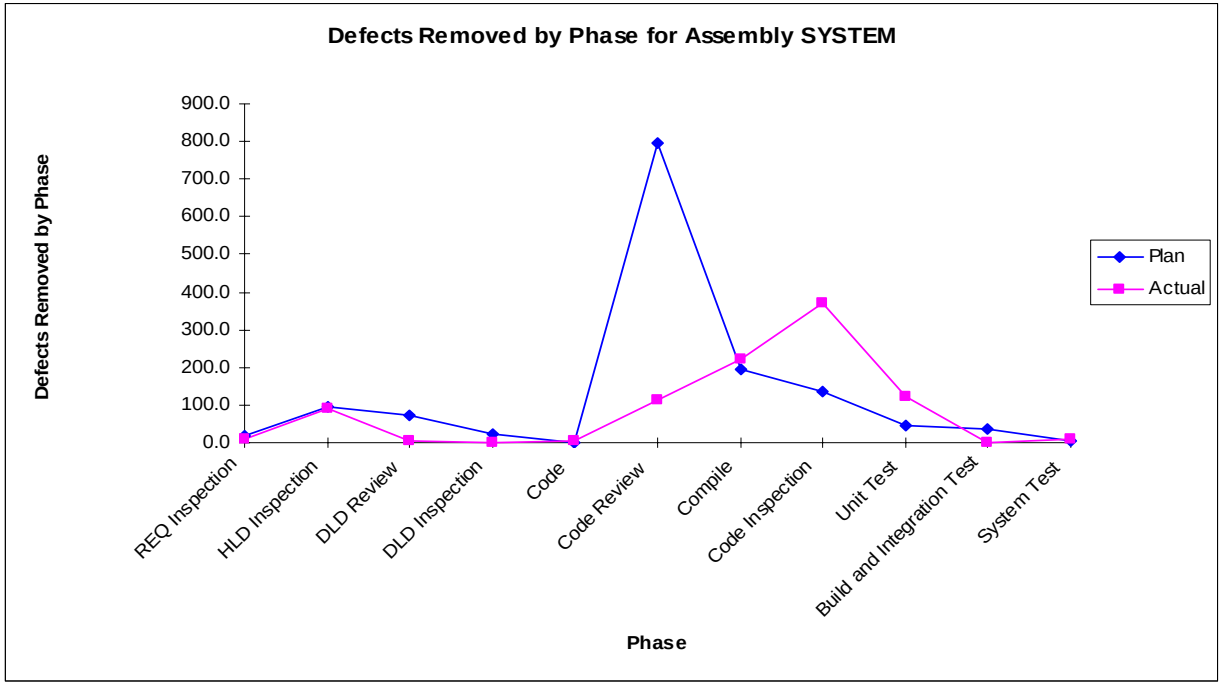


© 2007 Carnegie Mellon University

Defect Removal Profile

Again, using a Software Development example, a Defect Removal Profile may be used to track

- plan and actual defects removed by phase
- early vs. late defect removal plan

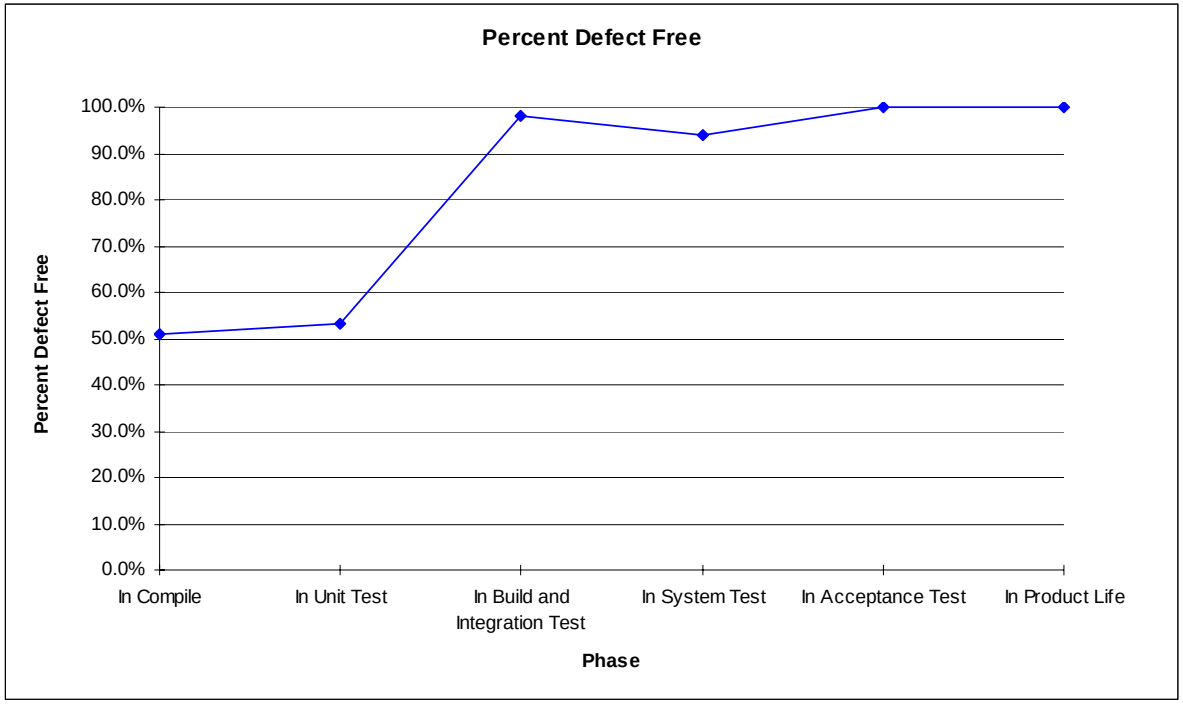


© 2007 Carnegie Mellon University

Percent Defect Free

Similarly, teams can use a Percent Defect-Free profile to track

- percentage of components that are defect-free in test
- pre-test process quality



© 2007 Carnegie Mellon University

Probing Questions for a Status Review

What Acquirers Should Look For

1. What percentage of the project has been completed based on earned value?

2. How does the earned value compare to planned value?

3. What is the end date based on predicted earned value?

4. How do planned resource hours compare to actual resource hours?

5. How do actual hours for completed tasks compare to estimated hours for completed tasks?

6. What is the gap between actual resource hours and actual hours for completed tasks?

7. How does this gap compare to the average actual hours per week?

8. How do size estimates compare to the actual size of completed components?

9. What is the correlation between actual size and actual hours for completed components?

10. What is the correlation between the planned size and actual hours for completed components?

11. How do planned defects compare to actual defects for completed phases and components?

12. What is the actual review rate for inspections and reviews?

13. What percentage of components are defect-free at unit test, integration test, system test, …?

14. How does the actual defect removal profile compare to plan for completed components?

© 2007 Carnegie Mellon University

Topics

Key Systems Development and Program Management Issues

Metrics for Managing Cost and Schedule Issues

Project Planning Principles and Application

Project Tracking Principles and Application

➡ The Way Forward . . . *Team Process Integration*

© 2007 Carnegie Mellon University

What Should Be the Way Forward?

Alternatives should address deficiencies in “conventional wisdom”

- Based on data collected as work is done
- Used to change development behavior
- Predictive metrics, not just reactive (leading vs. lagging)
- Using proven, repeatable methods
- Incorporating planning and tracking principles we have just discussed
- Addressing institutionalizing change

If we don’t address these issues, we are doomed to repeat the mistakes of the past . . .

© 2007 Carnegie Mellon University

What Alternatives Exist?

One example is the Team Software Process (TSP), an integrated set of practices for developing software.

TSP is an process-based solution to common software engineering and management issues.

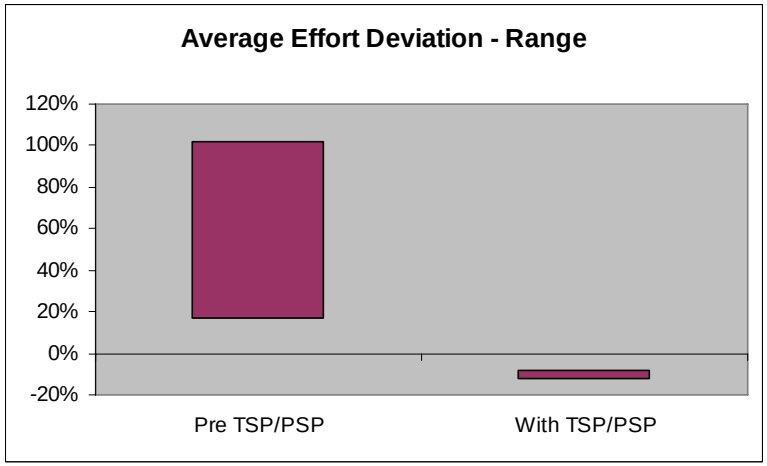
- cost and schedule predictability
- productivity and product quality
- process improvement

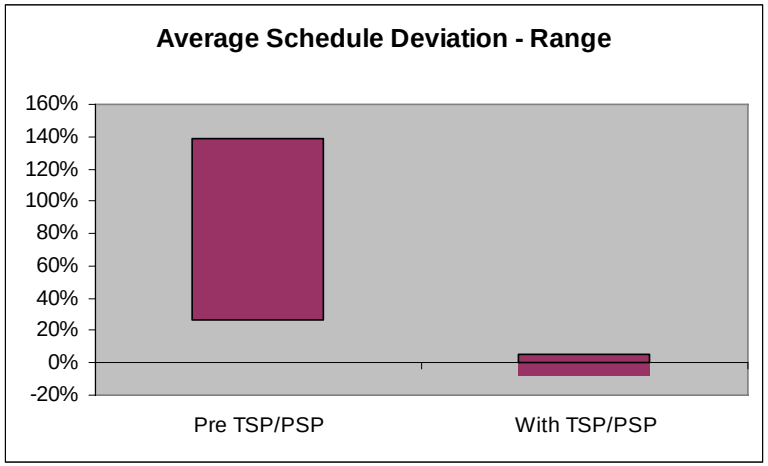
Unlike other methods, TSP

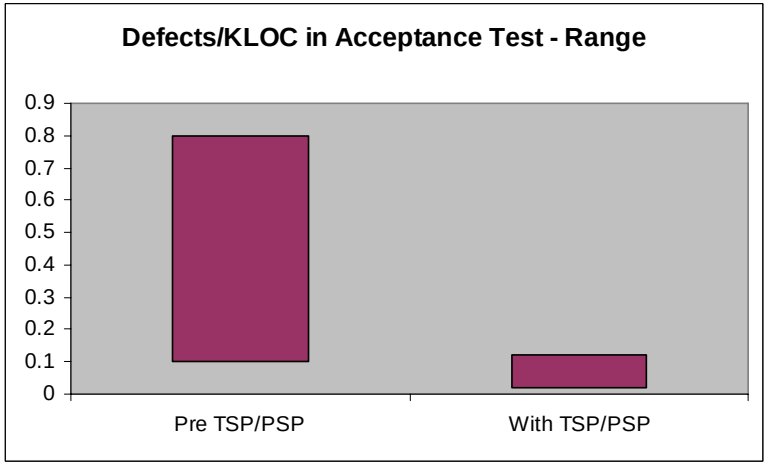
- teams are self-directed.
- emphasizes measurement and quality management.
- provides immediate and measurable benefits.
- accelerates CMMI-based improvement.

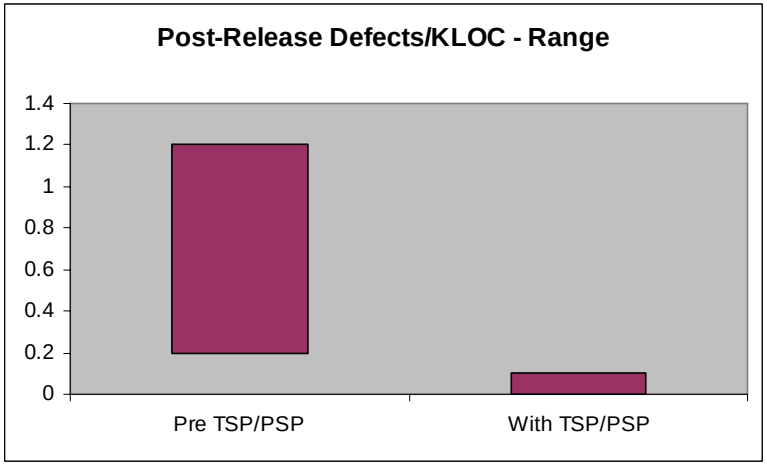
© 2007 Carnegie Mellon University

TSP Improves Performance









Source: CMU/SEI 2000-TR-015

© 2007 Carnegie Mellon University

TSP Performance Comparison

Performance Category	Typical Industry Performance[1]	Study Baseline (2000) [2]	TSP Impact Study (2000)[3]	TSP Impact Study (2003)[4]
Schedule Error	180% avg.	27% to 112%	5% avg.	6% avg.
Effort Error	130% avg.	17% to 85%	-4% avg.	26% avg.
System Test defects per KLOC	5 to 25 approx.	NA	0.0 to 0.9	0.4 avg. 0.0 to 0.9
Released defects per KLOC	1 to 7	0.2 to 1+	0.0 to 0.35	0.06 avg. 0.0 to 0.2

1. Gartner Group
2. Control projects from CMU/SEI-2000-TR-015, a study of 18 TSP projects in four organizations conducted in 2000.
3. TSP projects from CMU/SEI-2000-TR-015, a study of 18 TSP projects in four organizations conducted in 2000.
4. TSP projects from CMU/SEI-2003-TR-014, a study of 20 projects in 13 organizations conducted in 2003.

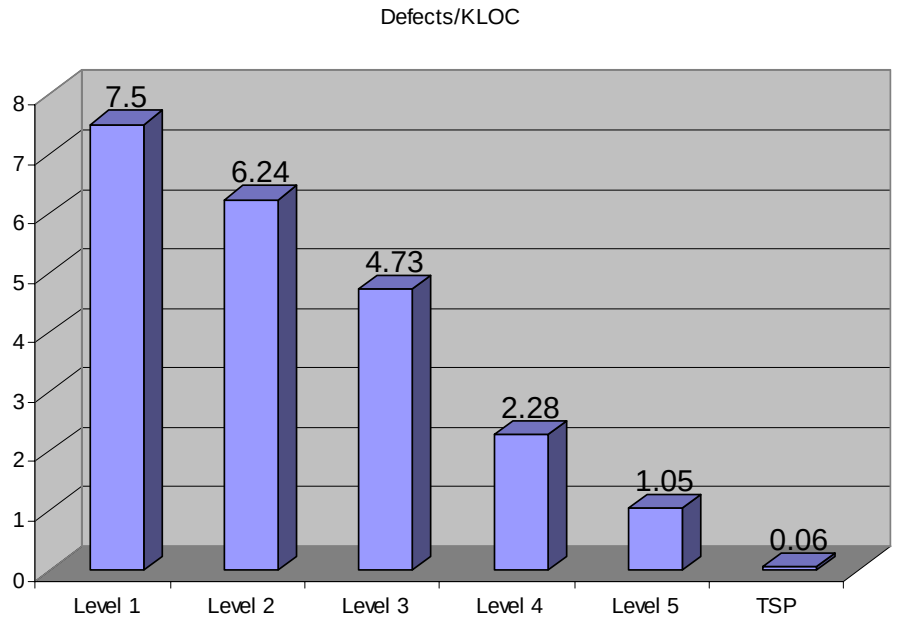
© 2007 Carnegie Mellon University

TSP Improves Quality

An analysis of 20 projects in 13 organizations showed TSP teams averaged 0.06 defects per thousand lines of new or modified code.

Approximately 1/3 of these projects were defect-free.

These results are substantially better than those achieved in high maturity organizations.



© 2007 Carnegie Mellon University

TSP Principles

Software is developed by individuals and teams. To improve overall performance, improve individual and team performance.

Empowered teams, coaching, and rational management are the most effective way to manage engineering teams.

Process is a performance determinant…an ad-hoc chaotic process cannot produce predictable, high-quality results.

Measurement is necessary for high-performance.

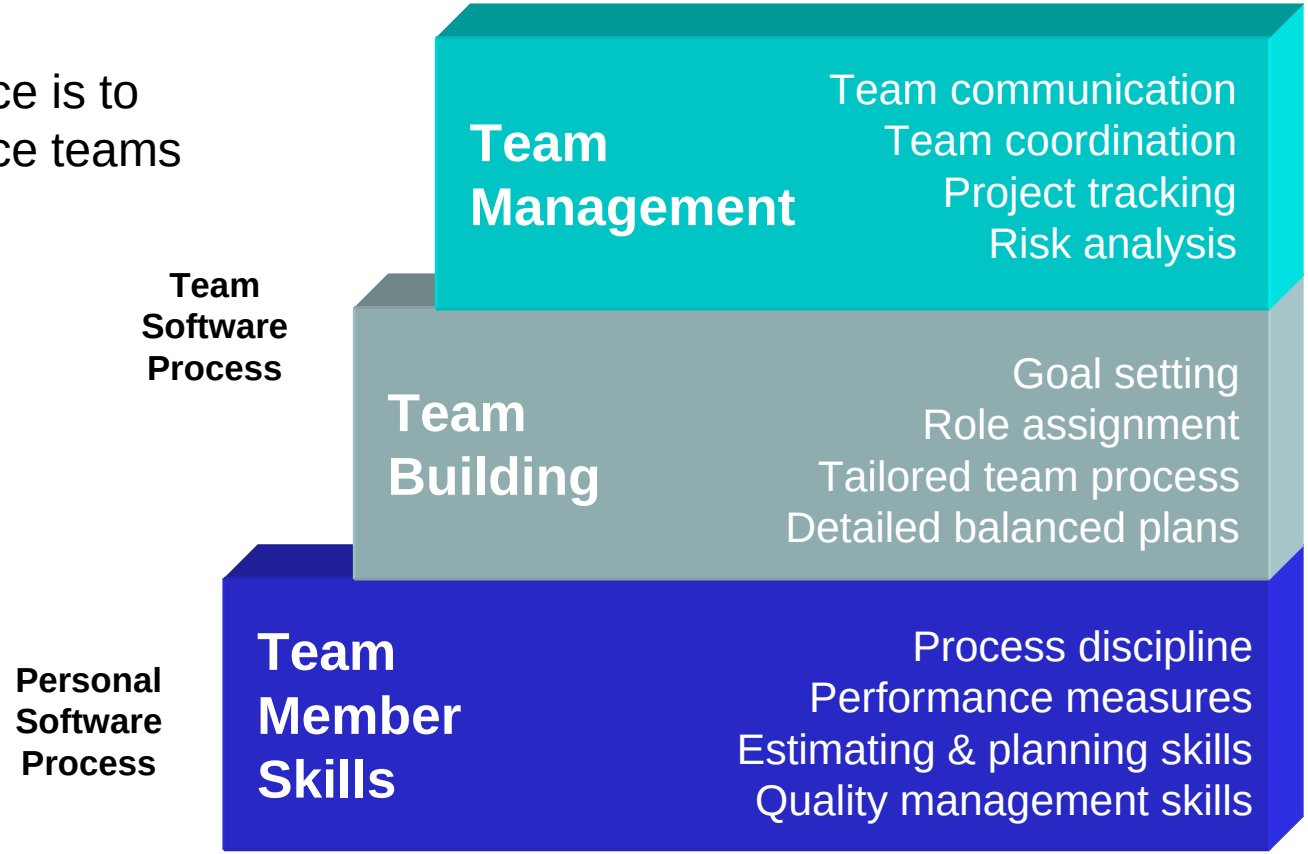
To get a quality product out of test, you must put a quality product into test.

Quality must be measured and managed at every step.

© 2007 Carnegie Mellon University

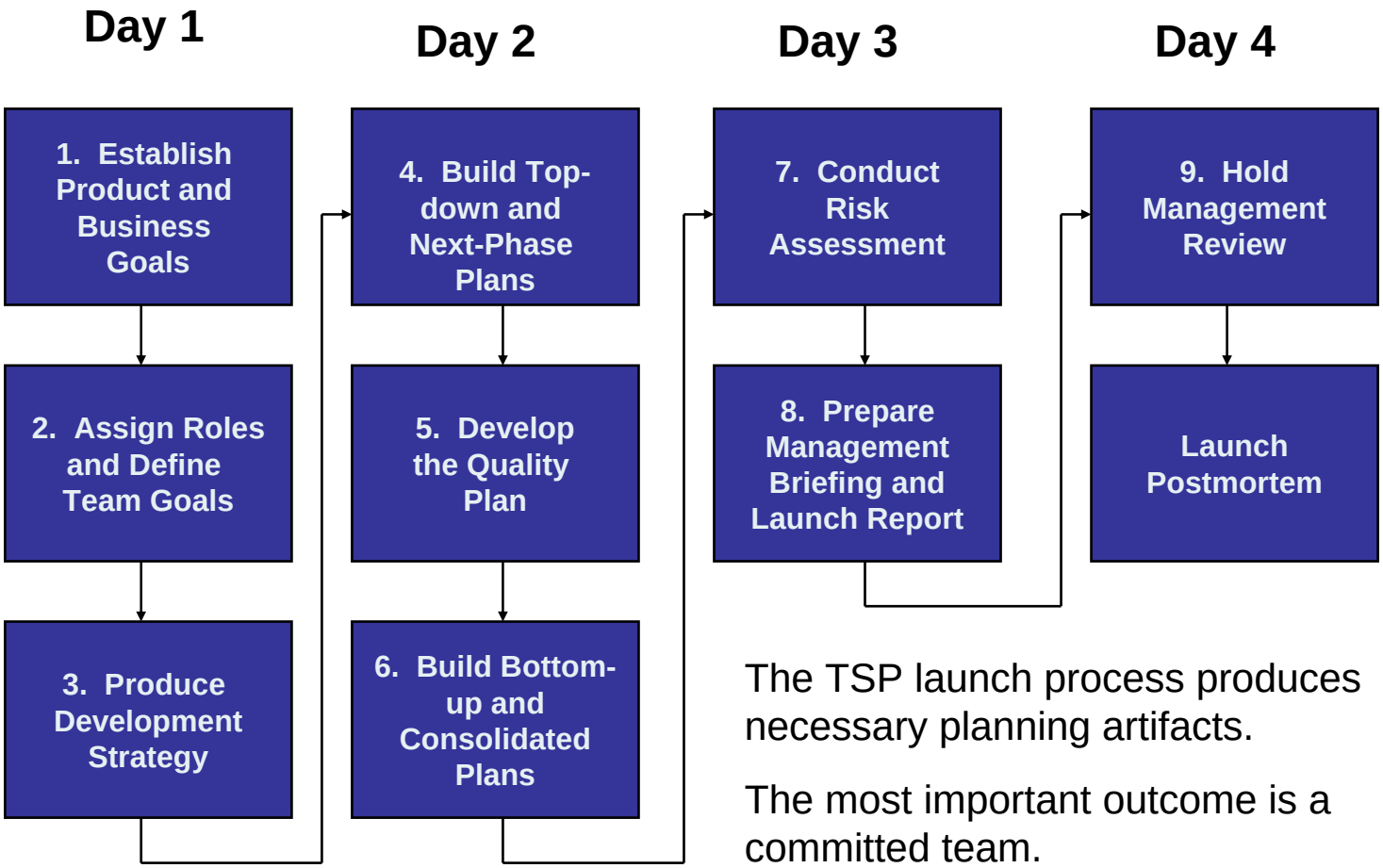
Building High-Performance Teams

The TSP strategy for improving performance is to build high-performance teams from the bottom-up.



© 2007 Carnegie Mellon University

The TSP Launch Process



The TSP launch process produces necessary planning artifacts.

The most important outcome is a committed team.

© 2007 Carnegie Mellon University

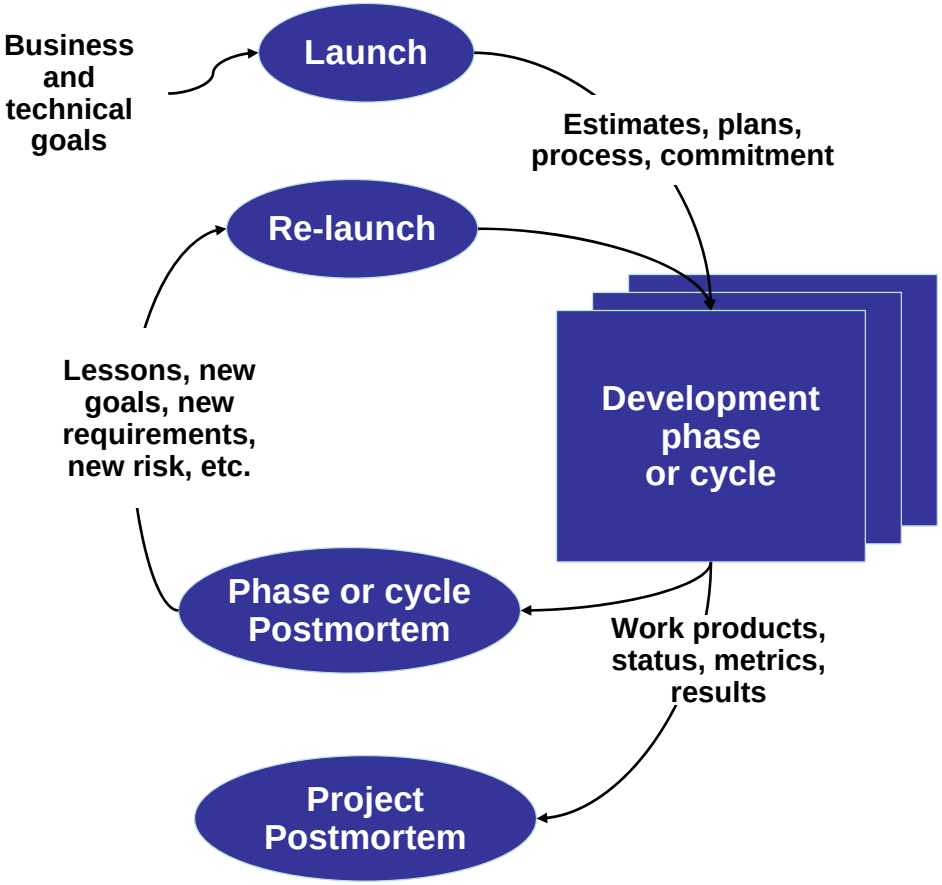
TSP Process Structure

The TSP process elements can be organized into whatever process structure makes the most business and technical sense.

The phases can be implemented iteratively in small cycles, in a spiral with increasing cycle content, or sequentially as in a waterfall,

TSP projects can start on any phase or any cycle.

Each cycle starts with a launch or re-launch and ends with a postmortem.



© 2007 Carnegie Mellon University

Reasons for extending the TSP Method to the Systems Engineering Domain

Cost-benefit numbers exist from TSP on software projects

Requests from existing product teams using TSP to extend to their Systems Engineering (and other) process areas

Increasingly difficult to solve software problems without addressing systems engineering issues

© 2007 Carnegie Mellon University

Extending TSP Method to the SE Domain—Team Process Integration (TPI)

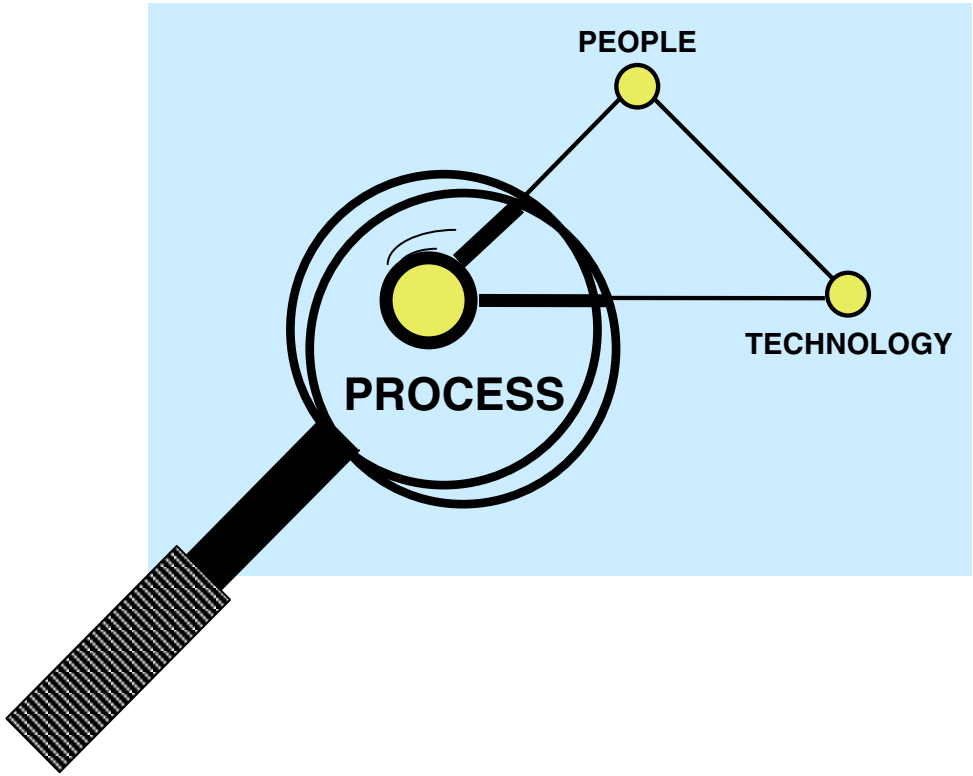
How do we tailor TSP for other, non-software disciplines?

The research challenges include:

- Processes
- Measurement
- Role Definition
- Training
- Tool Support

© 2007 Carnegie Mellon University

Research Challenges - Processes



- Develop prototype processes/scripts for SE and other engineering disciplines

- Use “traditional” TSP Launch Process

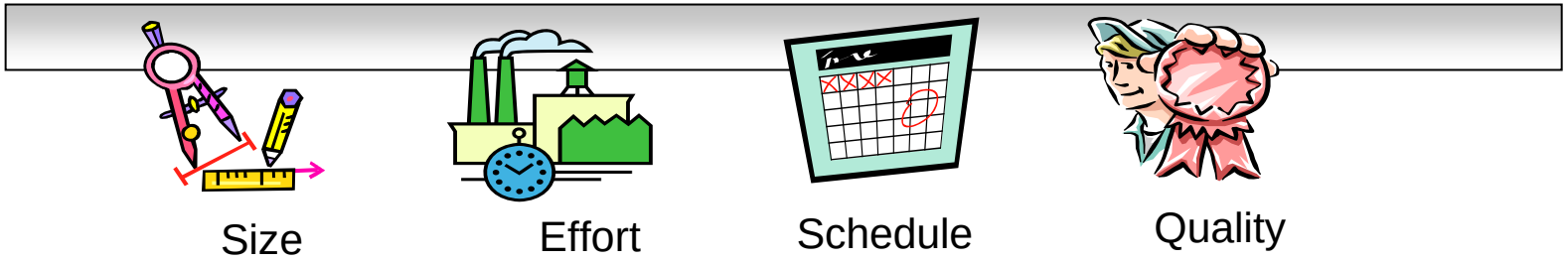
© 2007 Carnegie Mellon University

Research Challenges – Measurement - 1

Schedule and effort measures are essentially unchanged.

Lines of Code/Function Points would not serve as relevant size measures for SE. Formulate size measures for SE. Examples:

- DOORS objects
- Requirements
- Items Verified



© 2007 Carnegie Mellon University

Research Challenges – Measurement - 2

Quality measures in SE

- Define what “quality” means in SE
- Where in the process do you collect data?
- What are the derived quality measures (e.g., defects/DOORS object?)
- Establish initial quality baseline

© 2007 Carnegie Mellon University

Research Challenges – Measurement - 3

What are the quality goals?

Examples:

- Goal: Accuracy in the work

  Measure: # of problem reports against requirements and test documents

- Goal: Conformance to standards

  Measure: # of defects in peer reviews; # of defects in requirements and test documents, etc…

© 2007 Carnegie Mellon University

Research Challenges – Role Definition

Apply four primary roles—planning, process, quality, support

Assess applicability of remaining roles and define additional roles needed for SE.

- Add Requirements Manager
- Combine Design and Implementation roles into one role
- Expand Test Manager role as needed, e.g.,
  - Flight Test Manager
  - Lab Test Manager

© 2007 Carnegie Mellon University

Research Challenges – Training - 1

Currently, training is geared to software teams.

So challenges include:

- building conviction and discipline in teams that don't write software programs
- providing just the right amount of training to get a team started
- supplementing with additional training modules as needed

© 2007 Carnegie Mellon University

Research Challenges – Training - 2

Develop “JIT” training to support SE teams

Develop Leadership Seminar and Team Member Training to focus on:

- providing the fundamentals of TPI
- launching a team
- maintaining a plan

Follow-up with additional, “JIT” training, e.g.,

- Inspections
- Measurement, data analysis, and reporting
- Checkpoints and Postmortem Analysis
- Tool

© 2007 Carnegie Mellon University

Research Challenges – Support Tool

Develop an extensible tool that allows for:

- Defining any process
- Collecting data unobtrusively
- Defining a measurement framework

© 2007 Carnegie Mellon University

TPI Project Goal

Extend TSP method and practice to other domains

- Demonstrate that distributed teams using
  - **Tailorable** process scripts
  - Metrics
  - Automated tool support

can deliver **quality** products and services on **cost** and **schedule**

© 2007 Carnegie Mellon University

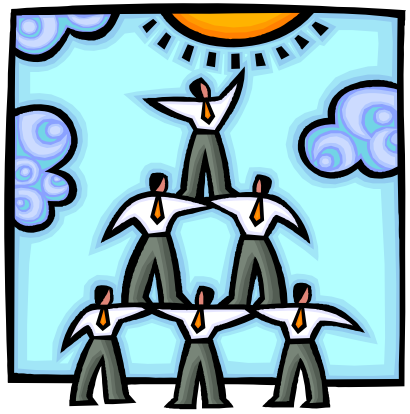
TPI Fundamentals

To be successful, teams need

- clear goals and roles
- agreed-upon processes and plans

Build self-directed teams

- team members own their own processes and plans
- team members understand their goals and roles
- team members voluntarily commit to their goals
- team members believe their commitments are achievable



The TPI launch builds self-directed teams.

Team members of self-directed teams practice disciplined personal processes.

TPI data rolls up to the project and program level.

© 2007 Carnegie Mellon University

TPI Systems Engineering Pilots

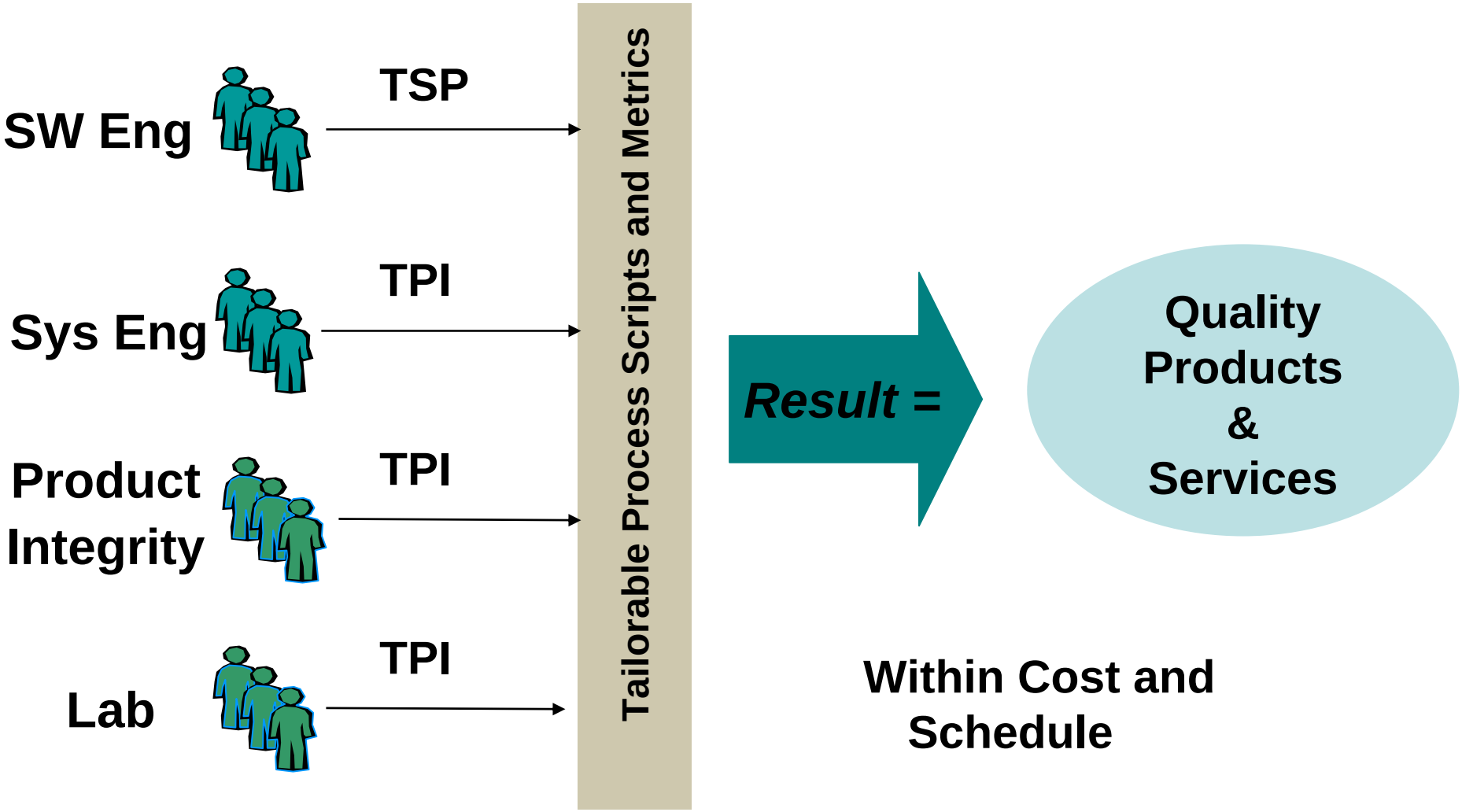
Conduct a series of pilot projects to determine if extending TSP practices to Systems Engineering and other engineering disciplines results in measurable improvement

- Engineering Discipline A
- Engineering Discipline B
- Systems Engineering/Integration
- Product Integrity
- Lab Team
- Test Team

Use these results to establish common processes/measures within and across projects/programs

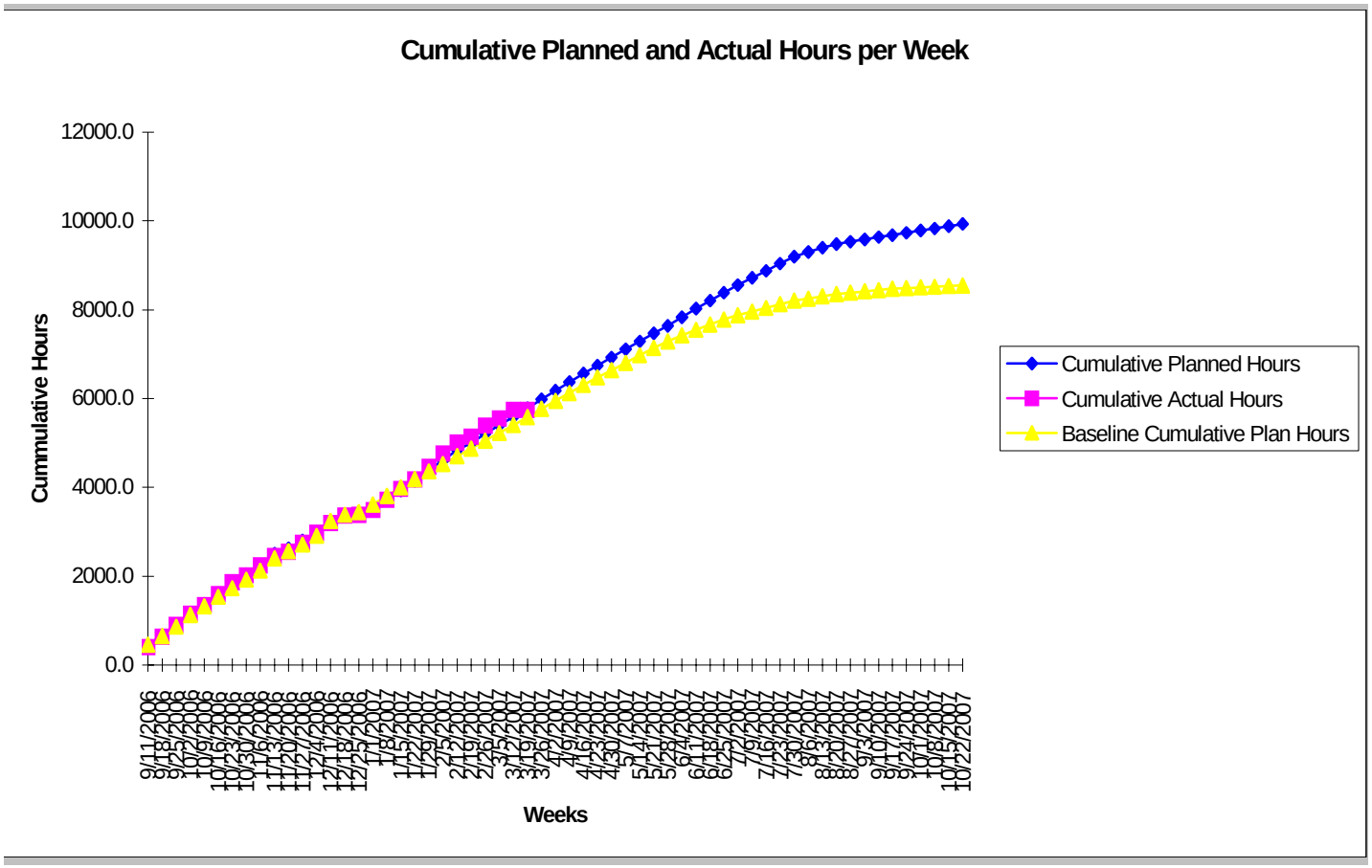
© 2007 Carnegie Mellon University

Pilot TPI Implementation Approach at Org A



© 2007 Carnegie Mellon University

Preliminary Results from Org A



Team has been consistently meeting or exceeding planned task hours (Faltering EV trend is not due to failing to meet task hour commitment.)

© 2007 Carnegie Mellon University

Org A Qualitative Benefits - 1

High value in launch process – management communicating plans and priorities, team developing plans, coordinating & integrating across program/project teams

Tracking the work is helping team members to stay on track

- "I need to stop doing X to get back on track."
- Very easy to see the impact daily and weekly of not working to the plan
- Realization that one can't keep doing extra tasks without impact to the plan

© 2007 Carnegie Mellon University

Org A Qualitative Benefits - 2

Team meeting—weekly coordination/communication with team members and off-loading/balancing of tasks

Team members functioning as a self-directed team by taking on team management roles--Planning Manager, Process Manager, Test Manager, and Lab Manager roles being executed

Better estimates for second launch cycle based on having . . .

- Common processes
- Common language
- Common metrics . . . within team and across teams and the entire program

© 2007 Carnegie Mellon University

Lessons Learned

First Launch – estimates based on time, experience, and engineering judgment

Second Launch – estimates based on data from the first launch and proxies developed for systems engineering work

- Resulted in plan coming in much closer to desired schedule dates
- Better able to define work
- Agile team able to adapt easily to program/personnel changes—i.e., saves the team six months of defining new work for new team member
- Making course corrections as needed

© 2007 Carnegie Mellon University

Team Leader Feedback from Org A

"Prior to TPI, we made estimates in a bubble."

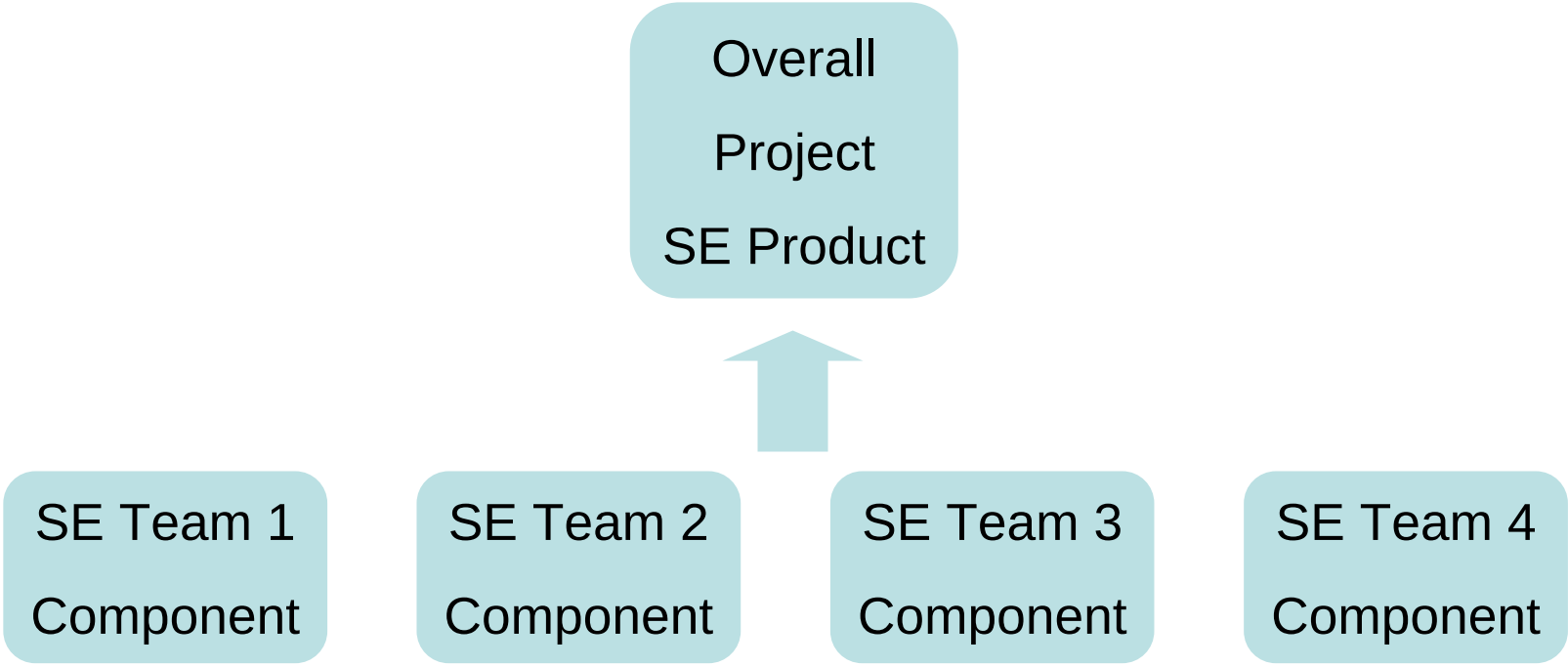
"Now we are establishing and maintaining baselines for all of our releases which allows us to make better estimates and more realistic plans and schedules."

"We have a team concept across our program with all of the subteams (SE, PI, SW, Test, Lab, …). We also have a common set of processes and metrics to help all of the teams better communicate and address dependencies across the teams."

"We are starting to roll-up our team data.
Next, we need to roll-up data across all of the teams in the XYZ Program."

© 2007 Carnegie Mellon University

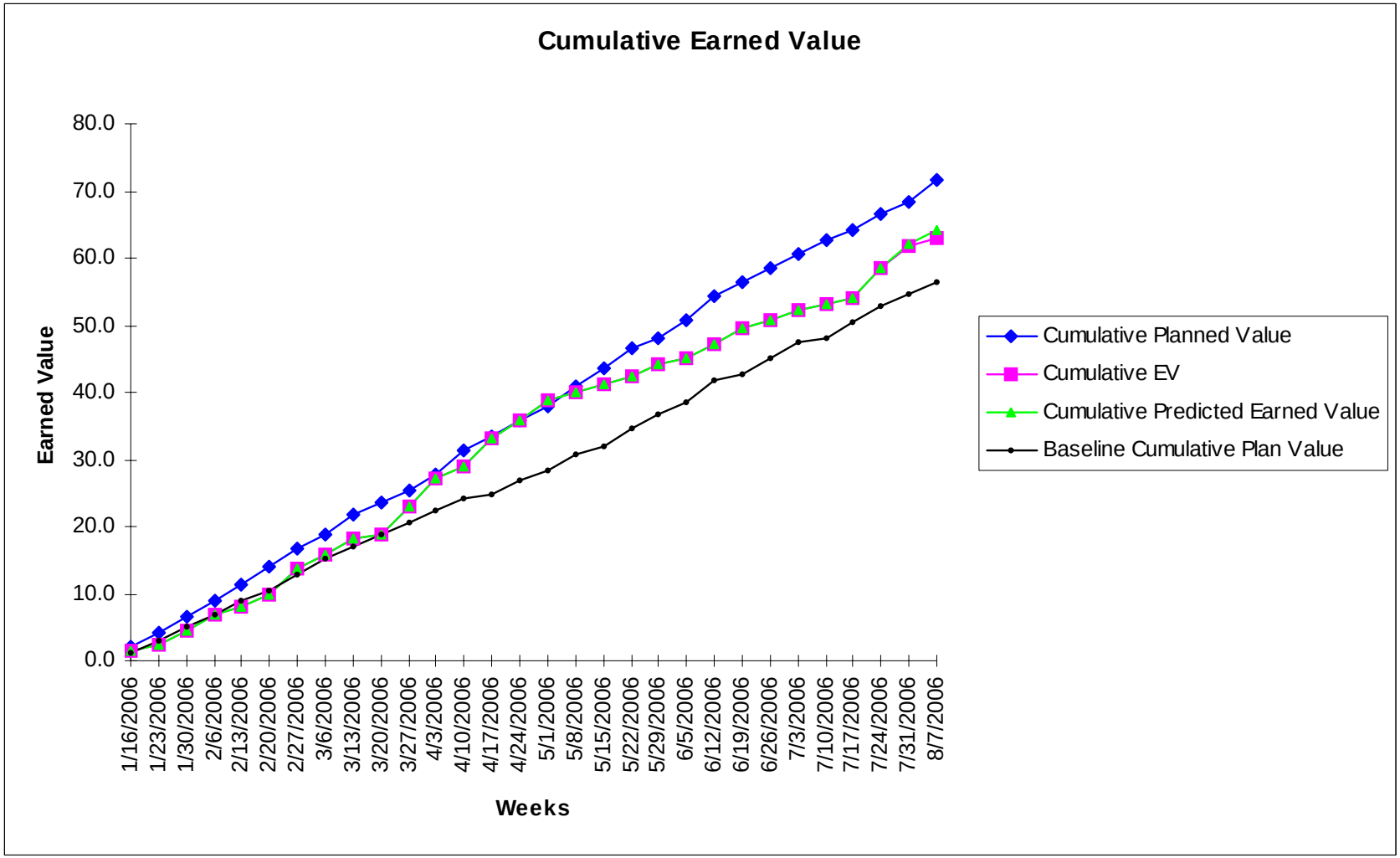
Pilot Approach at Org B



- Using TPI for 2+ yrs
- Extended to 6 teams and growing

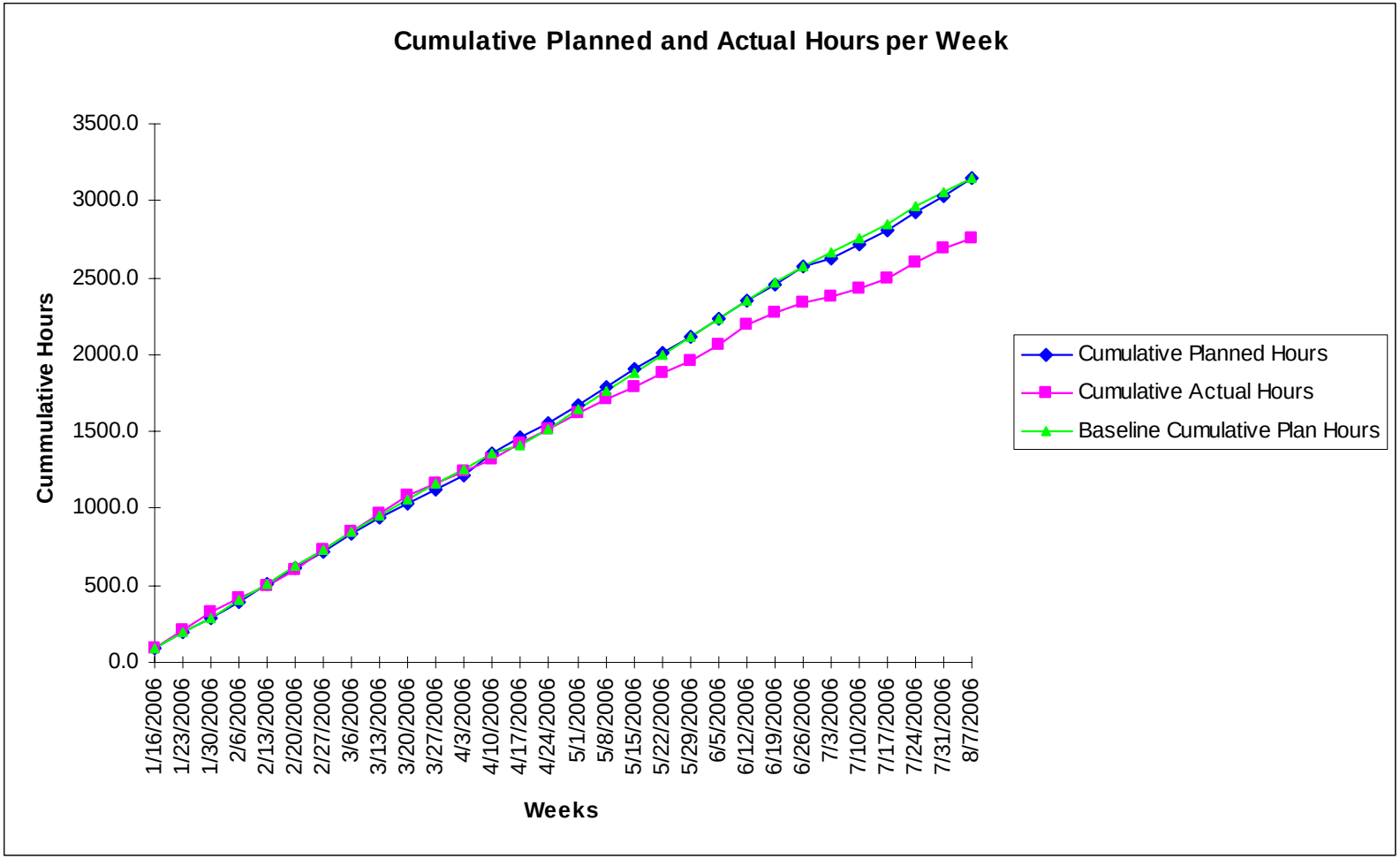
© 2007 Carnegie Mellon University

Preliminary Results from Org B (SE Domain)



© 2007 Carnegie Mellon University

Another example from Org B



© 2007 Carnegie Mellon University

Comments on Previous EV Chart

Schedule performance diverged from the plan at the time this sample was taken.

The team was unable to apply the planned hours.

This often results from resource allocation to other projects.

This information permits the project manager and the program manager to make informed decisions when allocating scarce resources among multiple projects.

© 2007 Carnegie Mellon University

TPI Approach Shows Promise

Eights teams are pilot testing TPI

The launch process is applicable in any engineering domain to:

- produce an executable team plan
- get management's agreement to the plan
- to build an enthusiastic, self-directed team

Provides data that PM's, team leads, team members can use

Provides common language and framework across teams

A support tool is being prototyped

- based on MS Access
- can be adapted to any engineering discipline

Training has been developed and tailored for SE and other teams
(e.g., Senior Leadership Training, Team Member Training, Tool Training)

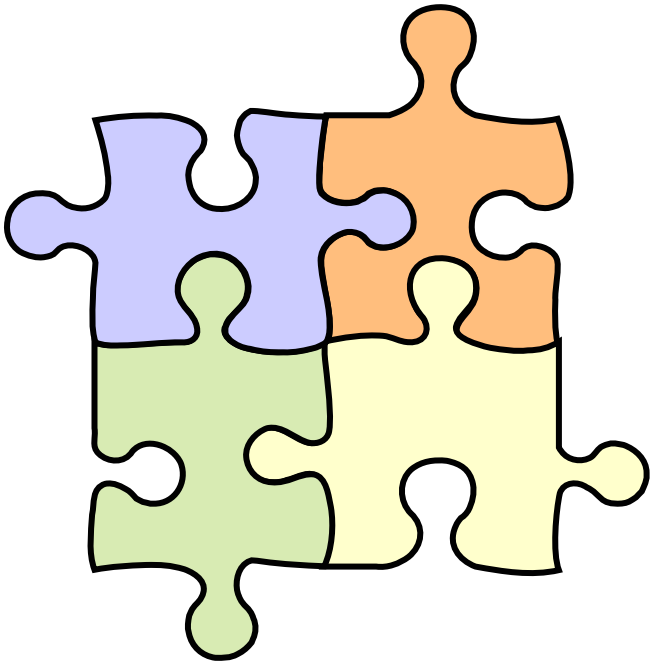
© 2007 Carnegie Mellon University

What makes TPI unique—Putting the pieces together

You may already be using some of the planning, tracking, and quality management practices in the TPI.

What makes the TPI unique is that all these practices are integrated into a simple, operational framework that addresses

- Planning
- Tracking
- Quality
- Measurement
- Tool
- Training
- Team building



And that framework can not only be used at the project level, but at an individual team member level to guide day-to-day work.

© 2007 Carnegie Mellon University

Improving How *YOU* Talk to a Program Manager

Reliable size, effort, cost, schedule, and quality data are required.

You get what you ask for, so know what to ask for and how to use what you get.

Remember—if you try you might just get what you need

Detailed plans that support weekly tracking and reporting of these data are necessary to detect trouble early.

Know how to use the data to identify the real issue, define a realistic course of action, and take it.

Team Process Integration shows promise as means to improve How ***YOU*** Talk to a Program Manager!

© 2007 Carnegie Mellon University

For Further Information . . .
If You Are Interesting in Piloting TPI

John Mishler jmishler@sei.cmu.edu

Anita Carleton adc@sei.cmu.edu

Bill Nichols wrn@sei.cmu.edu

Carnegie Mellon University

Software Engineering Institute

4500 Fifth Avenue

Pittsburgh, PA 15213

© 2007 Carnegie Mellon University