

Analyzing Project Management Indicators



Copyright 2017 Carnegie Mellon University. All Rights Reserved.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material is distributed by the Software Engineering Institute (SEI) only to course attendees for their own individual study.

Except for any U.S. government purposes described herein, this material SHALL NOT be reproduced or used in any other manner without requesting formal permission from the Software Engineering Institute at permission@sei.cmu.edu.

Although the rights granted by contract do not require course attendance to use this material for U.S. Government purposes, the SEI recommends attendance to ensure proper understanding.

DM17-0428



About the Software Engineering Institute (SEI)

DoD R&D Laboratory FFRDC

Sponsored by OUSD (AT&L)

Created in 1984

Under contract to Carnegie Mellon University

Main office in Pittsburgh

Mission: Improve the practice
of software engineering



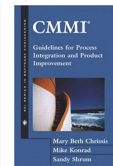
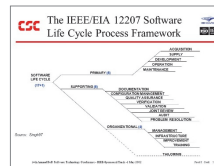
What You Will Learn in this Course

How to

- **analyze** indicators to obtain project insight (status, causes of problems)
- **communicate** objectively via measures
- **identify** possible problems with the measurement system
- **define** indicators/displays that address the things a project manager would want to know

Where measurement is used from the perspective of a generic management process

We will reference models such as IEEE 12207, the CMMI® and the PMBOK®



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

5

Audience



- Project management practitioners and other software managers
- Assumes some prior experience or training in project management



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

6



Introductions

Instructor introductions

Participant introductions

- name
- organization
- position
- experience in project management
- What are your expectations for the course?



Course Materials

Participant notebook

- slides
- exercises



Solutions and other handouts will be distributed

Logistics

- Wifi connection
- Designated smoking area
- Lunch break
- Restroom location



Schedule

Day 1	Day 2	Day 3
1 – Introduction	6 – Progress and Change	9 – Diagnosing Problems
2 – Project Management Dashboard	7 – Quality and Defects	10 – Goal-Driven Indicators and Charting Data
3 – Schedule	8 – Basic Estimation and Planning	11 – Course Summary
4 – Resources and Cost		
5 – Size		

Approach and Rules of Engagement

Course Approach

- Lecture
- Exercises
- Questions and answers
- Discussions
- Participation

Rules of Engagement

- Participate.
- One person speaks at any given time.
- Ask relevant questions.
- Turn off your cell phones and other communication devices.
- Be prompt returning from breaks.

Completion Criteria

Successful completion of this course requires that participants

- actively participate in classroom discussions and exercises all three days
- do not miss any classroom time

Carnegie Mellon grants 2.5 CEUs for this course.

- We do not automatically grant PDUs, but you can request it directly from PMI. The course is 22 hours.



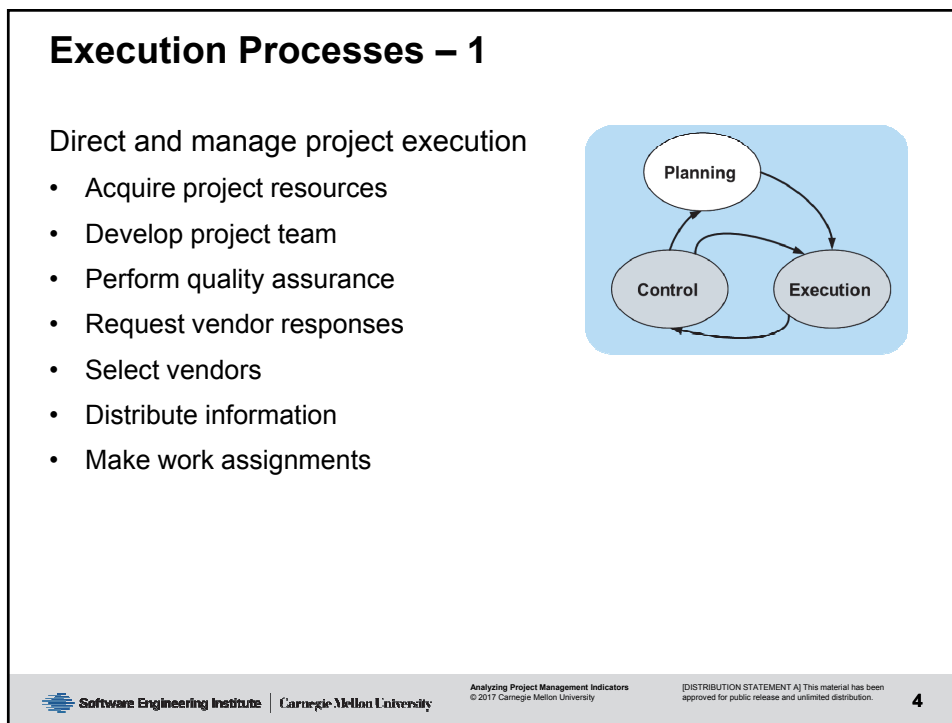
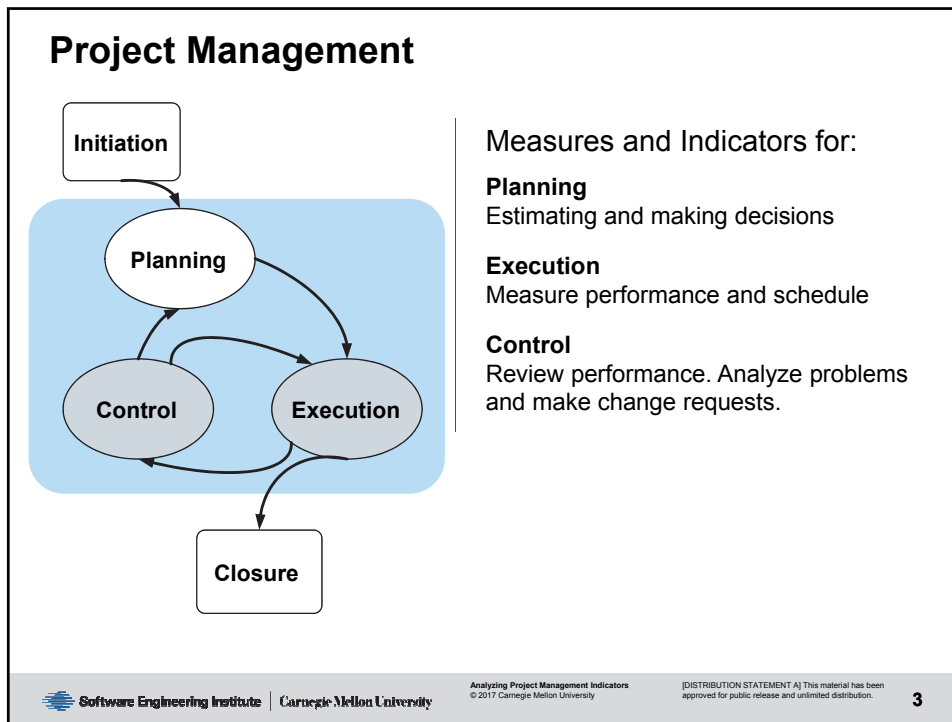
Module Outline



Overview of project management (using PMBOK)

Dashboard structure for project application

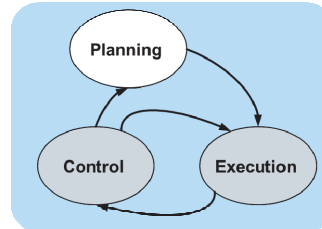
- Relate the dashboard data to normal project activity
- Describe the relationship to decision making



Execution Processes – 2

Inputs

- Plans
- Corrective and preventative actions
- Approved change requests and defect repair
- Validated defect repairs
- Administrative closure procedure



Outputs

- Deliverables
- Requested changes
- Implemented change requests (defects, risk actions)
- Work performance information

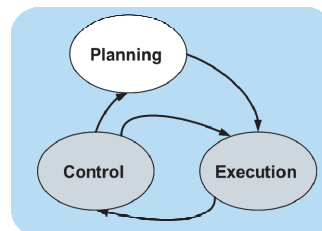
Monitoring and Controlling – 1

Monitor and control

- Consolidate measurement data

Integrated change control

- Scope control
- Scope verification
- Schedule and cost control
- Quality control
- Manage project team
- Performance reporting
- Manage stakeholders
- Risk monitoring and control
- Contract administration

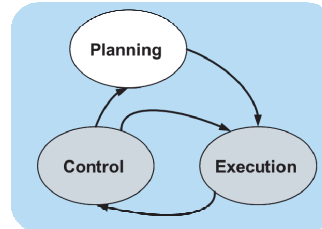


Monitoring and Controlling – 2

Monitor and Control Project Work

Outputs

- Recommended corrective and preventative actions
- Forecasts
- Recommended defect repair
- Change requests



Integrated Change Control

Outputs

- Approved (rejected) change requests
- Plan updates
- Scope updates
- Corrective and preventative actions
- Approved defect repair
- Validated defect repair

Defining a Dashboard

Data from project planning and execution

Supports information for multiple perspectives

Each quadrant suggests specific opportunities for decisions

Influences behavior to achieve project goals

The Project Dashboard Layout

A useful dashboard will contain data in each of these categories:

Schedule, Cost, & Resources	Effective Process
Scope, Progress, & Change	Quality

Content of Dashboard Quadrants

The first three quadrants represent opportunities to control and direct the work.

- The project manager decides on schedule and resource allocation.
- The project manager must involve both customer and development organizations to make changes in scope.
- Requests to change a process may be suggested by the project manager, but only developers and subcontractors can make the changes.

The fourth quadrant, product quality, represents the target objectives.

- Frequent quality checks during the development work help to avoid rework at the end of the project caused by testing failures.
- The dashboard records the frequency and outcome of all product quality checks, including the results of testing.

Schedule, Cost, & Resources	Effective Process
Scope, Progress, & Change	Quality

How to Read the Following Slides

Goal explains the connection of the information to the project goals.

Tracking Measures describe some typical measures related to the goal.

Questions represent ways to probe for potential problems and other sources of information for decision making.

Decision Potential describes several possibilities for action, change management, and change requests.

Following each dashboard element description are 1-2 slides about concerns specific to project managers and development teams.

Project Level describes measures that are needed to address project management decisions.

Team Level describes measures that teams use to monitor and adjust their own performance or to send a message to management.

Predictable Schedule, Cost and Resource Availability

Goal

- Project delivers on its forecast date at the forecast cost.
- Forecasts based on available data are recalculated on a regular basis.

Tracking Measures

- Percentage of predicted tasks completed within the week (or month)
- Differences between planned resources and resources used
- Earned Value Management (EVM) data used to estimate completion

Questions

- Has the project been predictable in recent milestone performance?
- Are forecasts regularly updated (every program review)? Does the current forecast predict a problem in project execution?
- Have resources been available and used as planned?

Decision Potential

- Project manager can change schedule and task sequence. Changes to scope or schedule and resources more than contingency require a change request.

Schedule, Cost, & Resources	Effective Process
Scope, Progress, & Change	Quality

Charts and Measures for Schedule and Resources

Project Level

- Milestone hit-rate or EVM measures Cost Performance Index (CPI) and Schedule Performance Index (SPI)
- Estimate at Completion (EAC) for date and cost (typically uses EVM data)
- Resource utilization and variance (used to forecast efforts)
- Critical resource availability (chief safety engineer, test lab)

Team Level

- Milestone slip chart (shows whether planning adjustments have been made)
- Milestone hit rate (shows performance to plan)
- Resource utilization and variance

Scope, Progress, and Change Management

Goal

- Communicate scope and progress, and prepare change estimates.
- These decisions require more stakeholders than “predictability” decisions.

Schedule, Cost, & Resources	Effective Process
Scope, Progress, & Change	Quality

Tracking Measures

- Actual progress measured in size compared to estimate of size at completion
- Changes in requirements and changes in other size estimates

Questions

- Is there a change to estimated size (e.g., more requirements)?
- Does adjusted size suggest an estimation error (e.g., planning problem)?
- Is the change request too large or too late to accomplish within budget?
- Will teams complete their separate efforts in time for integration?

Decision Potential

- Defer portions of scope to a later release or eliminate a deliverable.
- Create a change request (for cost or schedule); re-allocate scope to other team.
- Move scope or people between teams.

Measures of Scope, Progress, and Change

Team Level

- Count requirements change and estimated effects on size.
- List of all intermediate and final deliverables {object, estimated-size, completed-size}.
- Track (by group is OK) estimated-total and actual-completion. Use key milestone dates to anchor the discussion to project dates.
- Show updates for estimates as changes are received.

Project Level

- Consolidate separate deliverables by % of estimated cost or other normalizing factor. Track completion compared to key milestones.
- Track changes by same percentage of baseline.
- Use spotlight chart to show status of multiple deliverables.

Effective and Efficient Process

Schedule, Cost, & Resources	Effective Process
Scope, Progress, & Change	Quality

Goal

- Process produces few defects and removes most.
- Process uses resources efficiently avoiding rework.

Tracking Measures

- Quality assurance audits (processes performed as expected)
- Performance measures are within expected ranges
- Behavioral observations (team, technology)

Questions

- If processes are not followed, why (e.g., behavior, training, problem)?
- Is performance data available and reviewed when the data suggests?
- Are technology changes reflected in process changes (e.g., training)?

Decision Potential

- Demonstrates the need for process change or behavior intervention.

Measures of Effective and Efficient Process

Project Level

- IPT members actively participate on action items, interfaces, etc.
- Monthly project QA report
 - # of quality checks by type and completed as scheduled
 - # of audit findings and # of defects found
 - risk associated with findings and any mandatory corrections
- Defect processing/rework analysis

Team Level

- Selected process performance measures (e.g., inspections)
- # of quality audit findings (e.g., process and product errors)
- Quality events skipped or delayed and defect escapes
- Defect processing/rework analysis

Product Quality

Goal

- Helps users deliver improved value to their customers.
- Shows product validation results

Tracking Measures

- Measures of product performance
- KPP and TPM test results, including simulations
- Results of successful/unsuccessful product validation tests and reviews

Questions

- Will the product achieve desired quality and performance?
- Do validation checks prove the value-in-use potential?

Decision Potential

- Risk reduction activities.
- Update test cases and validation cases.
- Renegotiate performance target

Schedule, Cost, & Resources	Effective Process
Scope, Progress, & Change	Quality

Measures of Product Quality

Project Level

- Scenarios for product validation are defined and accepted.
- Technology readiness
- Architectural review (e.g. *ATAM and *AADL)
- Summary of product verification and validation results (e.g., tests passed)

Team Level

- Defects removal target achieved
- Current test results of key performance criteria
- Quality thresholds evaluated (e.g. churn)

**ATAM: Architectural Tradeoff Analysis Method*

**AADL: Architecture Analysis & Design Language*

Exercise 1

Divide into groups.

Each group take a current chart that they use.

Address the following:

- What is the goal?
- What questions does it answer?
- What is the decision it informs?

Discussion.

Outbrief

Our dashboard measure is ...

Applies to goals ...

- Predictable Project
- Scope
- Quality of Work Product
- Effective Use of Process

Satisfactorily useful for which of these purposes

- Decision
- Look ahead
- Lead
- Telling the tale of the project

Summary

Project management indicators are most successful when they

- have clear objectives
- are interpretable and actionable
- involve the technical staff in deciding what to collect and how
- identify the criteria for evaluating their effectiveness
- support decision making and action based on results



Module Outline



- Questions about schedules
- Selecting charts representing schedule
- The project network diagram
- Events for measuring schedule

Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

2

Schedule Goal: Predictability

The schedule designates assignments and synchronizes work.

- Resource availability is critical to schedule performance.

Schedule changes are frequent and frequently needed.

- Errors in scheduling cause inefficiency in allocating resources and thereby increase in project costs.

Project Manager must communicate trust in the schedule

- Convey that resources and materials are ready.
- (What are resources and materials?)

Start change analysis when these cannot be ready in time.

- Provide evidence that current schedule cannot work.

Measures related to schedule/availability

Task completion dates and count

- How many tasks are late? How many were completed on time?

Schedule slip – several late dates from a team.

- Is there a general schedule slip? Is critical path late or moving?

Look ahead and Preparation

- Are any tasks waiting for resources?
- Are resources waiting for input?
- Has schedule forecast changed? Have resources been notified?

Monitor resource pool as well as schedule completion.

- Under utilization may reduce budget but work may not get finished.

Sample Charts and Indicators

Gantt Chart: List of project tasks represented by time bars

Slip Chart: Changes to planned completion dates

Network (PERT) Diagram: Task dependency on other tasks

Critical Path Tasks and Dates: Overall project slip analysis

Milestone Hit Rate: Team level measure of task completion

Gantt Chart: Telling your schedule story

What's the story here?

Lifecycle Activities

Req'ts Analysis

Architecture

Build 1

Detailed Design

Code & Unit Test

Integration & Test

Build 2

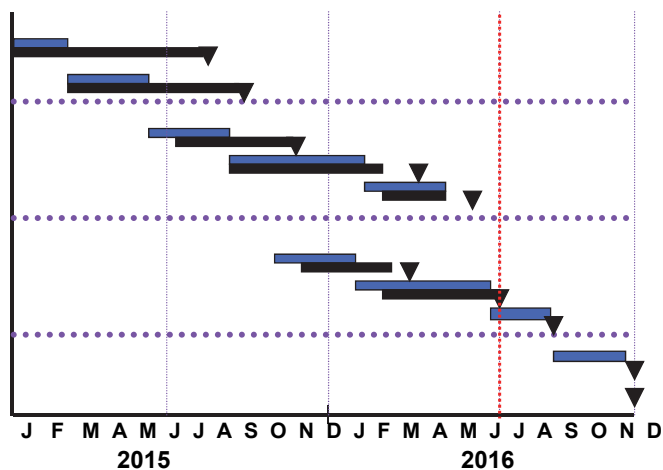
Detailed Design

Code & Unit Test

Integration & Test

Product Integration

Formal Qual. Test



Gantt Chart Questions

History:

- Does the chart indicate the date of record?
- Is there evidence of task slippage?

Forecasting:

- Which immediate tasks are likely to be affected?
- Will we need to notify teams?
- Is some re-planning needed?

General:

- Gantt chart does not have a lot of detail.
- It may suggest which tasks/teams we need to probe.

When to Use a Gantt Chart

Pros

- Provides overview and stages.
- Status reporting to show how much of the plan has been completed by displaying the progress of an activity.
- Easy to see major milestone slips.

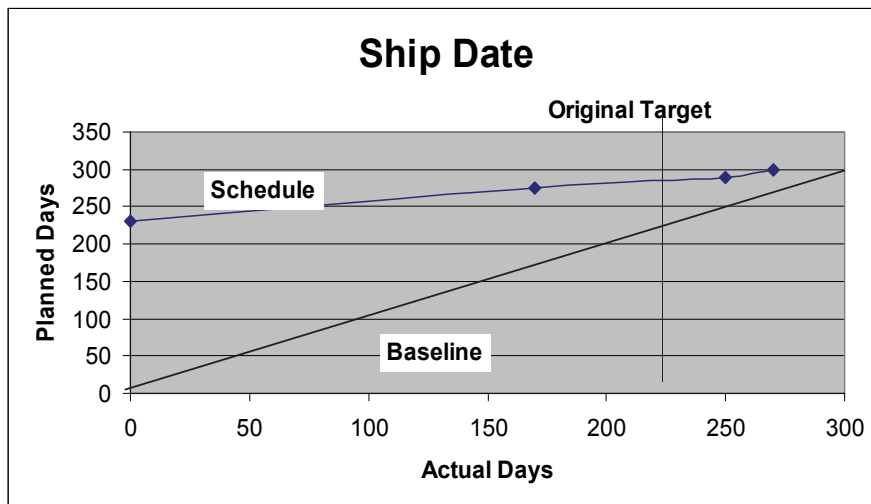
Cons

- Makes no sense until you have estimates.
- The Gantt chart doesn't really allow for the correlation and/or dependency between tasks.
- Very difficult to see the effects of task overrun.
- Not good for forecasting or analyzing changes

Tools

- Every project management tool produces a Gantt chart.

Slip Chart

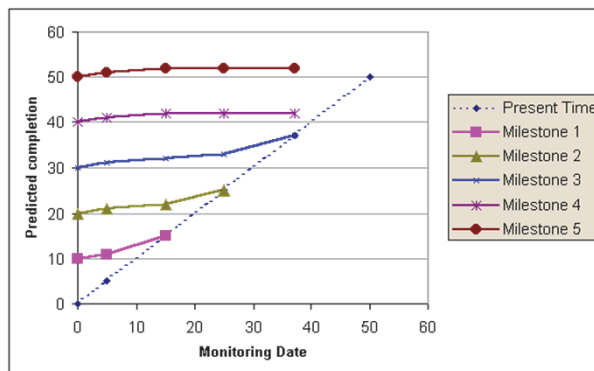


Each marker represents a schedule slip.

Milestone Slip Chart

Each milestone date is indicated on both axes.

- Current date is horizontal axis
- Dotted line is $x=y$
- Completion and revision are plotted at current vs projected.



Slip Chart Questions

History:

- Slip chart shows past patterns.
- Is a team or deliverable consistently falling behind?
- Has forecast been adjusted?

Forecast

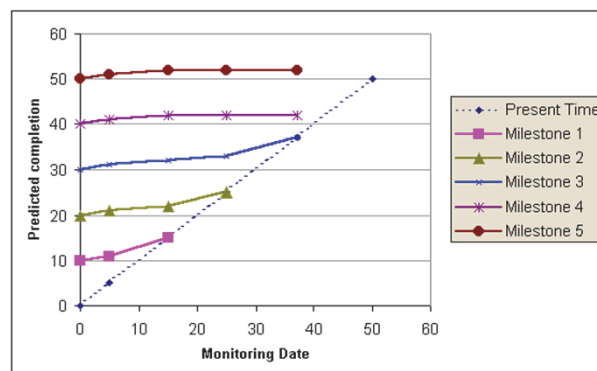
- Can we identify a task that may slip because of a late completion?
- Who owns the affected tasks?

Milestone Slip Chart

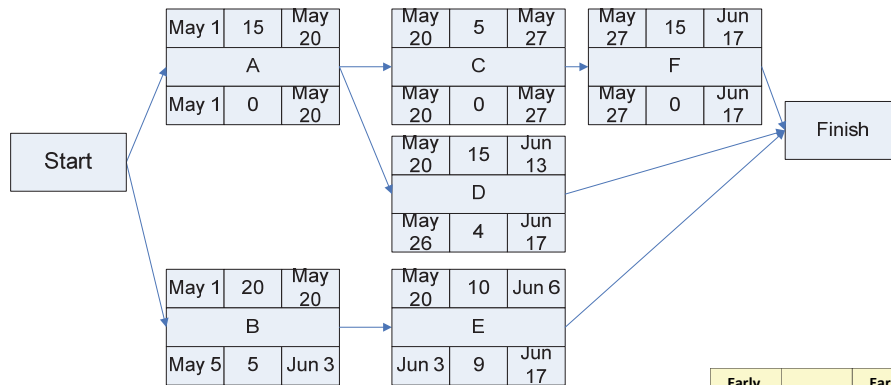
The last completed milestone is the blue line.

Ask “Do you believe the current projection for top two lines?”

Useful for teams to track their own planning.



Project Network



Project network diagrams are used to identify which tasks directly affect other tasks.

Early Start	Duration	Early Finish
Task Name		
Late Start	Slack	Late Finish

Schedule Building Block (PERT)

Use network diagrams rather than Gantt charts for planning and looking ahead. This will take two passes through the network.

Calculate Early Start and Early Finish by working forward through the network.

Calculate LS, LF, working backward. Slack = LS-ES

The Critical Path always has Early Finish=Late Finish (i.e. zero "Float" or "Slack").

Duration is estimated.

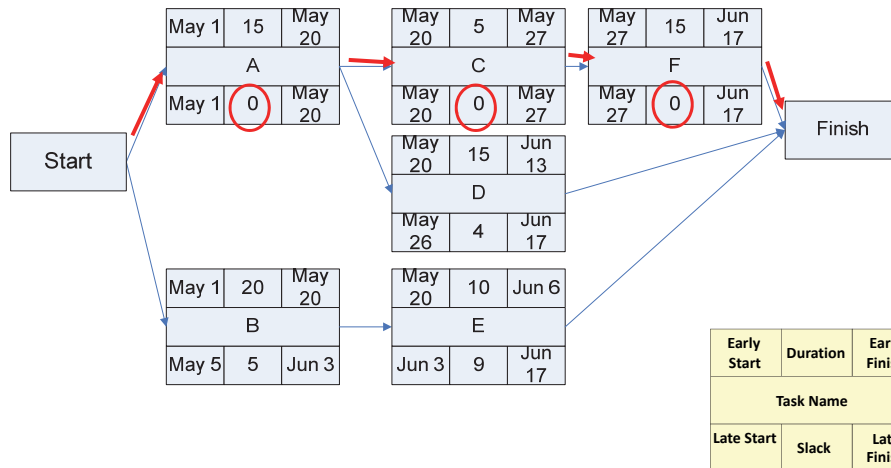
Early Start	Duration	Early Finish
Task Name		
Late Start	Float	Late Finish

Analyzing Project Management Indicators

Project Network

Tasks on the “critical path” have zero float.

Length of the critical path is sum of durations of connected blocks.



Using a Network Diagram

Makes a large wall chart.

Color code completed tasks and active tasks.

Late tasks and synchronization problems should be apparent.

Why use the critical path?

Critical path analysis is a forward look at the work.

- What tasks are on the immediate horizon?
- Which tasks have the potential to delay the project?
- Who do I need to influence to get things done on time?
- What activities can/should be crashed?
- Is there an opportunity to reallocate people or scope?

An activity that feeds into the critical path should not be late.

Use critical path analysis to plan and prioritize work.

If critical path changes frequently, it is time for re-planning.

Milestone Hit Rate

A single team can measure its schedule on regular intervals by
Milestone Hit Rate

- $\frac{\text{\#milestones (or inchstones) completed on time}}{\text{\#planned}}$

Watch for teams that consistently over-promise.

The “Lean Construction Institute” calls Hit-Rate
 $\frac{\text{\%promises_completed}}{\text{\#planned}}$.

Milestone Hit Rate

A Team-Level “Binary-Inchstone” Measure
Also called “% promises complete”

Team	1-Mar	8-Mar	15-Mar	22-Mar	29-Mar	Trend
Component A	98%	85%	100%	98%	98%	→
Component B	100%	98%	95%	92%	88%	↘
Systems Engineering	97%	87%	100%	94%	100%	↗
Board 1	92%	98%	95%	100%	100%	↗
Test Team	100%	80%	97%	100%	100%	→

Green > 95%
95% ≥ Yellow > 85%
85% ≥ Red



Events for Measuring Schedule

Milestone completion

- “binary inch-stones” (weekly or? task duration)
- Milestones that are coordinating points of project.

Deliverable completion and handoff

- Whenever responsibility for deliverable changes hands.

Lifecycle events

- Program reviews
- External demonstrations
- Delivery to customer or non-project group
- *Typically required for all similar projects within the organization.*

Remember: how we collect the data is an important aspect!



“Agile” and Schedule

Agile teams use a fixed “cadence” rather than a schedule.

- Project sprints have a defined duration (e.g. 2 weeks).
- Only handoffs between teams represent visible milestones.

Teams that receive deliverables from agile teams:

- Mark the date received compared to plan
- Verify the deliverable “readiness for use.”

Your Turn: Discussion

Consider schedule representation in a program review deck.

Does it answer the questions?

- How are we currently performing against the plan?
- What tasks are ahead? What tasks are behind?
- What adjustments might be made?
- What are the upcoming tasks?
- Does schedule suggest a major handoff or synchronization occurs soon?

What questions are suggested by the current data as represented?

Thoughts on Compressing the Schedule

Add people and overtime to the critical path activities.

- It just doesn't help much if activities off the critical path finish earlier than planned.
- Make sure tasks that feed the critical path have some buffer time for critical path tasks.
- Remember: adding many people will create additional delays.

Change the scope of a task by:

- Reducing the scope (increase risk?)
- Split the task into parallel tasks.
- Combine tasks to increase efficiency (decrease task communications and switching)

Looking Forward on Your Schedule

What do teams need within the near term?

- Their planned deliverables and assignments for work
- Information and products that others supply
- Issues that need to be decided or assumptions will they make instead

What do you need from other stakeholders?

When can you tell the team to start their work?

What do you need to tell your stakeholders?

How much schedule slip requires a change to the plan?

We measure for communication, establishing trust, and supporting decisions.



Module Outline



- Goals for Managing Resources
- Resource Problems
- Resource Charts
- Earned Value Management
- EVM Charts

Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

2

Resources & Cost

Goal – Acquire and assign resources at the right time.

Questions

- Are we still within planned cost?
- What is the current estimated cost and schedule?
- Does the difference suggest a need for a change order?
- Have we utilized resources appropriately?
- Are resources available for the next set of tasks?

Then following this slide we have

- Question(s)
- Indicators for the Question
- Decisions

Issues and tips

List of indicators

Resources (staff, hours, facilities, and tools)

- Utilization
- Resource planning
- Resource turnover
- Example

Earned Value charts

- Cost account structure
- EVM sample charts

Resource Chart Goals

Goal

- Avoid over-allocating resources.
- Find resources who have available time.
- Identify mis-allocated resources

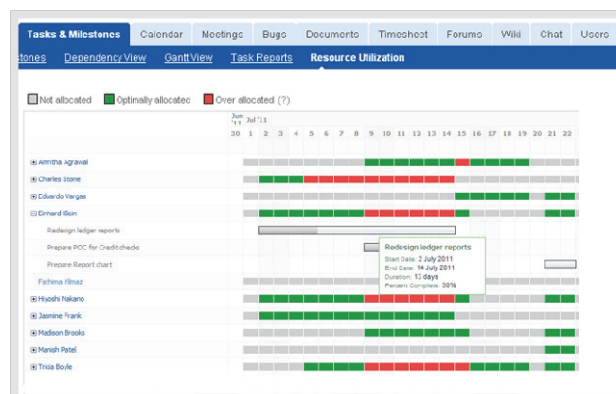
Questions

- Are we expending resources at the expected rates?
- Which resources are over-allocated?
- Are there critical availabilities? (test labs, special skills)
- Are there tasks being executed that are not scheduled yet?

Over-Allocation / Available

Chart shows resource over-allocation and availability.

- Over-allocation suggests that tasks may not always be completed on time.



Resource Misallocation

What does this mean:

- Team may begin charging to cost account even though work is not ready for the team.
- Sometimes an individual is assigned in order to keep the individual available to the team when work becomes ready.

Strategy:

- Look at open cost accounts and identify on WBS.
- Check that corresponding portion of WBS is on current schedule.

Chart

- Team hours by month
- Chart shows 2 suspect cells that could warrant examination during early development.

Role	Jan	Feb	Mar	Apr
Architect	150	306	290	60
Developer	0	450	780	1240
Tester	94	108	180	611

Staff Allocation

Can you see possible misallocation?

Reporting Period		1	2	3	4	5	6	7	8	9	10	11	12	13	14
Mgt	Planned	1	2	2	2	2	3	4	6	6	6	5	5	2	2
	Actual	2	2	2	2	2	4	4							
Analyst	Planned	1	3	4	4	4	4	4	4	4	5	4	3	1	1
	Actual	2	2	2	4	4	4	4							
Designer	Planned		3	10	11	12	11	13	12	8	3	1	1	0	0
	Actual	4	4	4	7	12	12	12							
Programmer	Planned		1	1	1	2	6	18	33	35	35	19	14	3	3
	Actual	7	7	7	11	20	20	20							
Tester	Planned			1	1	2	2	4	8	14	16	14	10	6	2
	Actual														
Other (QA, CM, etc.)	Planned		1	1	1	2	2	2	2	2	2	2	2	2	2
	Actual			0	1	2									
Total	Planned	2	10	19	20	24	28	45	65	69	67	45	35	14	10
	Actual	15	15	15	25	40	40	40	0	0	0	0	0	0	0

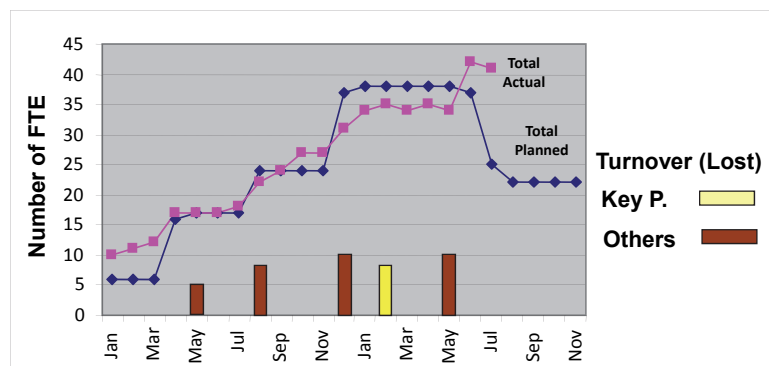
Resource Availability Indicator

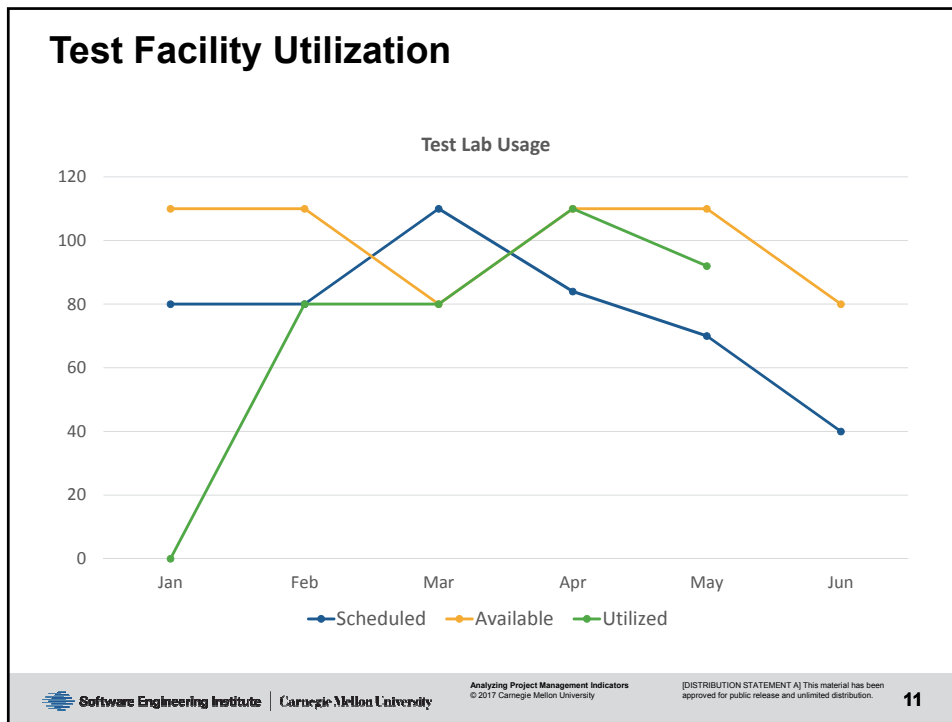
(Example)		Jan		Feb		Mar		Apr	
		Actual	Auth.	Actual	Auth.	Actual	Auth.	Actual	Auth.
Project Mgt									
Engineer	Entry Level								
	Journeyman								
	High Grade								
Architect	Entry Level								
	Journeyman								
	High Grade								
Tester	Entry Level								
	Journeyman								
	High Grade								
Specialty	Entry Level								
	Journeyman								
	High Grade								

Significant differences between Actual and Authorized levels could be a warning indicator.

Staffing and Turnover

Staffing Levels





Earned Value Charts

Goal

- Identify and communicate current and future costs.
- Predict completion dates and costs

Questions

- Are we expending resources at the expected rates?
- Are we completing work at the expected rates?
- What are cost and schedule estimates for completion?
- Do differences between plan and current status suggest a need for a change order?

Introduction to Earned Value

Earned Value Management

- Uses accounting information to track cost against schedule.
- Provides insight into utilization of staff.
- Provides insight into project progress.
- Can be used to forecast cost and schedule completion.
- Required by DCMA and federal law.

The basics are easy to learn.

- Understand the structure for EVMS
- Learn the formulas
- Practice

EVMS Accounting Structure

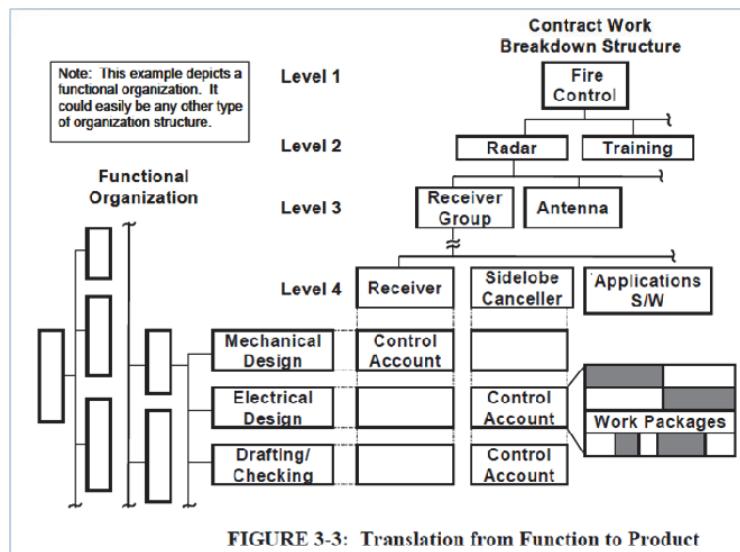
A project always has two intersecting structures:

- A work breakdown structure (WBS) associated to various deliverables and tasks that occur during the project.
- An organizational breakdown structure (OBS) associated the accounting hierarchy of the corporation(s)
- In other words, a development project always has a matrixed structure – one aligned to the hierarchy of the project, the other aligned to the organizational hierarchy.

The WBS View is the one usually shown on reports.

The OBS View can show resource problems.

Control Accounts: WBS and OBS



Importance of Control Accounts

WBS: Analyze for progress

- Sum up control accounts associated to deliverables.

OBS: Provides insight to staffing problems

- Under-running the plan suggests that deliverable will make slow progress.
 - Perhaps resources are not available.
 - Perhaps there is a technical problem that must be solved before people can be added.
- Over-running the plan
 - A deliverable was underestimated
 - Staff is loaded to an open control account but not necessarily restricted to the identified deliverable.

Be prepared to dig deeper.

Plan and Actual

Nomenclature (modern and traditional)

- **PV=Planned Value or**
BWCS = Budgeted Cost of Work Scheduled
 - Created by allocating estimated resource requirements.
- **AC=Actual Cost or**
ACWP=Actual Cost of Work Performed
 - Recorded as resources are utilized per control account
- **EV=Earned Value or**
BWCP=Budgeted Cost of Work Performed
 - Recorded as work is completed and sometimes partially completed.

EVMS Measurement Activities

PV

- Contractor estimates task matrix for schedule and resources.
 - Early information appears in basis of estimate.
 - Later detail is in control accounts.

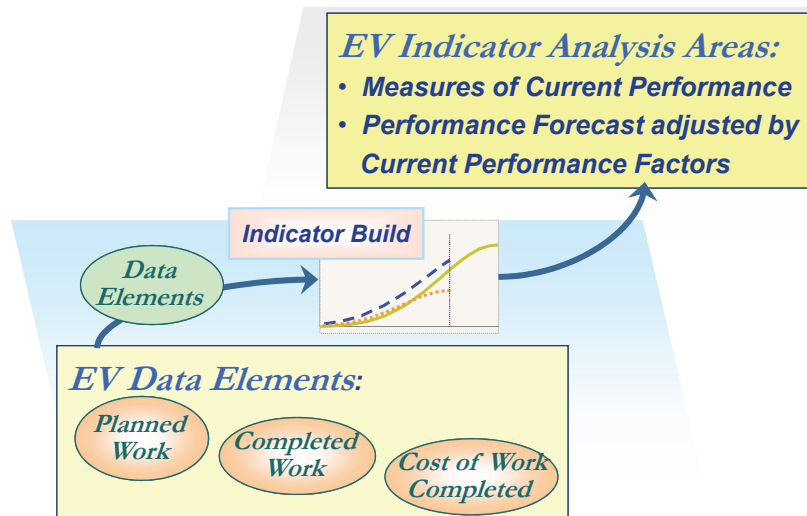
AC

- Accounting data is provided at some *standard level* of WBS.
Further detail may be requested for lower levels of WBS.

EV is calculated: a typical example for a single control account

- EV=0% of PV until task has started
- EV=50% of PV if task has started but has not completed*
- EV=100% of PV **only** if task has completed
- *Caution: Check to see that work has really started.*
- *❖ Some companies only use 0% and 100%, eliminating 50%.*

EVM Data Elements



Example: Planned Value (PV) -1

Planned Value is a time-phased budget of: defined, estimated, and scheduled work.

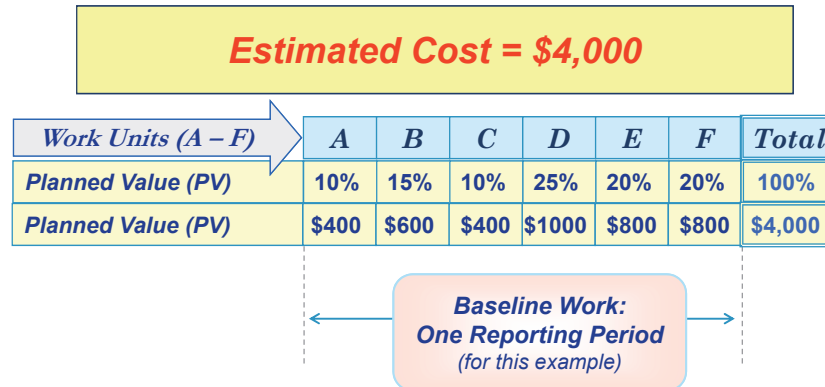
Step1: Allocate estimate across project milestones or deliverables.

Step 2: Calculate projected cost of each unit.

Work Units (A – F)	A	B	C	D	E	F	Total
Planned Value (PV)	10%	15%	10%	25%	20%	20%	100%

Ref: DoD EV

Example: Planned Value (PV) -2



Ref: DoD EV

Software Engineering Institute | Carnegie Mellon University

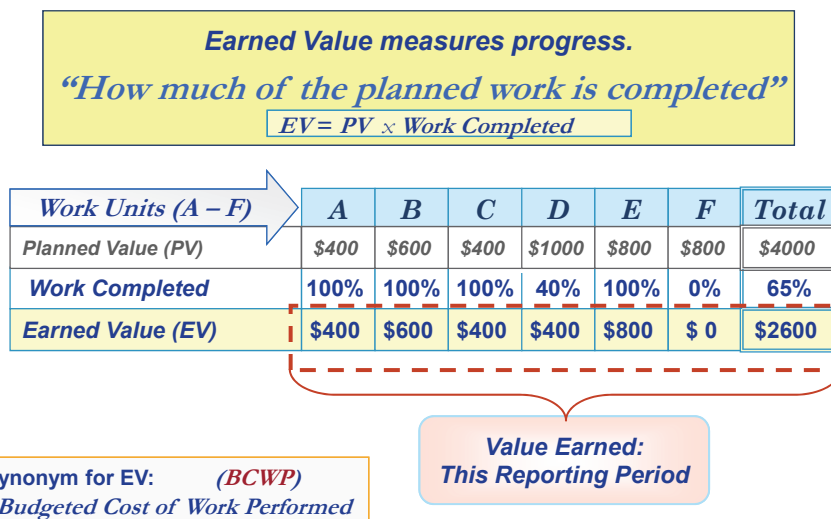
Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

21

Example: Earned Value

We planned that tasks A through F will be completed by today.



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

22

Example: Schedule Variance (SV) Indicator

Compare Planned Value to Earned Value to measure Schedule Variance

Work Units (A – F)	A	B	C	D	E	F	Total
Planned Value (PV)	\$400	\$600	\$400	\$1000	\$800	\$800	\$4000
Work Completed	100%	100%	100%	40%	100%	0%	65%
Earned Value (EV)	\$400	\$600	\$400	\$400	\$800	\$0	\$2600
Schedule Variance (SV) as \$	\$0	\$0	\$0	-\$600	\$0	-\$800	-\$1400

Schedule Variance

Schedule Performance Index measure of schedule performance.

$$SPI \text{ (represented as a \%)} = EV / PV = 65\%$$

Example: Cost Variance (CV) Indicator

Compare Actual Cost to Earned Value to measure Cost Variance

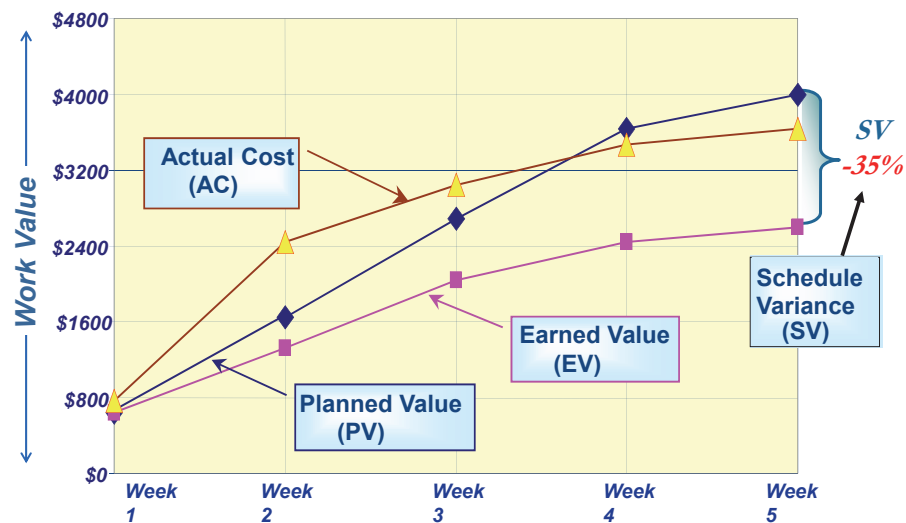
Work Units (A – F)	A	B	C	D	E	F	Total
Earned Value (EV)	\$400	\$600	\$400	\$400	\$800	\$0	\$2600
Actual Cost	\$360	\$880	\$320	\$1200	\$880	\$0	\$3640
Cost Variance (CV)	\$40	-\$280	\$80	-\$800	-\$80	\$0	-\$1040

Cost Variance

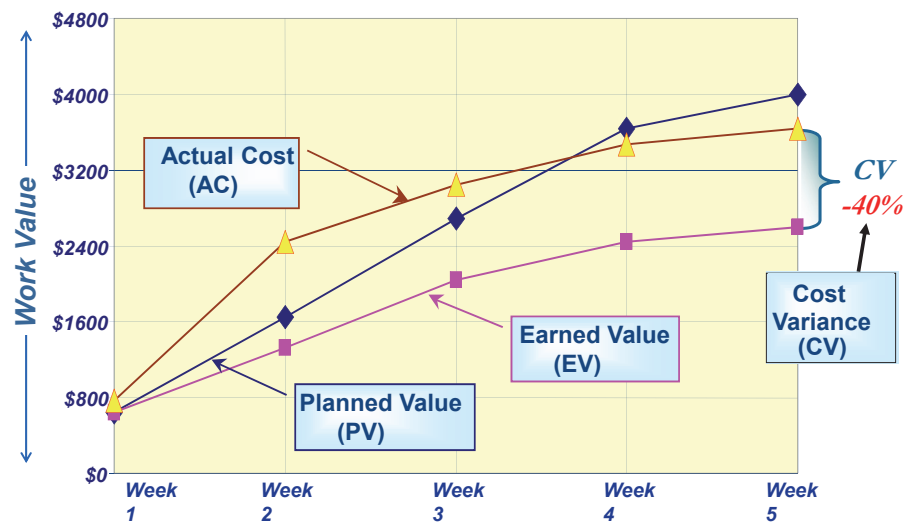
Cost Performance Index measure of Cost performance.

$$CPI \text{ (represented as a \%)} = EV / AC = 71\%$$

Sample Earned Value Chart-1
Schedule Variance – $SV = EV - PV$

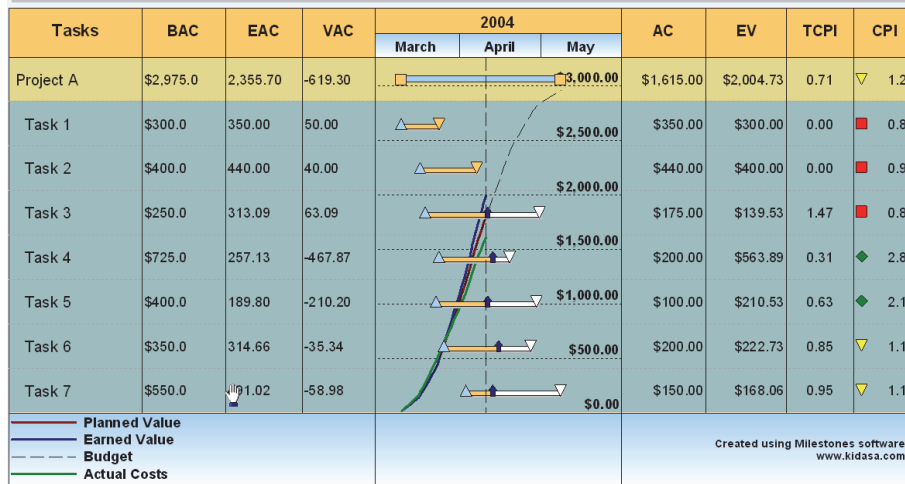


Sample Earned Value Chart-2
Cost Variance – $CV = EV - AC$



Earned Value Management – Forecasting

Performance Review

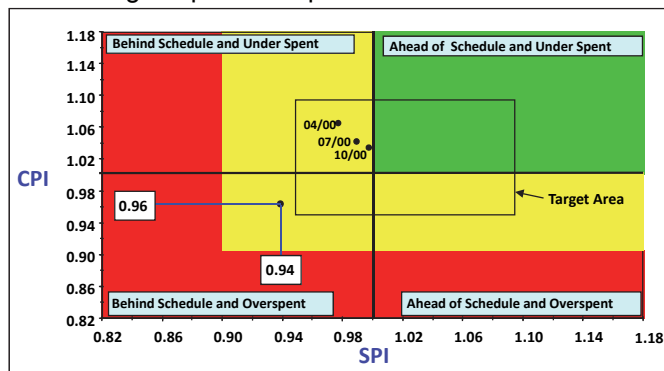


From www.earnedvaluemanagement.com

Efficiency Status Indicator

Change thresholds for CPI and SPI set at 10% below nominal.

- Green area is ok – within budget and schedule plan
- Yellow is a warning.
- Red signals and change in plan is required.



EVM Exercise

Calculate EAC-cost and EAC-schedule for the data provided on hand-out.

Comments about EVMS

EVMS aggregates a lot of information into 3 numbers.

- Problems that do not immediately contribute to those numbers may hide for a long time.
- Accurate estimation is critical.
- We need many accounts for milestones/deliverables.
- We must have clear completeness criteria.
 - What is meant by “80% complete?”
- Earned Value data is obtained from the accounting department.
 - It is often a month later than the date it was captured.
- EVMS can tell you there is a problem and how big it is.
It does not tell you where to find the cause or how to fix it.

Summary

You can manage resources.

- Costs are calculated by accounting. You cannot actually manage the cost except indirectly.

Think about how resource changes and resource utilization might affect your project. Create a picture that lets you see these things quickly.

Use EVMS to estimate completion and to help estimate change requests.



Module Outline



What to measure and why

- A simple size measure
- The importance of operational definitions

Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

2

What do we mean by size?

How big is it?

How many things can it do?

How many resources will it require?

How long will it take?

How many are there?

-
-
-

Reasons for Estimating or Measuring Size

Plan resource allocation and duration for various activities.

- We need to balance team sizes so that the work can be reasonably coordinated.

Plan quality activities using size ratios.

- We must decide how much time is used for reviews and inspections.

Track progress towards completion

Each separate activity and each team may have its own productivity or performance ratio.

- Coding
- Inspecting documents
- Various types of testing
- (name your own)...

Normalize by size for comparison and other uses.

What Measures for Size Are Needed?

Few projects deliver only software.

- All elements of project scope need to be sized and some internal deliverables should have size measures.
- If the task duration or effort is affected by the size of input or the size of a completed work product, then some (simple) measure of size is appropriate.

Identify work products (internal and external)

- Which work products should have a size measure?
- Why? Who needs to know?

Entities and Attributes

Entities are things with names

Attributes are characteristics that distinguish, measure or otherwise describe instances of the entity.

Interestingly enough, attributes may themselves be entities.

What can we measure the size of?

Software entities

products	artifacts	organizations
processes	activities	environments
resources	agents	tasks

Entities have attributes

products... size, language, defects ...
processes... start, delivery, resources, tools, skills...
resources... people, skills, facility, constraints...

Attribute measures

products... SLOC, # of modules, function points,
of bubbles in a dataflow diagram,
of open defects per KSLOC
processes... cycle time, staff-hours used, unit cost,
rework items, effort in rework
resources... due date, # of people assigned,
#workstations, capacity measures

Discussion

Describe at least two size measures for each of the following:

- Code
- Test cases
- Design documents

Module Outline



What to measure and why

A simple size measure

The importance of operational definitions

Super Simple Size Method

A development team (rather than a department) can often agree on size categories for their work.

You ask the team to think about each type of deliverable – (design document, use case, entity-relationship-diagram, test script, etc.).

For each deliverable ask them to decide on how to allocate the sizes into 3-5 size buckets.

- Very small is the smallest we ever do
- Small is larger
- Medium, and large are larger still
- Very large is the largest we have seen

Then decide how size is measured and create a table.

Super Simple Sizing Table

Unit Names	Small Task effort	Medium	Large
Design unit	(Count)		
Code unit			
Test unit			
Use case			
ERD			



SSS Example

Unit Names	Small	Medium	Large
Design unit	2 pages	6 pages	18 pages
Code unit	1 page	3 pages	9 pages
Test unit	Simple takes half day	Medium takes all day	Large takes 2+ days
Use case			
Operational instruction	1 page	2 pages	6+ pages



The Distribution of Size Data

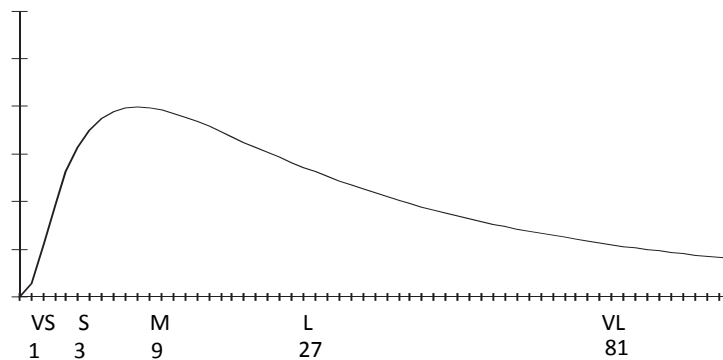
The probability distribution of this size data usually will not follow a bell-shaped curve (normal).

- many small and medium values
- a few large values
- **no negative values**

A normal distribution could cause us to make some estimates with a negative size, but we can avoid this impossible situation by using a log-normal distribution

A Log-Normal Distribution

The “geometric scale of the data appears on the x-axis.



If we were to graph $y=\log(x)$,
then we would see a normal bell-shaped curve.

Easier than Log-Normal

To create useful size data that mimics Log-Normal:

Size = (very small, small, medium, large, very large)

Very small = smallest useful size

Very large = largest you have ever done

Assume

Small = 2 x Very small (or 3 x)

Medium = 2 x Small (or 3 x)

...

Very large = 2 x Large (or 3 x)

This geometric scale will mimic the log normal distribution

Why use 2x or 3x?

2x is usually the smallest difference between objects that people can detect with their normal senses (hearing, etc.)

3x is usually the smallest difference between objects that most everyone will agree there is a difference.

The implied ranges are:

2x: $0.67 < x < 1.33$ (high = 2 x low)

3x: $0.5 < x < 1.5$ (high = 3 x low)

SSS Recap

- Identify each deliverable (hand off or required storage) in your development process.
- Identify the smallest and largest examples you can.
- Determine how many sizes you can distinguish.
(small, medium, large) or (VS, S, M, L, VL)
- Identify a “standard effort” for these sizes.
- See if standard effort for these sizes differs by 2 or 3 if possible.
- Test your hypothesis by thinking about both the creation process and the peer review process.
 - “Analysis of Variance” can be used to confirm these sizes.

Super Simple Sizing Summary

The Super Simple Size is a proxy-based measure.

- The concept requires a team to think about its internal work products
- To develop a sizing standard for each of these work products (and also a procedure for recognizing completion)
- The team can then measure and report its own performance data.

Using SSS requires consensus and usually historical experience to develop the sizing standard.

The size must eventually be translated into effort.

Story points in the Agile method are an example.

Module Outline



What to measure and why

A simple size measure

The importance of operational definitions

How Many Physical Source Lines in This C Language Program?

```
#define LOWER 0    /* lower limit of table */
#define UPPER 300  /* upper limit */
#define STEP 20    /* step size */

main() /* print a Fahrenheit-Celsius conversion table */
{
    int fahr;
    for(fahr=LOWER; fahr<=UPPER; fahr=fahr+STEP)
        printf("%4d %6.1f\n", fahr, (5.0/9.0)*(fahr-32));
}
```

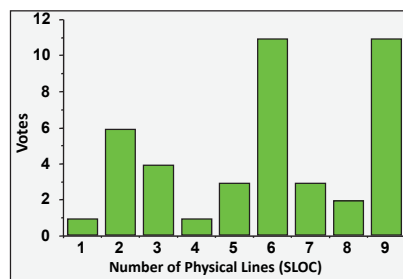
REF: Kernighan & Ritchie 78

Poll – Physical Source Lines

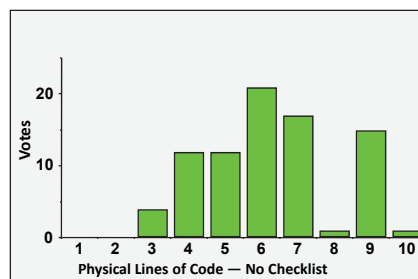
		Count										
		1	2	3	4	5	6	7	8	9	10	11
Votes	1											
	2											
3	3											
	4											
5	5											
	6											
7	7											
	8											
9	9											
	10											
11	11											
	12											

Poll Results: Fahrenheit to Celsius

COCOMO Users' Group



SEPG Conference

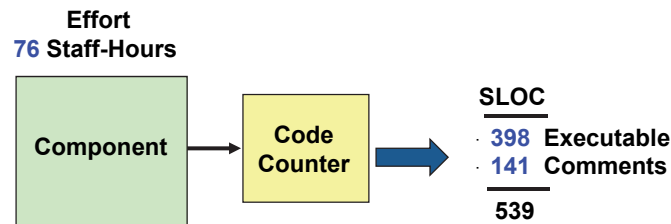


$$\text{Development Effort} = A (\text{KSLOC})^B C$$

A = Productivity Constant
B = Scale Factor
C = Proportionality Constant

[DSMC 96]

Size and Computing Productivity



Which productivity number do you use?

- $539 \text{ LOC} / 76 \text{ Staff-Hours} = 7.1 \text{ LOC/Hr}$
- $398 \text{ LOC} / 76 \text{ Staff-Hours} = 5.2 \text{ LOC/Hr}$

Source Statement Counting

This section will discuss the importance of rigorous measurement specification.

It will also provide some guidance for how to create a rigorous specification for your internal use.

The Checklist: Rules for Counting

Operational support for:

- data specification
- data collection
- data retrieval & reporting

The forms shown are:

- Definition checklist**: A checklist with five items, each with a checkmark.
- supplemental rules form**: A form with three rows, each starting with 'xx' followed by a line.
- data spec A:**: A checklist with five items, each with a checkmark.
- recording form A**: A form with three rows, each starting with '#' followed by two checkboxes.
- data request**: A checklist with five items, each with a checkmark.
- reporting form**: A form with two rows, each starting with '#' followed by two checkboxes.

General Format of Checklist

Identification Section			
	Includes	Excludes	Optional
Attribute #1			
Value 1			
Value 2			
⋮			
Value N			
Attribute #2			
Attribute #M			

Attribute: Statement Type

Definition Checklist for Source Statement Counts

Definition name: <u>Physical Source Lines of Code</u> (basic definition)		Date: <u>7/9/96</u>
		Originator: <u>SEI</u>
Measurement unit:		
Physical source lines ☒		
Logical source statements ☐		
Statement type	Definition ☒ Data array ☐	
When a line or statement contains more than one type, classify it as the type with the highest precedence.		
Order of precedence ->		
1 Executable		Includes ☒ Excludes ☐
2 Nonexecutable		Includes ☒ Excludes ☐
3 Declarations		Includes ☒ Excludes ☐
4 Compiler directives		Includes ☒ Excludes ☐
5 Comments		Includes ☐ Excludes ☐
6 On their own lines		Includes ☐ Excludes ☒
7 On lines with source code		Includes ☐ Excludes ☒
8 Banners and nonblank spacers		Includes ☐ Excludes ☒
9 Blank (empty) comments		Includes ☐ Excludes ☒
10 Blank lines		Includes ☐ Excludes ☒
11		Includes ☐ Excludes ☐
12		Includes ☐ Excludes ☐

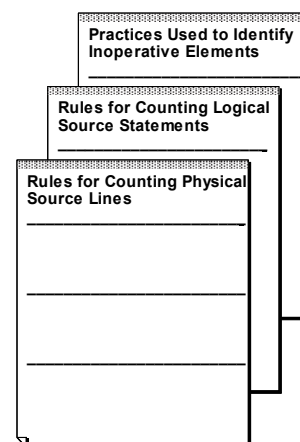
Supplemental Rule Forms-Size

Purpose

- to explain the rules that cannot be addressed with the definition checklist

Examples

- rules used to distinguish among statement types
- rules used to identify beginnings and endings of executable statements, declarations, compiler directives, and comments
- practices used to identify dead code



LOC Checklist: Principal Attributes

Definition Name _____		Date _____
		Originator _____
Measurement Unit	Physical source lines ☐	Logical source lines ☐
Statement Type	Includes	Excludes
How Produced		
Origin		
Usage		
Delivery		
Functionality		
Replications		
Development Status		
Language		

Software Engineering Institute | Carnegie Mellon University

© 2017 Carnegie Mellon University
 approved for public release and unlimited distribution.

29

Attributes Description – 1

Statement type: classifies source statements and lines according to the principal functions that they perform

How produced: identifies the process used to produce individual statements and lines e.g. **auto generated**

Origin: identifies the prior form upon which the product software is based, e.g. **modified, reuse**

Usage: distinguishes between software that is developed or used as an integral part of the primary product and software that is not e.g. **test scripts**

Attributes Description – 2

Delivery: identifies the form and destination of the source statements that are produced

Functionality: identifies whether or not source statements are functional parts of the product

Replication: distinguish a product's master source statements from various kinds of copies (replicates) of those statements

Attributes Description – 3

Development Status: describes the degree of which statements under development have progressed toward becoming integral parts of operational products.

Language: identifies the source languages or languages that are included in a statement count

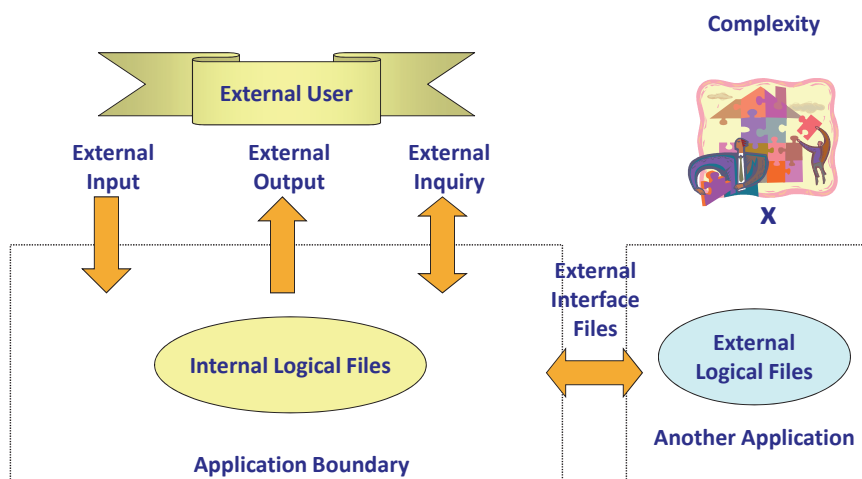
Clarification: clarify the rules we use to assign statements and lines to the “statement type”

Function Point (FP) Counting

The function point metric is composed of the weighted totals of five external aspects of software applications:

- **Inputs:** user inputs such as forms, screens, etc.
- **Outputs:** reports, screens, electronic signals, etc., leaving the application
- **Inquiries:** user queries requiring response
- **Logical files:** record sets maintained within or by the application
- **Interfaces:** record sets passed to or received by external applications
- **Weights:** are based on a complexity calculation

Function Point Definition



Pros and Cons of FPs

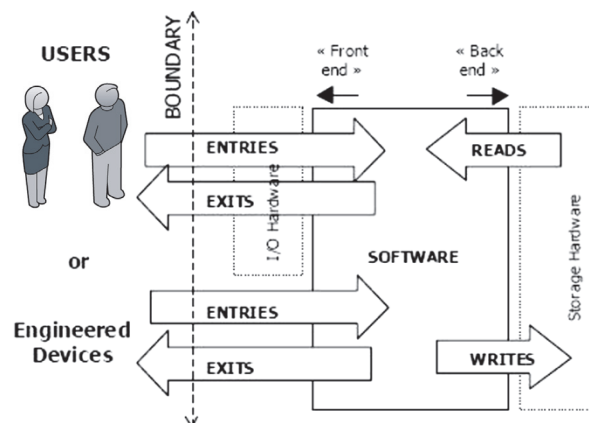
Pros

- Very good estimates of function points can be calculated from requirements and architecture.
- Has been used successfully in every type of system.

Cons

- Counting FP is not a natural part of the process.
- Requires significant training and effort.
- There are few automated tools and they work only in very specific domains.
- $F(A+B) < F(A) + F(B)$ so defining the boundary is critical.
- Not quite so good for math-intensive systems.

Alternative: COSMIC-FPP



Ref: Berg, K., et. al. "Functional size measurement applied to UML-based user requirements," *Proceedings of the 2nd Software Measurement European Forum*

COSMIC Functional Size Metric

ISO Standard

Provides consistent results between counters

Good results counting from use cases and requirements scenarios

Very time consuming to count from code

Summary

No one definition of size fits every use. It should reflect a natural aspect of the product or process.

- Different entities and attributes offer opportunities for measuring size.

Standardize the size definitions.

- Operational definitions are a must for having consistent data and creating actionable insight.

Useful for **normalizing** many measures.

- Comparing performance
- Expected productivity, defect density, review rates,...



Module Outline



- Elements of scope
- Monitoring progress
- Thresholds for change

Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

2

Progress and Change Management

Goals

- Track scope completion progress
- Recognize changes in scope and prepare the case for change

Questions

- What is the likely completion date we can estimate?
- How much schedule and cost contingency remains for change orders?
- How do we check progress of individual teams?
- Can we forecast completion dates by team? (Why do we care?)
- How do we consolidate progress across teams?

List of indicators

Requirements changes

- Allocation of requirements to teams

Scope progress

- By team; by intermediate work products; by tasks

What to Track

1. Any deliverable used outside the project team.

Includes usual components (software, platform, etc.), technical documentation, demonstrations for outside executives and customers and such things as test kits for other organizations.

2. Intermediate deliverables you want to track

Usually used because you want to estimate change effects for different teams.

Tracking Progress: a Story

Scenario: You have to move.

You have a spouse, 2 kids, and a dog. Your house has 12-14 rooms.

When you leave the house, it must be empty – not even a sock on the floor.

How do you get ready?

Taken from "Crystal Clear" by Alistair Cockburn, A-W, 2005
<http://alistair.cockburn.us/crystal/articles/evabc/earnedvalueandburncharts.htm>

The Usual Plan

1. Throw out all the junk.
2. Pack non-critical items.
3. Pack essential living items during the last 3 days.

This approach usually causes chaos at the end.

Questions

- How do you know you have thrown out all the junk?
- How do you know how much work remains?
- What does 60% complete mean? 80% complete?
- What indicators can you use?

Solution

List the rooms and sort in priority order from least used.

Pack everything in the rooms, one room at a time.

- If you find an item that you *must* have, move it to another room.
- When *everything* in the room is packed, move it all out and close the door.

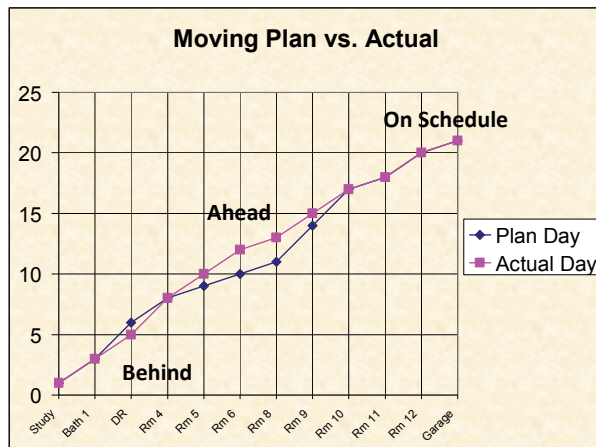
You can measure how many rooms are left and how long it took to do a single room.

Do an “Estimate to Complete” for remaining effort.

You can live out of suitcases for the last week.

Plotting Moving Progress

Room	Plan
Study	1
Bath 1	3
DR	6
Rm 4	8
Rm 5	9
Rm 6	10
Rm 8	11
Rm 9	14
Rm 10	17
Rm 11	18
Rm 12	20
Garage	21



Progress Values

Insight into the technical work

Measure of size and completion target

Change threshold

Always compare progress to the whole and the plan.

- Each team should monitor its own progress.
- Project manager may assess risk by team.
- Summary progress is important to project team as a whole.

We must also check the quality.

- Remember: ***Not even a sock!!***

Team Progress

A single team usually works a single deliverable at a time.

Operationalize the definition of measuring progress:

- Needs either a size count or quality check or both.
- Both the phase of work and the deliverable observed are important

Tracking opportunities:

- Inspection or static analysis tool or code counting tool
- CM System for components in build
- DR system for defect closure
- Test for test cases passed
- Requirements tracking for requirements accepted (by build)

Operationalize Progress Measures

By type of work:

- Requirements: accepted for current build
- Design: documents or tool outputs
- Code: size, usually in CM system and passed unit test
- Testing: number of test cases developed
- Testing: integration (or system) test cases passed
- Testing: defects remaining open < threshold

Indicators for Progress and Change

Chart should have three types of data:

1. Estimated target value (range) that should be adjusted with changes
2. Planned trend line
3. Actual observed completion size or count

Change Threshold

- Maximum allowable change in target before change is mandated.

Change Record

- Show change by adjusting threshold or target line.

Consolidating Teams

Method 1: Stoplight Chart

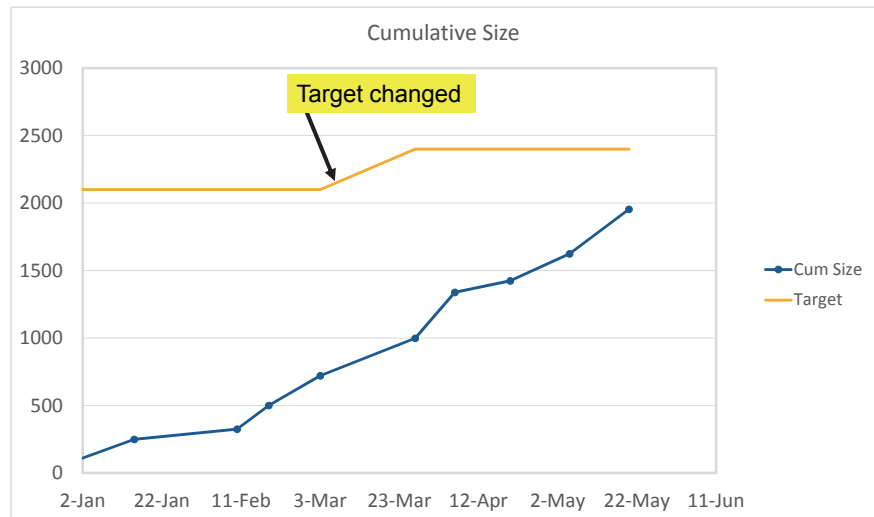
- Check within planned range or not

Task Name	Progress	Duration
Task 1		5 days
Task 2		5 days
Task 3		4 days

Method 2: Allocate percentages (more work)

- Component scope is assigned a % of total product size
- An area chart or stacked bar chart shows completion of total.
- Usually used with stoplight to draw attention to late teams.

Tracking by Size from CM System



Software Engineering Institute | Carnegie Mellon University

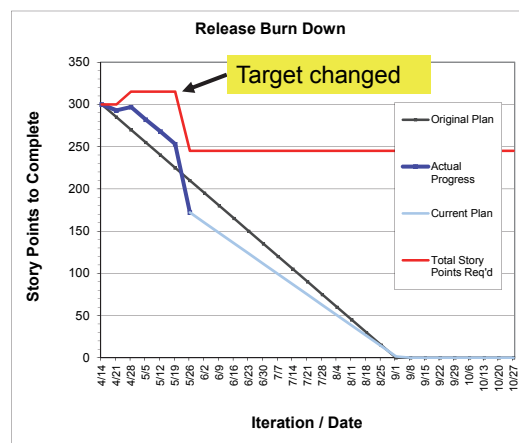
Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

15

Burn Down

Agile teams usually use a burn down chart for progress – No visibility into tasks or process. Do you care? Why?



Software Engineering Institute | Carnegie Mellon University

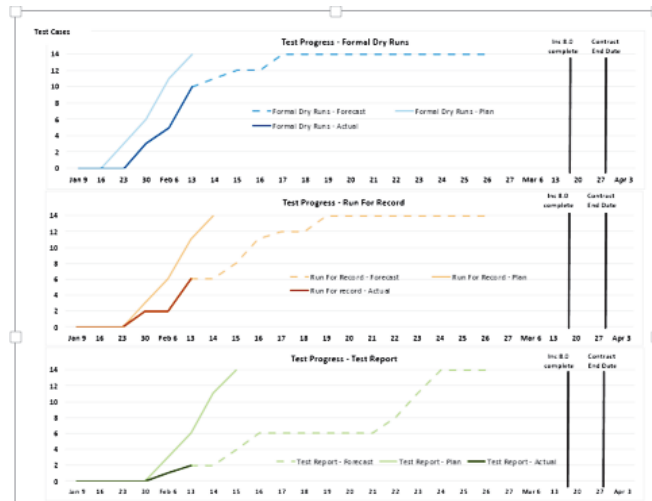
Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

16

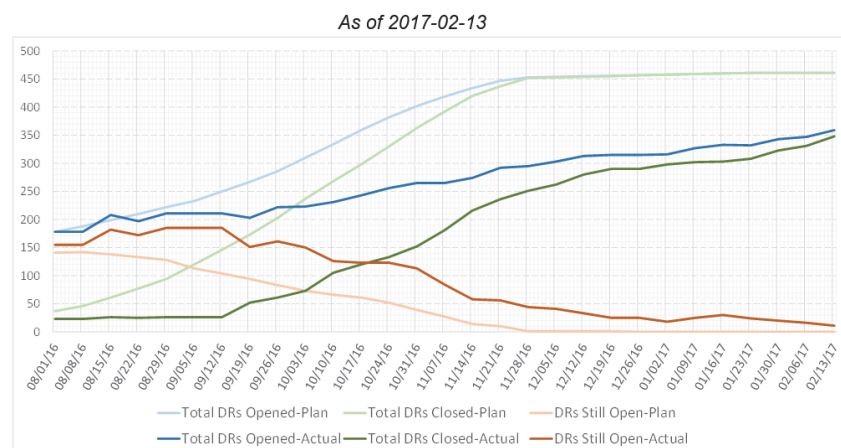


Progress Measured by Testing



Using Defects to Track Progress

Chart is missing threshold value



*Target: Still Open<10

Chart Type for Progress

Description

- Excel Chart type is Scatter (X Y)
- X-axis is a set of ordered dates representing date of observation
- Y-axis is size or count of things observed
- A “Target” is the estimated count (total) at the end of the project or lifecycle phase.

Track size completed compared to estimate for work products.

Track defect counts to target value for defects.



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been
approved for public release and unlimited distribution.

19

Where is technical progress visible?

When should we collect the observations?

Milestone Events

- Estimate: Staff estimates work and gets agreement to proceed.
- Design: Artifacts in CM system and validated.
- Entering, completing test.

Size Dependent (type of data recorded)

- #Requirements covered (Design inspection)
- #Pages of paper reviewed or delivered
- #Reviews completed
- #Lines of code written
- #Tests planned and documented
- #Tests performed
- #Tests passed

Should all observations be in SLOC?



Software Engineering Institute | Carnegie Mellon University

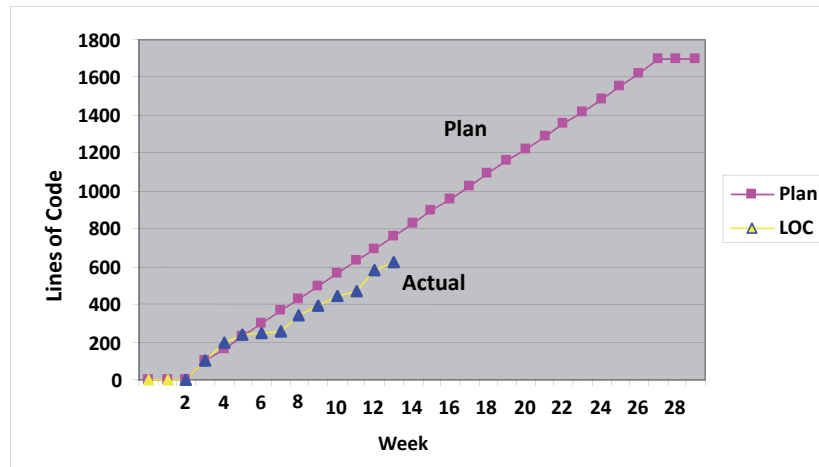
Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been
approved for public release and unlimited distribution.

20

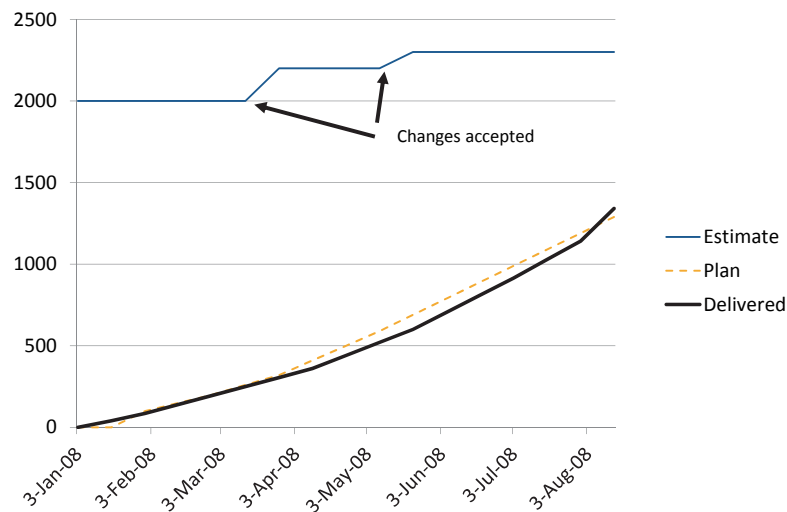


Example LOC Plan vs. Actual



Progress chart with estimate

Show Change Management on the Chart



Requirements Allocation

All requirements must propagate to:

- Software specification for component
- Design documents (or use cases)
- Code
- Test cases

“Requirements covered” provides some interesting insight.

- Does 100% coverage of design make sense?
- Is it 100% of original or current?
- What if there are changes in queue?

Design Coverage Analysis

Requirement Number	Module Name
101	A1, B2
102	A3
103	
104	C2

Calculate % = Rows with modules / #requirements

The CM System

Configuration Management check-in and promotion procedures may be used to track progress provided that the procedures include some quality check.

- Check-in means work may be awaiting quality check.
- Quality check implies fitness for use

The ability to monitor progress is one reason to place more intermediate deliverables under some configuration management.

Testing Progress

Test case coverage

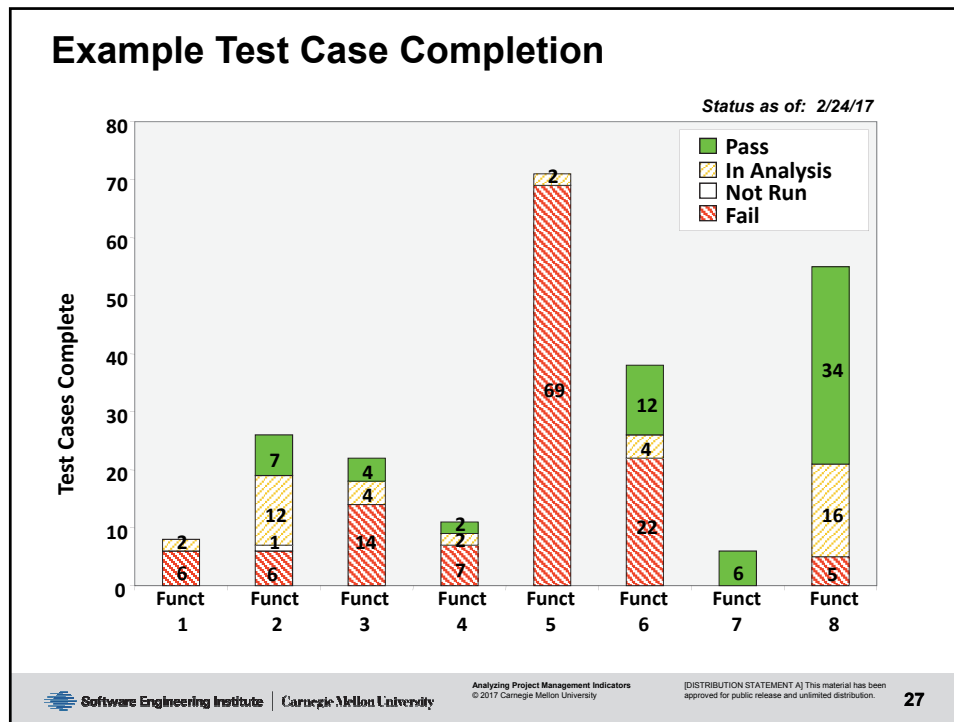
- %requirements, %specifications or %use cases
- %lines of code

Tests executed

- %requirements, %specifications or %use cases
- %lines of code
 - Typically counted with automated tool

%Tests passed (not #defects)

Remember that you need to define the satisfactory % to count for completion.

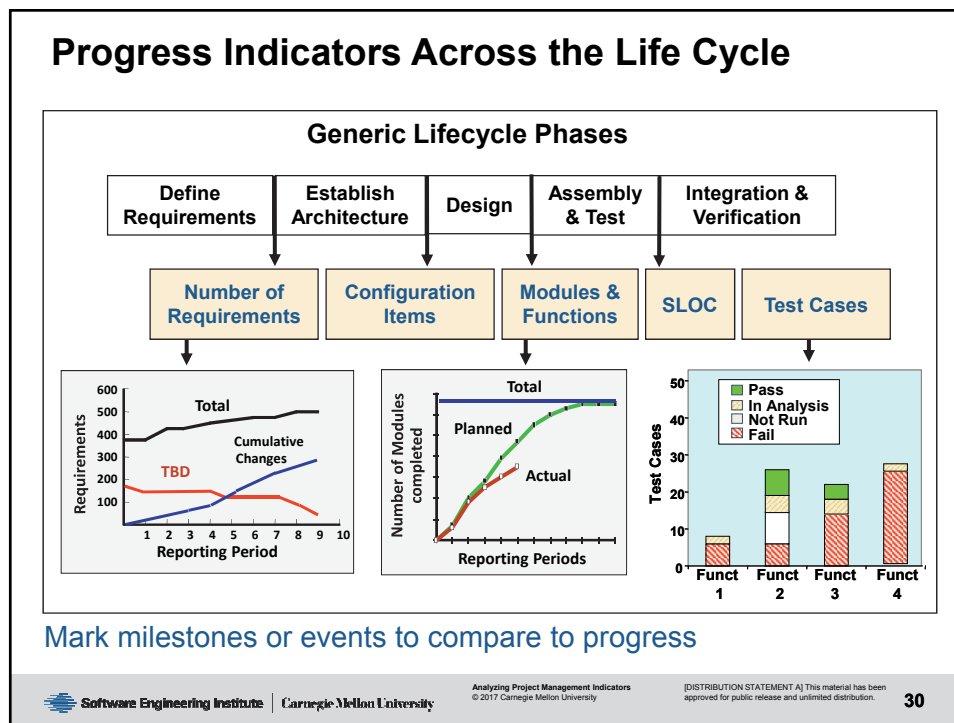
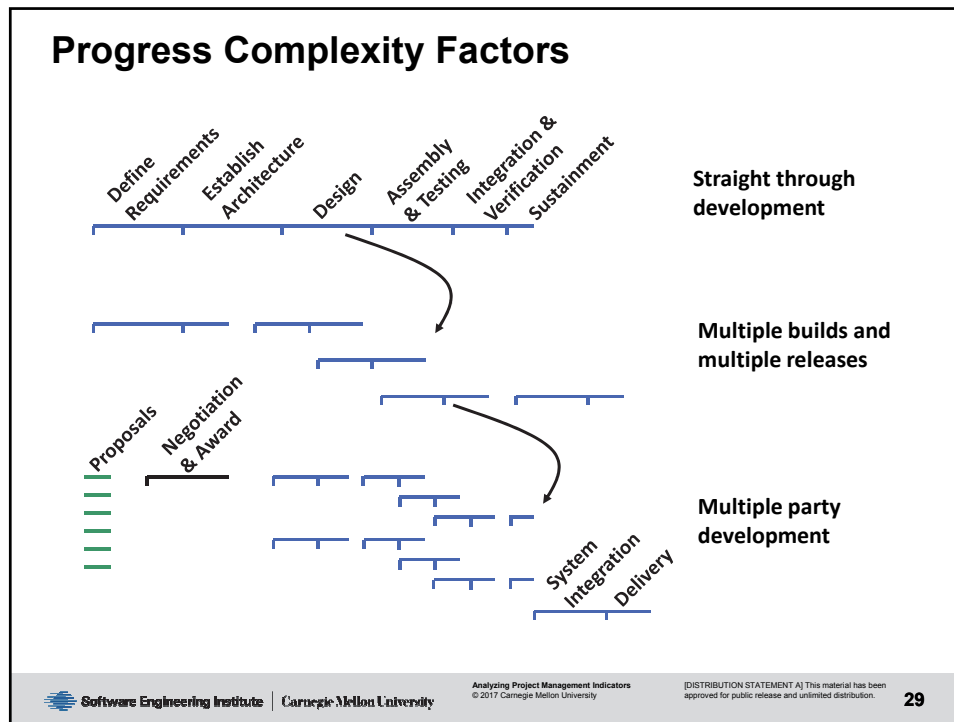


Life-Cycle Effects

The project life-cycle has a dramatic effect on progress measures.

- Different teams may be on different portions of the life-cycle at the same time.
- Reconciling the differences between teams can be the biggest challenge to creating standard measures of progress.
- SLOC works for code. What do you use for design? For test?

Either combine size measures or track them by separate teams and use stoplights for comparison.



Project Change Management

Provide evidence that change is required.

- Evidence will come from progress charts and enhancement requests.
- Make comparisons based on size of work.

Assess the state of current work.

- Completed work has a quality check.
- Make a case for incomplete work if necessary.

Present your case for a decision.

Adjust schedule and resource commitments.

Project Changes Look Backward

A change requires you to re-open something you thought had been completed.

New requirement: The software/hardware specification was derived from the requirements. A new requirement means those specifications must be adjusted.

Always identify the chain of deliverables that must be fixed when doing change management.

Summary

Track both milestones and completed deliverable size.

Show estimated totals on deliverable size charts.

A Team should plan and track its own progress and performance.

Project tracks

- aggregated progress
- team performance to plan
- aggregated performance

Problems come when any team cannot meet its performance plan.



Module Outline



Different Quality Values?



Product Manager
Focus = market acceptance



Project Manager
Focus = schedule & resources

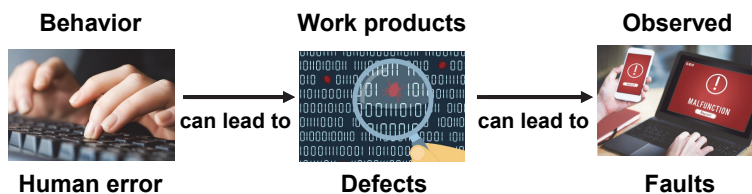


Engineering
Focus = quality of work & team

Basic Model

Humans make mistakes, which escape into work products and later phases of work and are observed in execution or during other change work.

Collectively, these are called software defects.



Development Process Affects Product Quality

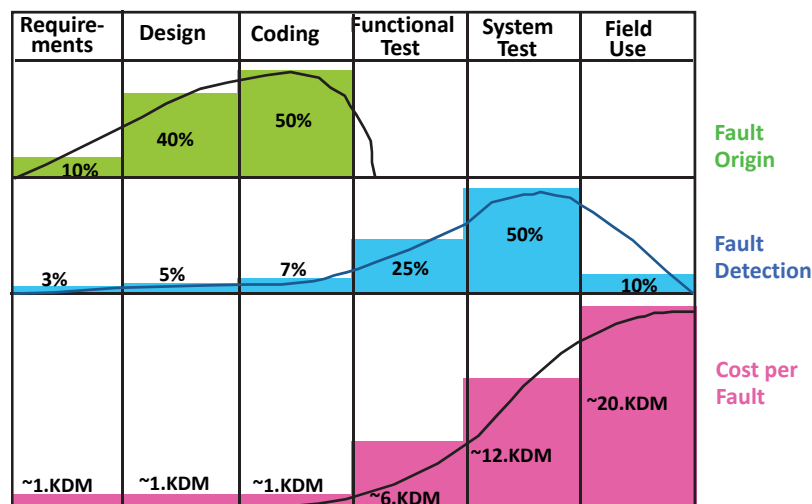
We have two types of Goal statement:

- Development tasks produce as few defects as possible.
- Work products are validated for performance when possible.

Questions:

- Which development tasks are likely to produce defects?
- Which practices prevent defects or identify and remove defects?
- How many defects do we estimate in the resulting work products?
- Which work products can be validated for what performance attributes and for which users?
- How many validation exercises can we afford?
- Do validation exercises identify problems?

Faults as a Cost Driver



[REF: Paulish 94]

Effects of Poor Quality

Poor quality during development generates rework.

- Every phase of work activity generates some defects. How many?
- If the defect is discovered after a handoff (next person), then there is rework.
- Typical development organizations spends 30% of development effort on rework.

After delivery, fielded defects become maintenance tasks.

- After about 3 releases, many organizations spend upwards of 70% of development effort on rework (no value added). [Capers Jones]

Development process and product quality are interlinked.

Reducing the Cost of Quality

“Cost of Quality” was coined by Crosby in “*Quality is Free.*”

We can either

1. Reduce the likelihood of creating the faults, or
2. Improve the likelihood of finding them early, or
3. Reduce cost to fix

Cost of Quality = Cost of Non-conformance to Quality
+ Cost of Conformance to Quality

CNQ = costs for repair, support, warranty, rework, waste

CCQ = cost for appraisal, testing, audit, inspections, etc.

Product Quality

Goal

- Helps users deliver improved value to their customers.
- Shows product validation results

Tracking Measures

- Measures of product performance
- KPP and TPM test results, including simulations
- Results of successful/unsuccessful product validation tests and reviews

Questions

- Will the product achieve desired quality and performance?
- Do validation checks prove the value-in-use potential?

Decision Potential

- Risk reduction activities.
- Update test cases and validation cases.
- Renegotiate performance target.

Schedule, Cost, & Resources	Effective Process
Scope, Progress, & Change	Quality

Measures of Product Quality

Project Level

- Scenarios for product validation are defined and accepted.
- Technology readiness
- Architectural review (e.g. *ATAM and *AADL)
- Summary of product verification and validation results (e.g., tests passed)

Team Level

- Defects removal target achieved
- Current test results of key performance criteria
- Quality thresholds evaluated (e.g. churn)

*ATAM: Architectural Tradeoff Analysis Method

*AADL: Architecture Analysis & Design Language

Dozens of “Quality Attributes”

System Availability for different classes of user

Response time for class of user

Functions work under what conditions of failure?

- Operator failure, power failure, rain, damage to antenna, bad input data

Safety

Cyber security

Effective and Efficient Process

Goal

- Process produces few defects and removes most.
- Process uses resources efficiently avoiding rework.

Tracking Measures

- Quality assurance audits (processes performed as expected)
- Performance measures are within expected ranges
- Behavioral observations (team, technology)

Questions

- If processes are not followed, why (e.g., behavior, training, problem)?
- Is performance data available and reviewed when the data suggests?
- Are technology changes reflected in process changes (e.g., training)?

Decision Potential

- Demonstrates the need for process change or behavior intervention.

Schedule, Cost, & Resources	Effective Process
Scope, Progress, & Change	Quality

Measures of Effective and Efficient Process

Project Level

- IPT members actively participate on action items, interfaces, etc.
- Monthly project QA report
 - # of quality checks by type and completed as scheduled
 - # of audit findings and # of defects found
 - risk associated with findings and any mandatory corrections
- Defect processing/rework analysis

Team Level

- Selected process performance measures (e.g., inspections)
- # of quality audit findings (e.g., process and product errors)
- Quality events skipped or delayed and defect escapes
- Defect processing/rework analysis

Discuss

What are defect detection processes?

Which are used in your organization?

Do the methods detect defects “in-phase”?

Tools

Types of testing

Types of reviews, inspections, etc.

What is your cost of quality?

Some organizations have started to calculate this by accumulating the cost of maintenance releases.

Operationalize: What should we measure? – 1

Defect discovery information

- What work product was under study?
- What was observed and when?
- What activity identified the problem?

Fix priority

- Severity (impact): characteristic of type of failure
- Priority: when to fix

Tracking the fix process

- Dates on change of state
- What was fixed and how?

What should we measure? – 2

Change backlog

- How much is in queue but not scheduled?
We will use this if we have to change the project plan.

Process effectiveness (why?)

- Training, process and common language
- Inspections, tests and validation
- Quality assurance
- Tracking of causes from later defect discovery

Should we track the cost of each fix?

- (May be sufficient to track or estimate the total).

What should we measure? – 3

Product Quality -- Tests Passed

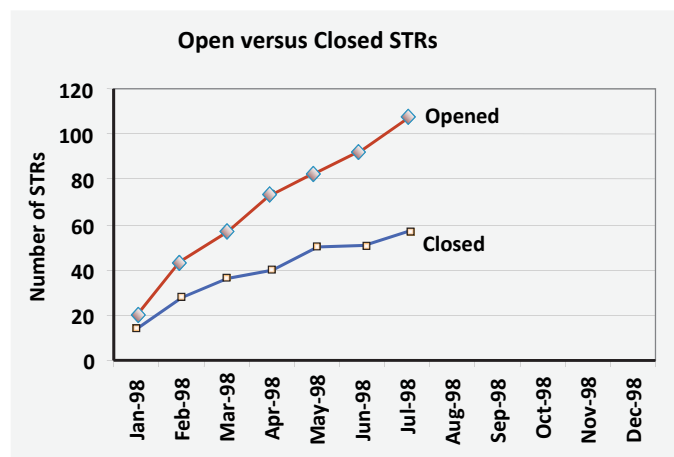
- The important tests for product quality are operational testing.
- We must anticipate these tests by using scenarios to evaluate intermediated products.
- Some methods are AADL (Architecture Analysis & Design Language) and ATAM (Architecture Tradeoff Analysis Method)
- Also an Analysis of Alternatives (AOA)
- “First Article Inspection” (manufacturing)

Risk Reduction

- Technology readiness
- Manufacturing prototypes, first

Defect Tracking Chart

We often see this chart. What is it telling you?
What else do you want to see?



*(STR): Software Trouble Reports

Charting Rework and Product Quality

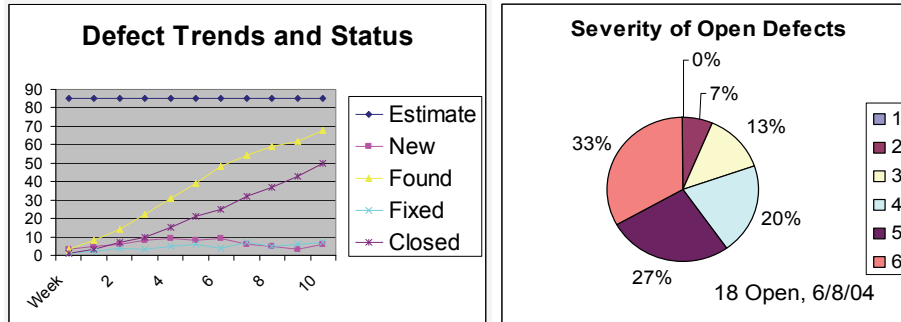


Chart has 2 types of information:

- 1) Found and closed (status)
- 2) New and fixed (weekly rates)

The pie chart shows the total of "open" defects. The pie tells us about the effect of defects on the user.

Defect Detection Process Efficiency

Why do you want the engineers to have these charts?

You have to think about what can go wrong and how the chart might suggest a problem.



CM Changes Identify Risk

Risks:

- A team has a process that involves a lot of rework.
- A selected module is complex and seems to involve a lot of rework.

What would you measure and display?

Cautions:

- We do not want teams to avoid the CM system.
- We do want to see that teams follow their process.
- Avoid measuring individuals – emphasize the team.

Component Churn

Identify problem components or teams.

Configuration management (CM) can show “churn”.

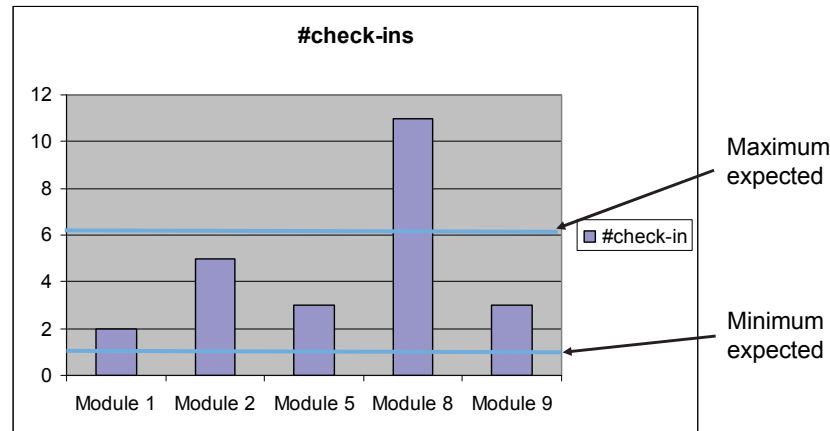
$\text{Rework} = \# \text{ of check-ins} - 1$

or $= \# \text{ promotions} - 1$ (if unit test does not count)

Churn is evidence of a component that has seen extra rework.
Rework describes work you thought was completed, but some portion had to be re-done.

Use CM to store requirements, design, test cases, etc.
Again you can see evidence of rework via CM system.

CM Churn Based on Check-in



Current average = 4.8
(Numbers here are made up.)

The Defect Management Process

Open, Closed

- What does "open " mean to you?
- What does "closed " mean to you?

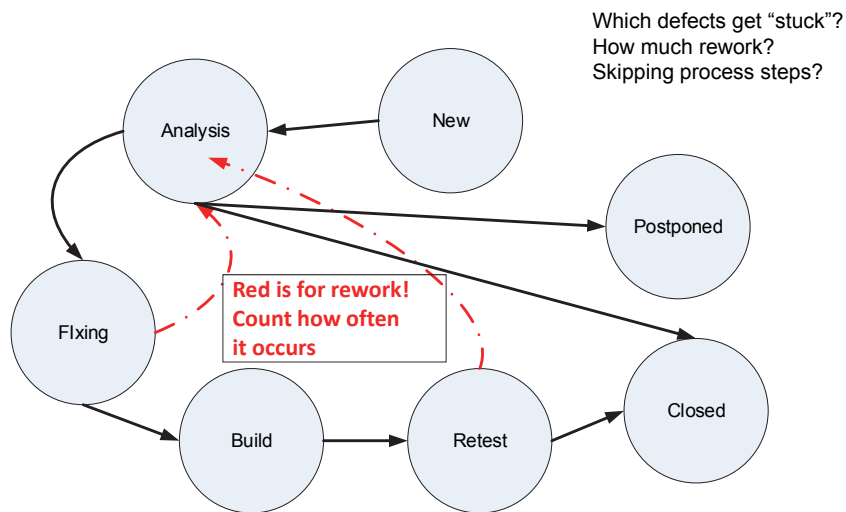
Different organizations and different people have different definitions based on experiences, processes, and situations

The problem status definition checklist provides an operational method for making these terms clear and precise

Defect management system provides tracking information.

Defect Management Process

Use term “State” to describe the type of work occurring now.



Problem Report / Defect Considerations

Attributes	Definitions	Counting & Reporting
Problem status	Open - Closed	Time - opened - closed ...
Problem fix type	SW, HW, etc.	
Uniqueness	Original, duplicate	
Criticality	1st level, 2nd, etc.	Originator
Urgency	1st (highest), etc.	Environment
When found	Life-cycle	Where Found - component level - type of artifact
When introduced		
Finding mode	testing, inspection	

REF: Florac 92

Fix Type Attribute

Problem Type	Include	Exclude	Value Count
Software defect			
Requirements defect			
Design defect			
Code defect			
Operational document defect			
Test case defect			
Other work product defect			
Other problems			
Hardware problem			
Operating system problem			
User mistake			
Operations mistake			
New requirement/enhancement			
Undetermined			
Not repeatable/Cause unknown			
Value not identified			



Discussion

We suggested measuring these types of information.

- defect discovery information
- fix priority
- tracking the fix process
- change backlog
- process effectiveness

As a project manager, what chart(s) would you want to see to check on schedule and schedule risk?

What charts will you watch for product acceptance risk?



Definitions: Purpose of Testing

Verification

- Confirmation by specific evidence, that certain requirements and specifications are fulfilled.



Validation

- Confirmation that the customer's requirements are covered and the product is fit for use in its intended environment.

Testing Process Objectives

Project Goal: Verify the system works

Test Team Goal: Deliver the test results
Document test failures
Estimate next time

Test Team Objectives

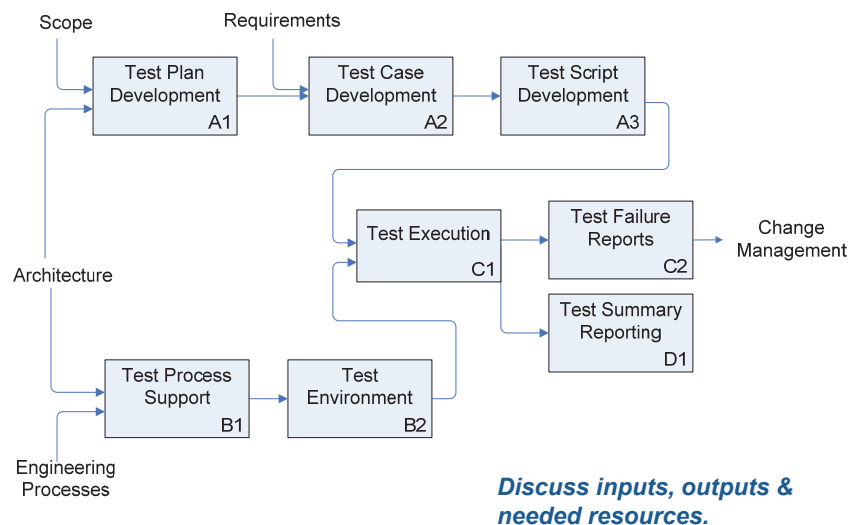
- Cover as many requirements and specifications as possible.
- Minimize escaping defects (from test).
- Maximize testing throughput (complete the job quickly).

What would you measure?

Processes Associated to Testing

- A. Development of test cases
- B. Process updates to accommodate technology changes
 - “How might this change affect how we should test?”
 - For verification? For validation?
- C. Test execution and failure analysis
- D. Summary reporting
 - Functions passed
 - Performance goals achieved
 - Defect density vs. target or other forecast

Project Testing Processes



Testing Decisions

How many test types do we need? How should we resource and staff?
When do we perform each type of test?

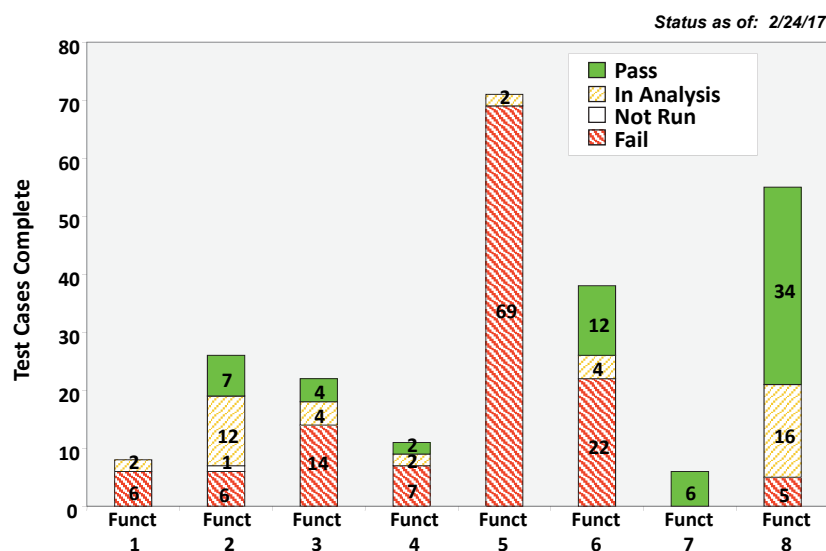
How many test cases do we need? How many defects do we expect?

Have we tested enough? Have we found sufficiently many defects? Have we covered the customer's need?

How long should we say it will take us to complete testing?

Some are planning and some are execution.

Requirements Coverage



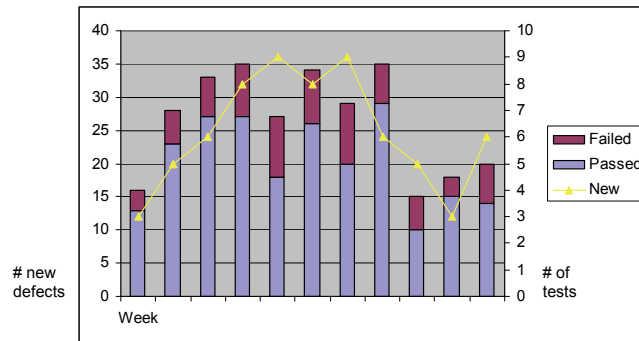
Maximize Testing Throughput

Are we executing as many tests per week as we planned?

- Since “size” of tests may differ, may measure %plan.

Are tests finding defects?

- If not, we may need to examine the test cases.



“New” – defects discovered this week; plotted with right-hand axis.

Process Characteristics

Cycle time to load & verify build runs (smoke test).

- Good idea to monitor this because it may cause a reduced level of testing throughput for a day or so.
- PM may want to see that development team is handing a reasonable product to the test team and not just delivering to the deadline.

Effort to document and track defect

Restarted test cases

- Test manager should watch this number. It represents the testers' level of rework.

When Is Testing Finished?

Test coverage indicator

- Tests cover the requirements
- How many have passed? How many more are needed?

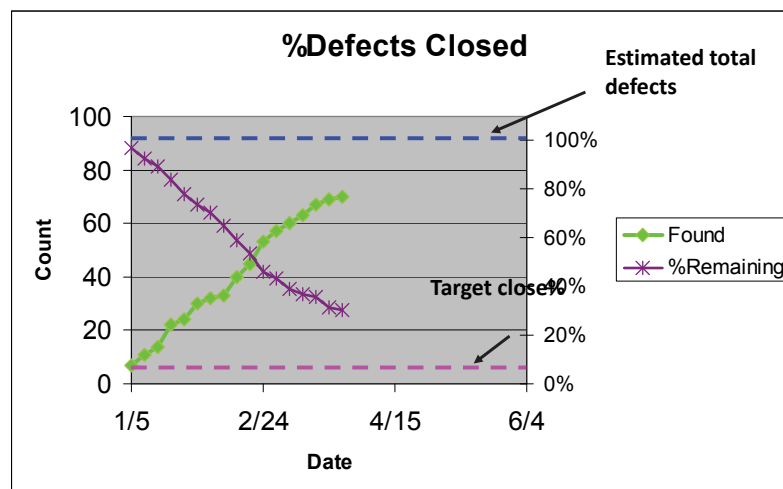
Defect density or Quality Target

- Provides a surrogate for Mean-Time-To-Failure (MTTF).
- Defect density compares estimated defects to size.

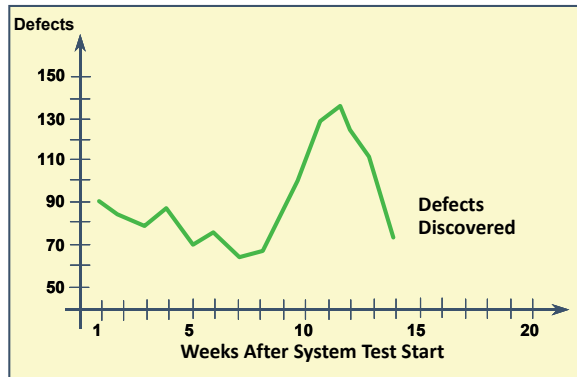
Find rate

- *If* there are sufficiently many good testers, the find rate provides another MTTF indicator.

Testing Quality Target Indicator

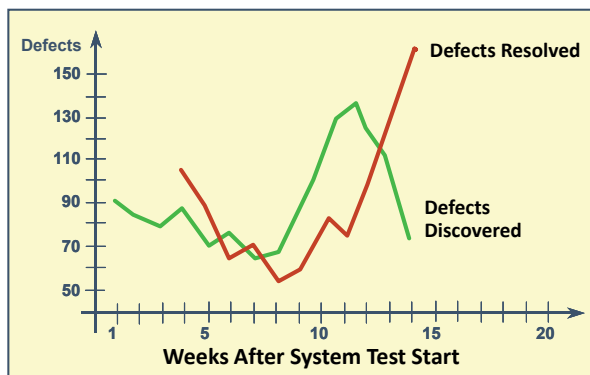


Are We Ready to Ship?



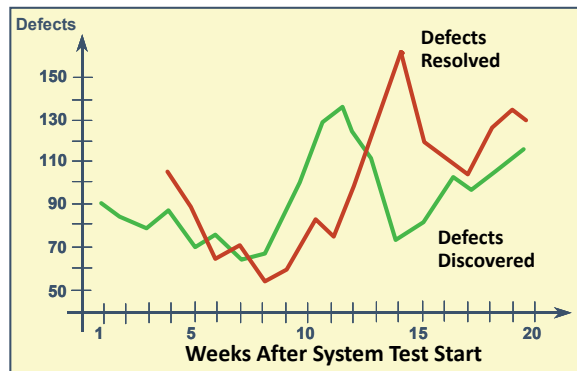
Would you consider shipping at week 20?

But How Many Were Resolved?



Now would you consider releasing at week 20?

If You Don't Look, You Won't Find Any



Testing Summary

Testing can provide a wealth of data.

As project manager, you are concerned about

- Testing process throughput
- Meeting project quality targets
- Making and keeping schedule commitments

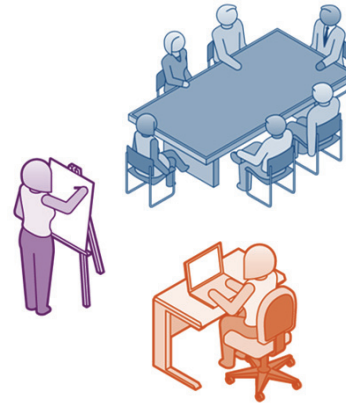
The test manager should be able to show you that he has an efficient and effective process as well.

Process quality assurance

Project manager does not often have the main responsibility for the engineering process.

However, he needs visibility into the development process in order to understand certain risks.

Quality assurance is supposed to explain the process risk.



Three viewpoints

Quality Analyst Report

List of quality interventions in period.

Overall performance ratio and trend.

Specific non-conformance list.

Risk assessment of non-conformance.

Without the risk assessment, you may be fixing unimportant problems. Make the QA analyst be a useful contributor to the project team.

An overall downward trend is symptomatic of other problems and pressures. ***Pay attention!***

Sample Quality Analyst Report

List of quality interventions in period.

- 4 checklist audits
- 12 inspections

Overall performance ratio and trend.

- A single audit of CM had one non-compliance
- All inspections were within expected ranges

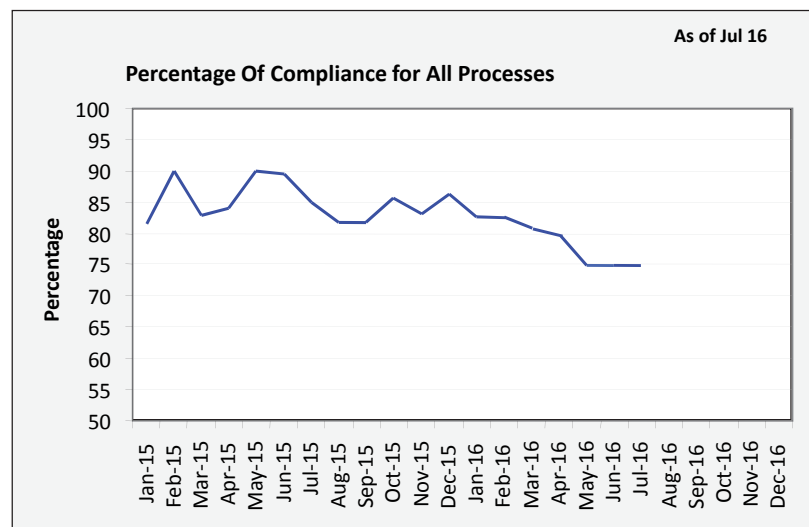
Specific non-conformance list.

- One team did not follow naming convention for modules and external variables.

Risk assessment of non-conformance.

- Recommend correction to source code compliance as this may cause future testing and design problems.

Process Audits



Process (Quality) Assurance Summary

Quality Measures highlight specific project risks.

Defects represent a **change backlog**.

- We must either fix problems, remove something from scope, or deliver a problem to the customer.

Defect management provides a lot of information.

Existence of defects suggests a process problem.

- Where is the problem?
- Fix the defects or fix the process?

A quality product cannot be expected without checking the quality of the processes, so **you must have QA!**

Dashboard Process Effectiveness Quadrant

Varies with phase of the development lifecycle.

Defect backlog and correction.

Rework such as churn via CM, DOORS, etc.

Quality Assurance audits

Process performance of critical processes.

- Defects created/removed by process.
- Outlier analysis, 6-sigma methods, etc.

Dashboard Product Quality Quadrant

Architecture tested by scenario analysis

Test cases cover user, maintenance, failure and logistic scenarios based on performance objectives.

New technology is ready for use (product and process)

Risk reduction is accomplished

Record of validation tests passed.

Capture/Recapture Exercise

Fun with Food

Background

In field biology experiments, it can be extremely difficult to know how large a population is. In the case of analyzing fish in ponds, biologists have tried this method.

- Count the number captured and tagged.
- Collect and count a second time.
- Calculate (tagged fish/total) in 2nd sample.
- Use the ratio to estimate the total population

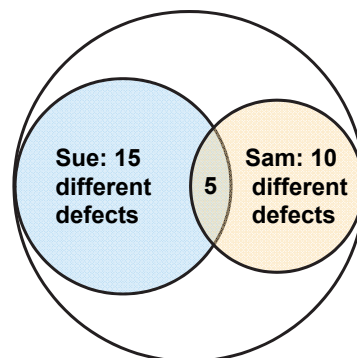
This method can work for parallel or serial reviewers.

Capture/Recapture Example – 1

Estimates remaining defects

Example:

- two engineers inspect the same software artifact.
- **Sue** finds 20 defects (20 captured) but does not fix the artifact
- **Sam** finds 15 defects in the same artifact and 5 of the defects are the same as 5 found in the group found by Sue (5 recaptured)
- How many defects are remaining?



[Humphrey 2000]

Capture/Recapture Example – 2

$\frac{1^{\text{st}} \text{ Sample (all marked)}}{\text{Total Population}} \approx \frac{\text{Marked in } 2^{\text{nd}} \text{ Sample}}{\text{Size of } 2^{\text{nd}} \text{ Sample}}$

5 defects were recaptured from the first 20 defects found by Sue

- the 20 defects found by Sue are considered a representative sample of the population of defects
- the 5 defects recaptured by Sam mean that 25% ($5 / 20$) of the defects have been found

This information can be used to predict the whole population of defects

- Total defects * 25% = 15 (captured by Sam)
- Total defects = $15 / .25 = 60$

The remaining defects can now be found

- Remaining defects = Total defects - defects found
- Remaining defects = $60 - (20 + [15 - 5]) = 60 - 30 = 30$

[Humphrey 2000]



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

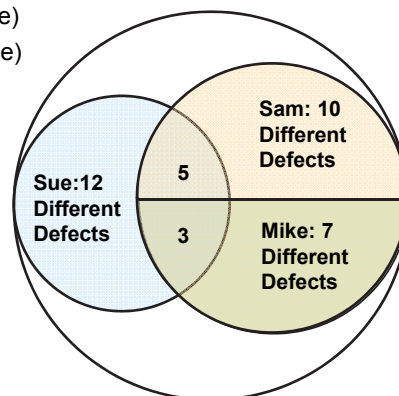
53

Capture/Recapture Example – 3

An inspection was performed on the same software artifact by three engineers: Sue, Sam, and Mike:

- Sue found 20 defects
- Sam found 15 defects (5 the same as Sue)
- Mike found 10 defects (3 the same as Sue)

(Combine Sam and Mike's defect counts for a total of 25)



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

54



Defects Remaining Example – 4

8 defects were recaptured from the first 20 defects found by Sue

- the 8 defects recaptured by Sam and Mike mean that 40% ($8 / 20$) of the defects have been found
- total defects = $25 / .4 = 62.5$ defects
- defects remaining = $62.5 - (20 + [15 - 5] + [10 - 3]) = 25.5$

Assumptions:

- population is homogeneous and randomly distributed
- there must be an equal chance of finding every defect
 - some people are diligent and some are not
 - all defect categories must be included
- the % of defects found must be reasonably large
- count defects that only effect the program's operation (not those that are only standards violations).

[Humphrey 2000]



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been
approved for public release and unlimited distribution.

55

Capture/Recapture Exercise

Instructions: (There are 4 counts.)

- pick out about half M&Ms from the cup and count them ($A = ?$).
- mark these M&Ms so they can be distinguished from the M&Ms in the cup.
- return the "marked" M&Ms to the cup.
- shake well to ensure all of the M&Ms are mixed together.
- without looking, pick out about half M&Ms from the cup ($B = ?$).
- count the marked M&Ms X and calculate $r = X/A$.
- use the formula to estimate the number of M&Ms in the cup:
 - Population of M&Ms = B/r
- empty the cup and count all the M&Ms. How close was the estimate?
- calculate the error

Error = (Actual number - Estimate) / Actual number



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been
approved for public release and unlimited distribution.

56





Module Outline



Some planning objectives & questions

The estimation process

Some estimation methods

Planning from the estimate

How do you know the estimate is OK?

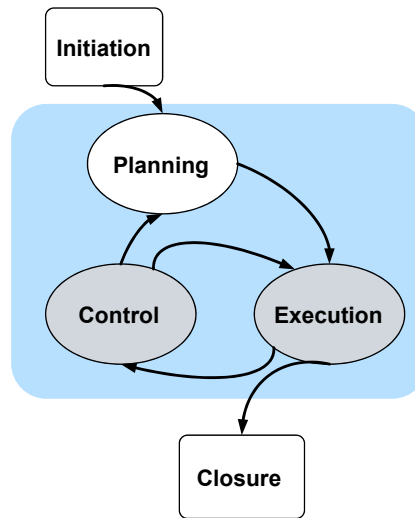
What's in a plan?

Project **Initiation** provides targets for

- Budget
- Schedule
- Scope
- Quality

Planning

- Estimate and allocate
 - Resources and schedule
 - Across project life cycle
 - To meet project goals
- May require some negotiation!



Identify some different plans

A Plan Is More than the Schedule

Scope: what will be delivered and to whom

Lifecycle: intermediate deliverables

Requirements: completion and acceptance

Resources: matching skills to needed activity

Schedule: sequenced list of activities and milestones

- contributing to deliverables
- communicating and coordinating
- monitoring and controlling

Work packages: how you will account for the work

Planning Questions – 1

Product Size, Performance, and Quality

- Are requirements feasible in this organization?
- Do alternative designs meet the need?
- What hardware capacity is required?
- What resources are required for development and support?
- What are the technical risks?
- What is technical baseline for tracking and controlling?

Project Effort, Cost and Schedule

- Is project feasible in terms of cost and time?
- What are the project risks?
- What work commitments have to be negotiated?
- How will you divide work packages?
- What are the cost and schedule baselines?

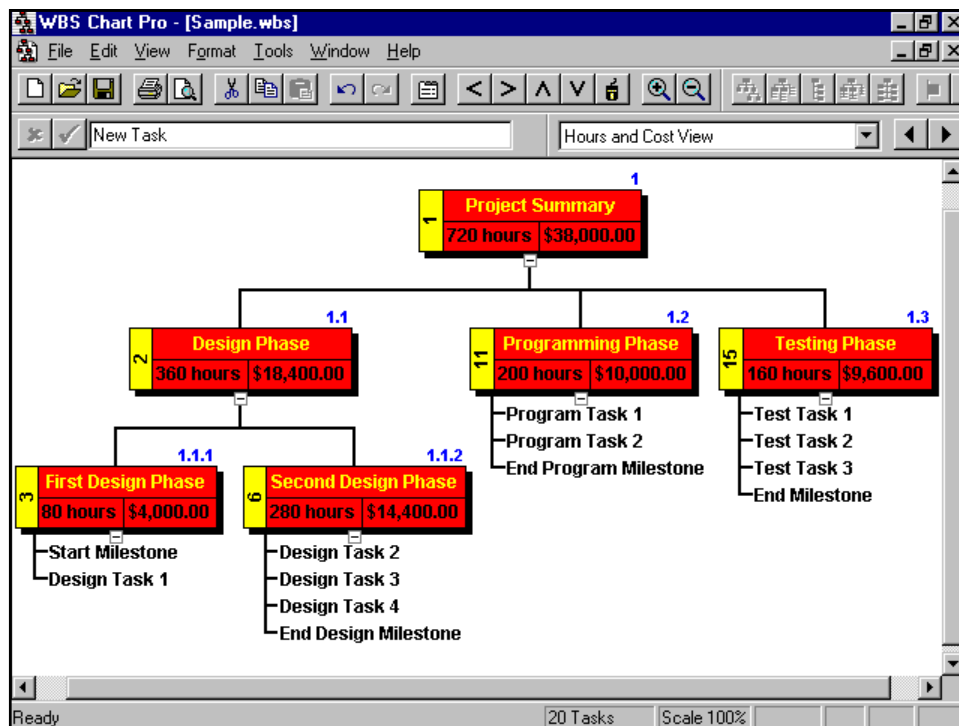
Planning Questions – 2

Process Capability and Performance

- What is the planned resource utilization and efficiency?
- What historical performance data can we use?
- What performance improvements might be needed?

Estimation informs planning and planning provides
inputs to estimation

Analyzing Project Management Indicators



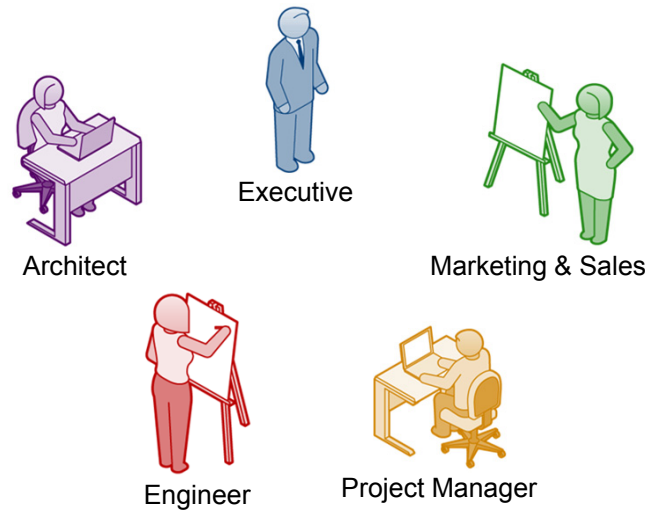
A Good Plan Should ...

- Help teams make commitments that they can meet.
- Establish agreements for scope, schedule, and resources.
- Help manager direct and coordinate the work.
- Facilitate progress tracking and reporting.
- Help ensure that key tasks are not overlooked.
- Establish targets for outstanding performance.
- Help manager identify and understand the impact of changes.

Discussion:

- *What artifacts and processes do you have that support these goals?*
- *Describe an artifact or process you think is missing.*

Different Stakeholders for the Plan and Estimate



Estimating and Planning Interaction

We are going to build a set of things and provide services:
code, documents, training

The **estimates** will tell us how much, what cost, how long.

The **lifecycle** describes how we will manage the work.

The **plan** allocates the estimates

-- size, cost, time, defects -- to the lifecycle.

The **plan** assigns work packages for tracking purposes.

We should test the estimate for sensitivity to assumptions.

Module Outline



Some planning objectives & questions

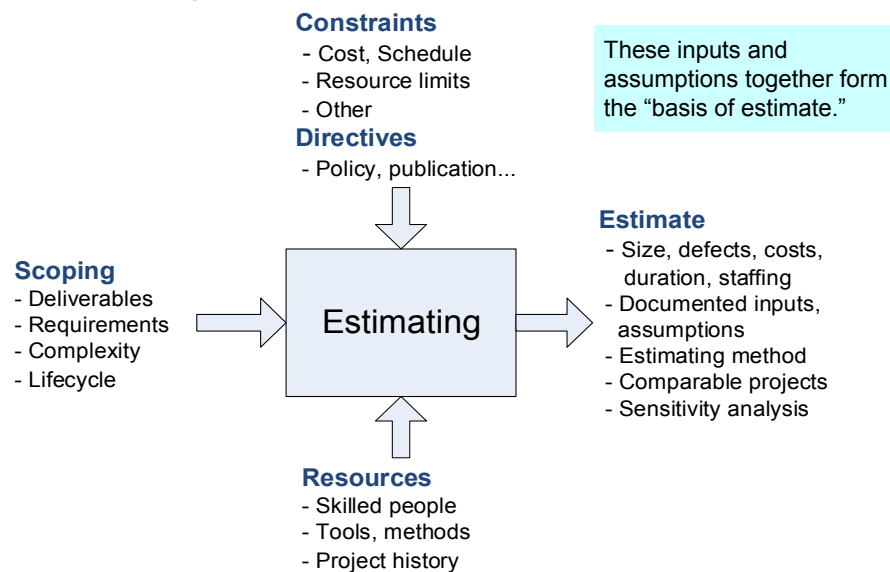
The estimation process

Some estimation methods

Planning from the estimate

How do you know the estimate is OK?

Estimating Context



Estimation Formula

The **cost** and **duration** of a project are basically a function of:

- Scope
- Team size
- Team capability
- Product quality objectives
- External factors

Measure or estimate:

- Scope by types of products and size of those products.
- Team size interacts with schedule to some extent.
- Team capability by past performance or assumptions of performance.
- Product quality objectives impact the process.
- External factors cause a range of uncertainty.

General Estimation Formula

Estimates for (Size, Duration, Resources, Defects)
are a function of the
(Requirements, Deliverables, Technology,
Staff and Process-Capability)

All the estimated values have a range of error, because all the inputs are uncertain and may also change.

What do you know about the inputs? Requirements? Deliverable scope?

What do you know about staff capabilities? About team performance? About process capability and capacity (defect removal)?

What needs to be learned to execute this project?

Since you are estimating ...

Each estimated value has a range of expectation.

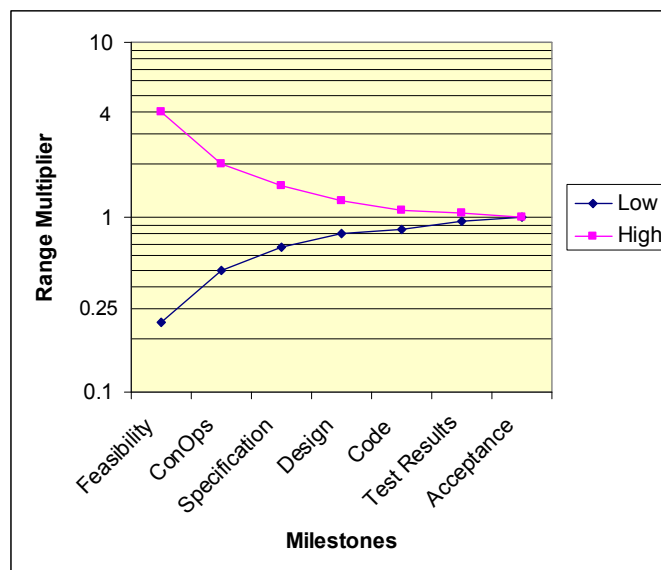
These values are related to each other
(size, cost, schedule, defects and risk),

We could estimate just one factor and derive the others, but

It works much better to estimate several factors.

Then we test the sensitivity of the overall project estimate to each factor.

Accuracy of Re-estimates Increases with Time



Sources of Uncertainty and Value of Information

Feasibility

- Types of users
- External interfaces
- Constraints

ConOps

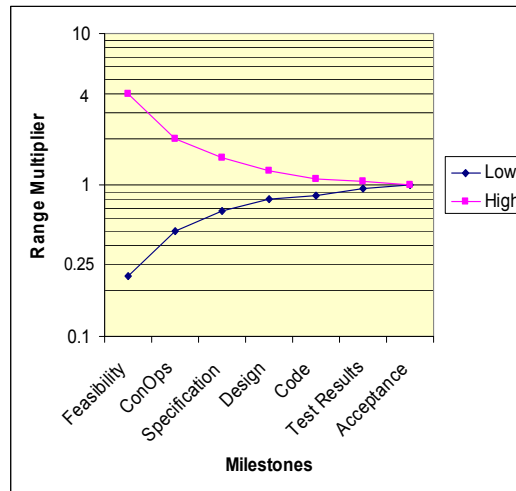
- Feasible performance bounds
- Understanding of user need

Specifications

- Secondary functions
- Storage needs
- Optimal architecture
- Developer skills

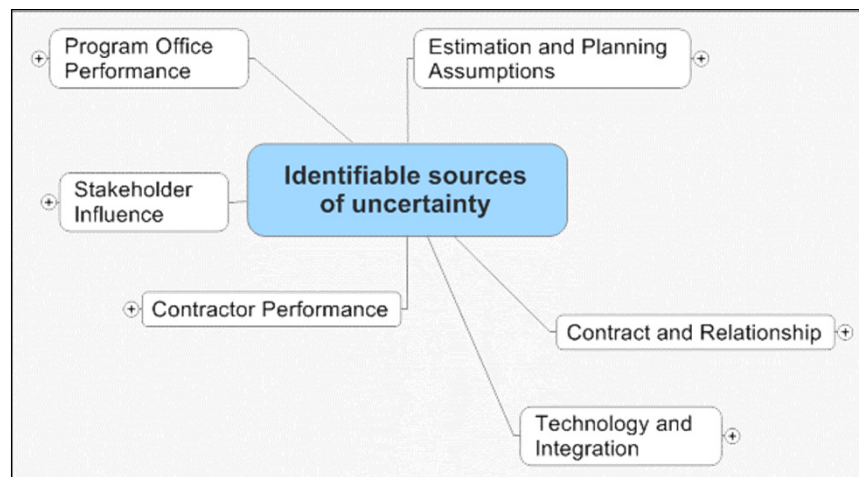
Code

- Reliability
- Achievable performance
- Tester understanding of scenarios



Identifiable Sources of Uncertainty

Things that may change during the course of development



Uncertainty

Deliverables: any chance of increasing scope? Requirements changes? A product demonstration?

Technology: is any new technology required in either product or process?

Staff: what exposure is there for staff turnover? For specialty staff who might not be readily available?

Process: is there a process that should be improved?

Dependencies: the timing and quality of components, information, or staff to be received?

Think about these questions, consider how the answers affect the estimate.

Module Outline



Some planning objectives & questions

The estimation process

Some estimation methods

Planning from the estimate

How do you know the estimate is OK?

Approaches to Estimation

We will assume that you are estimating in order to create a detailed project plan – not for a budget estimate.

- Simple estimating relationship
- Estimation by analogy
- Parametric model

Simple Estimating Relationship Questions

Identify as many intermediate deliverables as possible

- Use cases, database tables, components, etc.

Calculate resources and defects for (S, M, L) of each type

(remember the Super Simple Size measure?)

Identify how many (S, M, L) of each type

Estimate total resources and defects.

Add testing and coordination overhead to achieve quality

Use either $L = 2 * M = 2 * 2 * S$ or

$$L = 3 * M = 3 * 3 * S$$

Typical Questions for Estimate

How many tables in a (S, M, L) database project?

How many people and weeks in a (S, M, L) project?

How many test cases would be created?

In order to answer these questions, we need some historical data or documented project experience in our organization.

If we use enough of these small questions, it helps us with the project plan.

Unit-Cost Examples

Estimate unit-cost for each deliverable element

- Requirements (1 page per day, 10 requirements per day)
- Operational database (3 tables per week)
- TSP Team uses a “performance based earned value” LOC/person-week.
- XP Team: #n “story points” per week

Unit-cost works best at the team level.

- High-performing teams can describe their deliverables or services. They can calculate their own unit-cost.
- If their performance changes, there will be an identifiable underlying cause, and your plan will need to change.

Table – midpoints for each

Project Size	Small	Medium	Large
Design unit			
Code unit			
Test unit			
Effort			
Duration			

Preparing a Simple Estimate

Estimate number of intermediate work products of each time and the corresponding sizes (S, M, L) with 3x factor.

Use past history of effort for work products with the same 3x factor.

Sum up the mid-point values of work products and effort (you can also estimate the variance and standard deviation)

Multiply by a complexity factor based on number of teams and people to get duration

Or use COCOMO effort-to-schedule formula

Work Product Estimates

Element	Count	Unit-Cost	Effort hrs	Cost \$\$
Screens	19	40-h/sc	760	
DBA-Tables	7	52-h/t	364	
Reports	12	48-h/rpt	576	
Sys.Doc	60	8-h/p	480	

A "Proxy" Example

Everything translated to lines of code

Table 3: C++ Object Size Data¹

C++ Object Sizes in LOC per Method					
Category	Very Small	Small	Medium	Large	Very Large
Calculation	2.34	5.13	11.25	24.66	54.04
Data	2.60	4.79	8.84	16.31	30.09
I/O	9.01	12.06	16.15	21.62	28.93
Logic	7.55	10.98	15.98	23.25	33.83
Set-up	3.88	5.04	6.56	8.53	11.09
Text	3.75	8.00	17.07	36.41	77.66

Each method in each object is estimated for number of lines of code.

With practice, people are also able to estimate other deliverables.

Notice each successive size is ~2x the previous one.

**Taken from "The Personal Software Process" Humphrey, 2000
CMU/SEI-2000-TR-022*

Approaches to Estimation

- Simple estimating relationship
- **Estimation by analogy**
 - **Basic method**
 - **Wideband Delphi**
 - **Documenting the estimate**
- Parametric model



Estimation by Analogy

When used

- In absence of other estimating techniques
 - (relationships & models)

Based on past experience

- How many days have similar tasks taken?
- How is product size going to affect the estimate?

Potential Problems: risk of items not previously encountered

- forget an activity
- forget management reserve
- optimism in meeting cost and schedule constraints

Mitigate risks by documenting and making the rationale explicit.



Analogy

Does the work look like something we have done in the past?

Does the work look like something someone else has done?

Need to assure that the data collected is comparable

Are the assumptions documented?

Is the basis of estimate well documented?

Cautions about Historical Data

Common definitions

- What was included / excluded in the count?

Produced by similar processes

- Number of inspection processes, etc.

Collection procedures

- When in the lifecycle was the data collected?
- Does it include rework counts?

Aggregation

- Does the rolled up data include the same activities, phases, defect type, etc. that I need to estimate?

Document the Basis of Estimate

- This project has a similar number of requirements therefore delivered code should have similar size ~20K.
- This project team lacks experience with the tools, therefore we expect 20% more defects need to be found during testing.
- The time constraint, 7 months, on this project is severe, representing less than 70% of time required for the prior project. We expect delivered defects to be 2x greater.

Analogy Commentary

Analogy can be good because it typically takes less time to prepare an estimate.

The trustworthiness of an analogy estimate is derived from the correspondence of the current project to the selected analogy.

However, the data from the selected analogy must be well documented to enable its proper use including determining whether the analogy is appropriate.

Understanding the sensitivity of an estimate to assumptions and uncertainty may be challenging if the needed data are unavailable from the analogy.

However, it may be possible to do this with Wideband-Delphi by conducting an extra session for the purpose.

Method: Wideband-Delphi

Use a panel of **experienced** estimators.

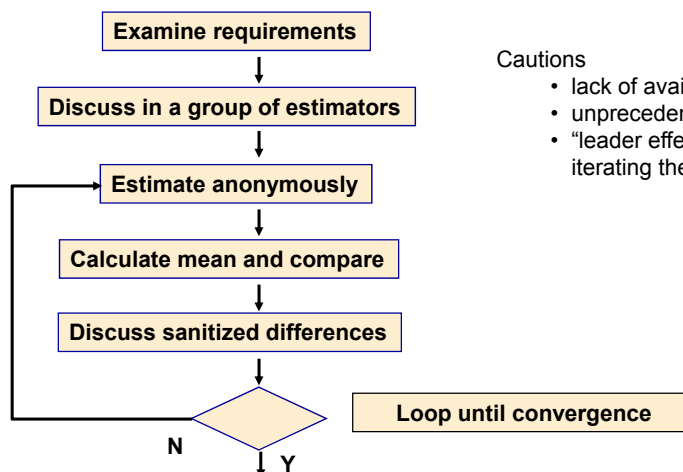
Give each the inputs to the current project.

Each estimator works independently.

Each estimate is reviewed and discussed by all the estimators together.

Iterate until consensus is reached.

Wideband-Delphi Flowchart



Cautions

- lack of available experts
- unprecedented application
- "leader effect" when iterating the estimate

Approaches to Estimation

- Simple estimating relationship
- Estimation by analogy
- **Parametric models**

Parametric Methods

All the commercial tools use parametric methods for estimating.
The basic principle is:

Inputs and Assumptions

- Project parameters:
 - Size, scope estimates
- Environmental Factors:
 - Productivity, development process, etc.

Output Estimate

- Duration, resource requirements
- Defects

These tools always require calibration to your environment.

Parametric Model Assumptions

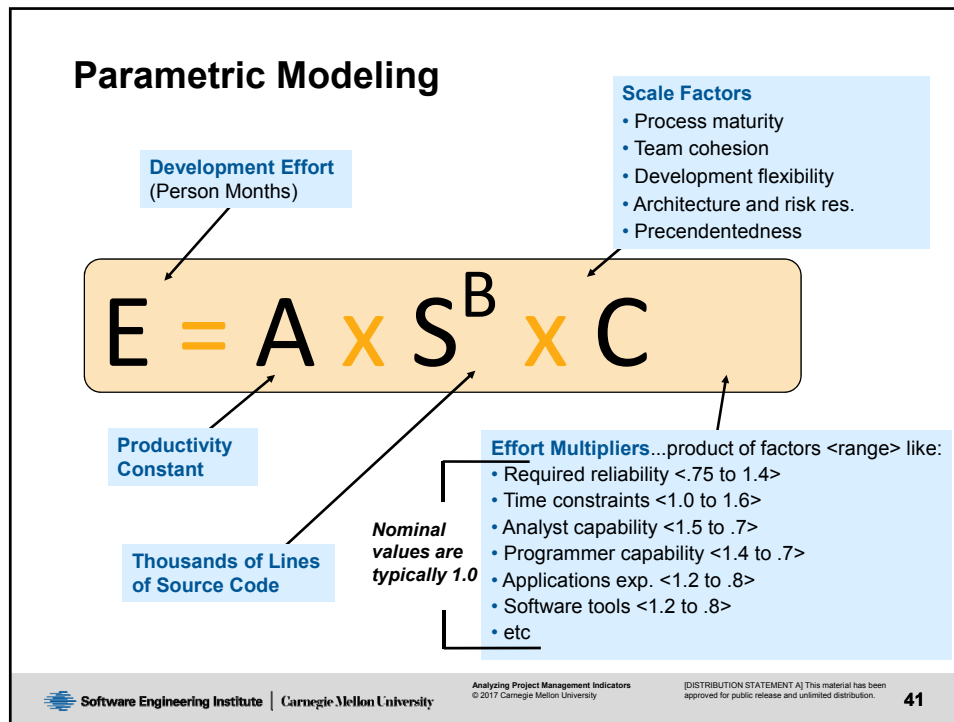
Each model has different breadth and depth

- **Breadth:** the number of contiguous phases covered in the estimate
- **Depth:** the number of activities and the effort required for each activity in each phase

Typical Parametric Model Inputs

May be as many as 300 questions to answer!

Attribute	Example Factors
Size	Source lines of code Function points
Personnel	Capability Application experience Platform and language experience
Computing resources	Time & storage constraints Platform volatility
Product characteristics	Required reliability Complexity Documentation Database size
Project characteristics	Tool usage Process maturity Schedule constraints



Calibrate Your Model!

Calibrating the model to local conditions will produce a more realistic estimate. The productivity constant on the previous slide must reflect local performance.

Use data from completed projects or early project data to calibrate the model.

- size and effort data
- activity distributions
- labor categories

Most tools come with a calibration capability built in.

A model is only as good as its data

Know What Your Model Estimates

Input to model

- Size
- Project
- Personnel
- Environmental



Parametric Models:

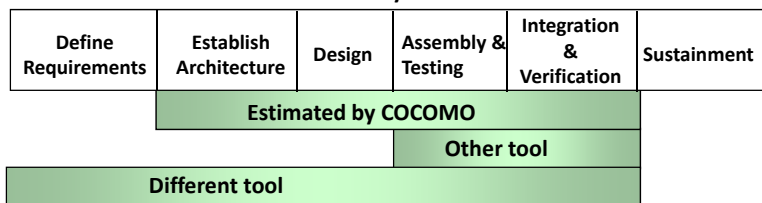
- COCOMO
- Slim
- KnowledgePlan
- Price S
- etc.*



Output

Effort
Duration
Tasking
Defects, etc.

Generic Life Cycle



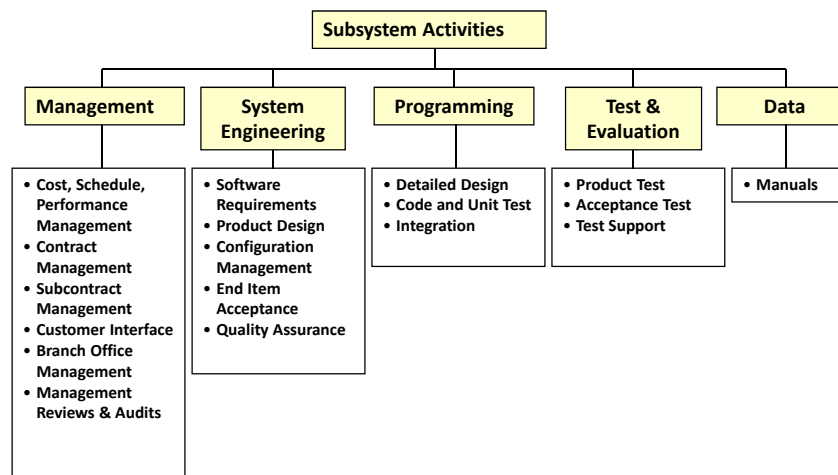
Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

43

WBS Estimated by COCOMO



[COCOMO 00]



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

44



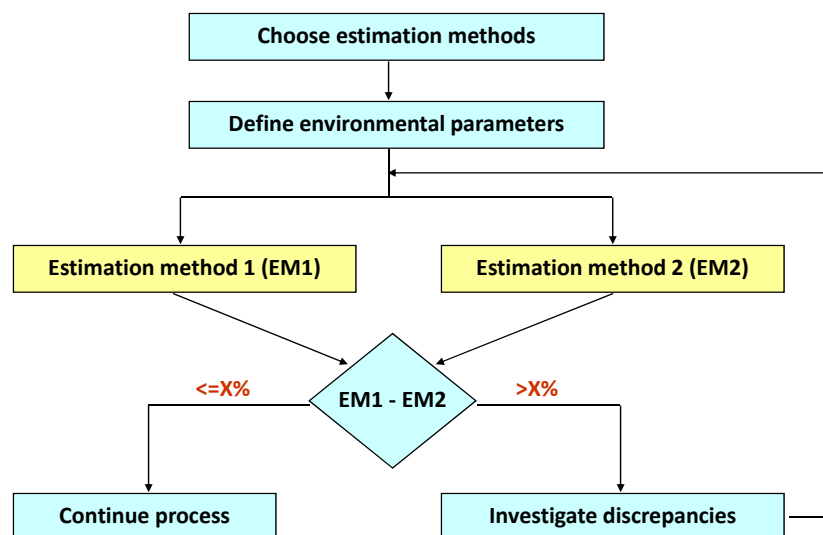
Estimates and Risk

Estimates are used to plan and manage risks.

- Business risks are associated with time to market, price leverage and market share.
- Product risks are associated with product quality, features and fitness for use.
- Project risks are team technical capability, resource availability, and change management.

Now, we'll start to talk about incorporating the risk concern.

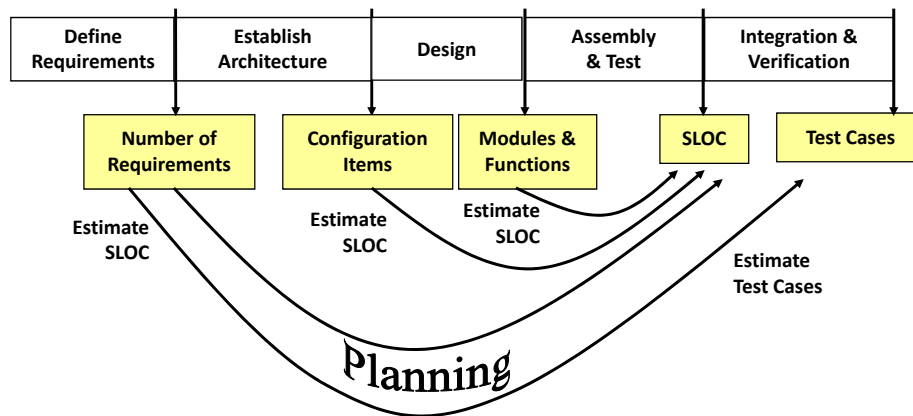
Use Multiple Estimation Methods



Continuous Size Estimation

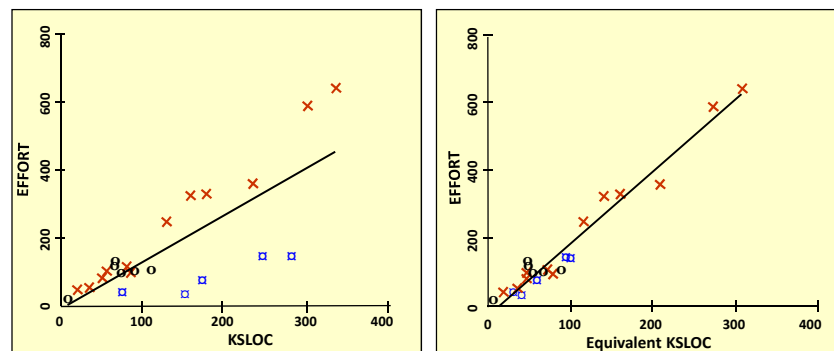
- Measure what is available at the end of a phase to predict the end-product size
- Compare changing size estimates

Generic Lifecycle Phases

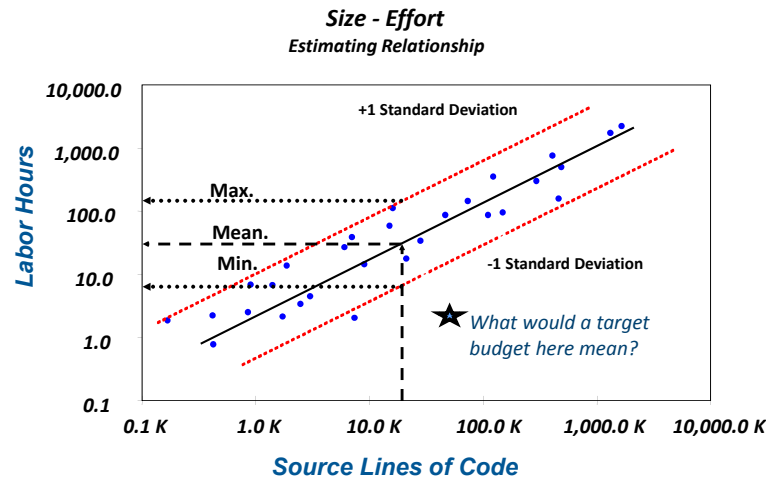


Accounting for Reused Code

- The left chart shows the relationship between **total KSLOC** and effort
- The right chart shows the relationship between total **Equivalent KSLOC** (count adjusted for reuse) and effort



Effort Estimation Using Size-Effort Relationship



[Putnam 92]

Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

49

Estimating Schedule

Determine the length of time needed to complete the project and determine when major milestones will occur

Schedule depends on effort: *How much effort is available to do the work?*

There will be a nominal schedule for the estimated cost, and a curve that provides a tradeoff between schedule and cost. There is also *a minimum schedule* that cannot be beat.

Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

50

Schedule Estimation

For **small tasks** with similar effort consumption:

- Depends heavily on people and their availability
 - 1 Person Day = 1 Calendar Day
- Task interdependencies can cause delay

For **larger projects** with variable effort consumption:

- Parametric models can account for compressed / extended schedules

$$\text{Months} = C \times \text{Effort}^{0.28}$$

$C = 3.67$

Effort in Staff Months

Ref: COCOMO II, 2000

Quality Considerations

Several tools calculate the expected number of defects

Removed by quality activities.

- includes testing, inspection, etc.

Remaining in system at time of release.

Quality factors can heavily influence the size and schedule.

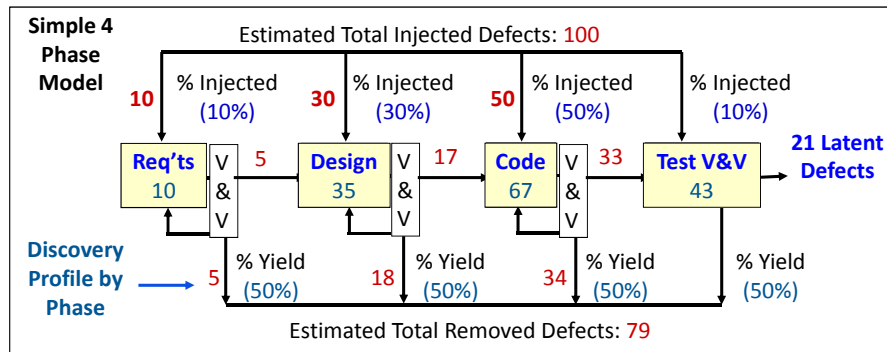
- Safety critical systems require more documentation and testing.
(medical devices)
- Defect removal is a surrogate for progress.

What methods can we use to estimate or predict defects?

Empirical Defect Prediction Technique

Description - # of Defects per Size (Defect Density)

- Historical defect density (# of Defects / KLOC)
- Defect classification history can show enhanced with historical data on injection distribution and yield



Empirical Defect Prediction

When to use

Use for planning and estimating (total defects) and in-process monitoring of defect discovery numbers (latent defects)

Required Data

Historical defect density data required for planning; in-process data required for monitoring

Strengths

Easy to use and understand; can be implemented with minimal data

Weaknesses

Requires stable processes and standardized life cycle; does not account for changes in the project, personnel, platform, or project

Module Outline



Some planning objectives and questions

The estimation process

Some estimation methods

Planning from the estimate

How do you know the estimate is OK?

The Estimate is Not the Plan

The estimate describes

- How large is this project by comparison to others?
- What will it cost in terms of how much work is required?
- How long it will take to complete the work?
- What is the tradeoff between cost and duration?
- What is a reasonable range for the estimated values?
- What assumptions had the greatest impact?



Mapping the Estimate to the Plan

The WBS was chosen as part of the estimating process.

3-step allocation method

1. Allocate resources to deliverables (supports earned value).
[Allocate defect information to deliverables]
2. Allocate duration to milestones.
3. Map people-skills to tasks.

You may need to iterate through the steps in order to build a schedule.

Create baseline plan and charts that map estimated values to schedule for reporting.

COCOMO II Activities vs. Phases

Activities

Requirements	45.0%	12.5%	4.0%	2.5%
Product design	17.5%	41.0%	8.0%	5.0%
Programming	5.5%	13.5%	56.5%	39.0 %
Test planning	4.0%	6.0%	5.5%	3.0%
V&V	7.5%	7.5%	8.5%	28.5%
Management	12.0%	10.0%	6.0%	7.0%
CM/QA	3.0%	2.5%	6.5%	8.0%
Manuals	5.0%	7.0%	5.0%	7.0%
	Req'ts (7%)	Product Design (17%)	Coding & Unit Test (55%)	Integration & Test (28%)

Phase

[COCOMO 00]

Model % Schedule Distribution

Size	Product Design	Programming	Integration and Test
2 KSLOC	24%	56%	20%
8 KSLOC	25%	52%	23%
32 KSLOC	26%	48%	26%
128 KSLOC	27%	44%	29%
512 KSLOC	28%	40%	32%

As with effort distribution, an organization should collect measurements for creation of their schedule distribution.

[COCOMO 00]



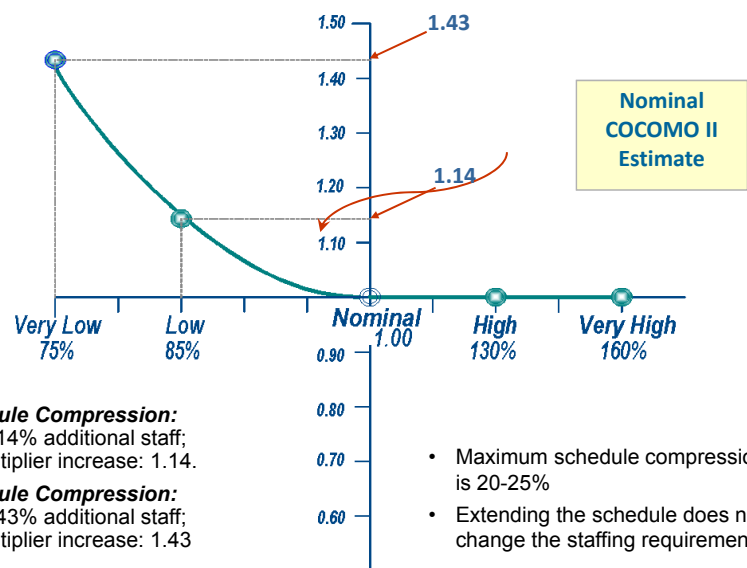
Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

59

Effects of Schedule Compression



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

60



Other Schedule Considerations

Look at project phases

- Is the required staff available to begin work at start of project?
- Is there enough time available?
- Are there any overlaps and dependencies among WBS elements and releases? Are they reasonable? What assumptions and risks do they entail?

Has time for training been allocated?

Other Cost Considerations

What other resources are required?

- computers
- office space, office supplies
- laboratories and test facilities
- hardware and software tools
- maintenance of equipment
- project support
- liaison with vendors, sponsors, external teams, financial management
- travel budget
- software licenses

Module Outline



Some planning objectives and questions

The estimation process

Some estimation methods

Planning from the estimate

How do you know the estimate is OK?

Requisites for Reliable Estimating Processes

1. A corporate memory (database, repository)
2. Structured processes for estimating size and reuse
3. Mechanisms for extracting history from projects
4. Audit trails (values used to estimate are recorded)
5. Integrity in dealing with cost and schedule constraints
6. Data collection and feedback processes that foster capturing and interpreting data from work performed

Park, Robert E., "Checklists and Criteria for Evaluating the Cost and Schedule Estimating Capabilities of Software Organizations", SEI-95-SR-005

Indicators of Estimating Capability

1. Management acknowledges its responsibility for developing and sustaining an estimating capability.
2. The estimating function is supported by budget.
3. Estimators are equipped with tools and training needed for reliable estimating.
4. People assigned as estimators are experienced and capable.
5. Recognition and career paths exist such that qualified people want to serve as estimators.
6. Process improvement resources and funds are committed to improving the estimating process.
7. The estimating capability of the organization is tracked and evaluated.



Manager's Estimation Checklist

7 questions to ask when assessing your willingness to rely on a cost and schedule estimate.

- Are the objectives of the estimate clear and correct?
- Has the task been appropriately sized?
- Are the estimated cost and schedule consistent with past project history?
- Have the factors that affect the estimate been identified and explained?
- What is the integrity of the estimating process?
- Is the organization's historical evidence capable of supporting a reliable estimate?
- Has the situation changed since the estimate was prepared?



Checklist for Issue 1

Issue 1. Are the objectives of the estimate clear and correct?

Evidence of Credibility

- The objectives of the estimate are stated in writing. ☐
- The life cycle to which the estimate applies is clearly defined. ☐
- The tasks and activities included in (and excluded from) the estimate are clearly identified. ☐
- The tasks and activities included in the estimate are consistent with the objectives of the estimate. ☐

[Park 95]



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

67

Part of Checklist for Issue 2

Issue 2. Has the task been appropriately sized?

Evidence of Credibility

- A structured process has been used to estimate and describe the size of the software product.. ☐
- A structured process has been used to estimate and describe the extent of reuse. ☐
- The processes for estimating size and reuse are documented. ☐
- The descriptions of size and reuse identify what is included in (and excluded from) the size and reuse measures used. ☐

[Park 95]



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

68



Feasibility Analysis

Do this as part of initial planning activity and at all subsequent re-plans.

Looks at:

- basis for estimate
- realism of adjustments
- confidence in process
- changes in assumptions or environment
- comparisons of project parameters

Conduct analysis to determine whether plans and targets are realistic and achievable.

REF: PSM 00



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been
approved for public release and unlimited distribution.

69

Plan Feasibility Checklist – 1

Single Plan Issues

- Are important activities missing on schedules?
- Is the overlap between project activities reasonable?
- Are holidays, vacations, etc., considered in plans/schedules?
- Is the rate of planned progress, shown by the slope, reasonable?
- Is planned performance consistent with past performance?

REF: PSM 00



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been
approved for public release and unlimited distribution.

70



Plan Feasibility Checklist -2

Plan Integration Issues

- Do totals, such as total number of units and total effort, match?
- Do summary figures match more detailed breakdowns?
- Do dates, milestones, and timeframes match?
- Does the effort allocated match workload?
- Does the effort allocated match cost/budget plans?
- Does any other project information contradict plans?

REF: PSM 00



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been
approved for public release and unlimited distribution.

71

Principles of Estimating

Estimates are made by people, not by models.

- They require reasoned judgments and commitments to organizational goals that cannot be delegated to any automated process.

All estimates are based on comparisons.

- When people estimate, they evaluate how something is like, and how it is unlike, things that they or others have seen before.

Before people can estimate, they must acquire knowledge.

- They must collect and quantify information from other projects, so that they can place their comparative evaluations on demonstrably sound footings.

[Park 95]



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been
approved for public release and unlimited distribution.

72



Summary – Content

A good estimate supports planning and decision making for the rest of the work in the project.

The “thought-space” of the estimate is the relationship between several controllable factors.

- Scope
- Resources and skills
- Schedule
- Quality
- Risk

The “estimate” is not negotiated. Negotiating one (or several factors) affects the assumptions underlying the estimate. Increased risk may be the consequence.

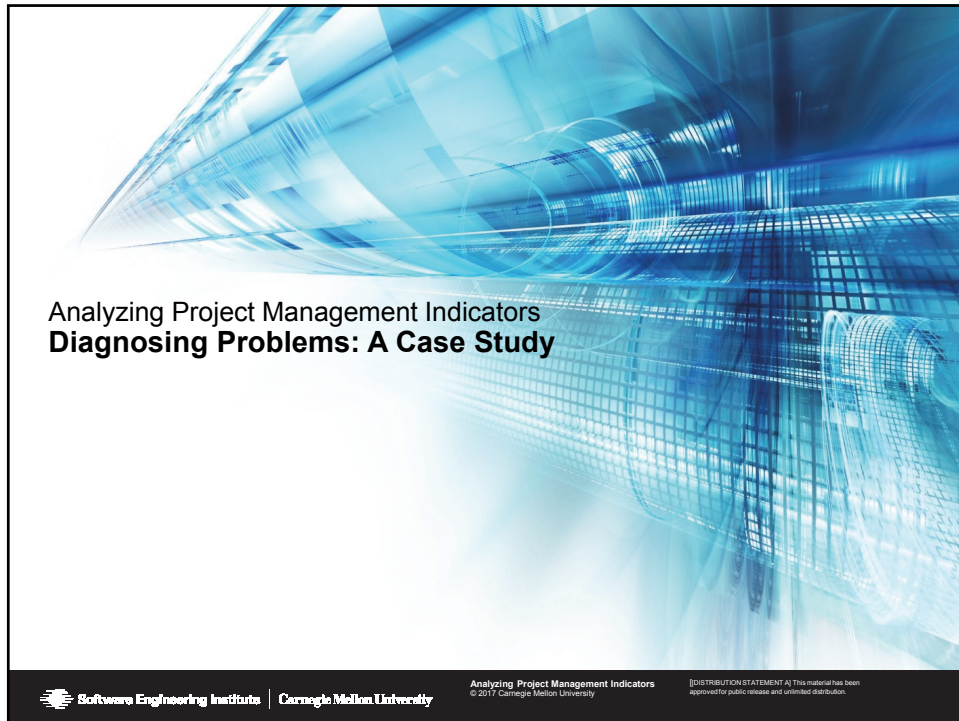
Summary – Process

The person(s) doing the estimate must have experience.

Every estimate requires the use of historical data.

The manager should expect to check the estimate and the process used to generate it.

Save the estimate and the basis of estimate and compare it to project results.



Module Objectives



- Analyze a problem project.
- Decide what actions you would take.
- Discuss.

Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

2

Overview of Decision-Making

Complex decisions are common in project management.

These decisions depend on:

- The current context of the problem or issue
- Some alternative choices (including no action)
- Attributes that have a range of values

Deciding means:

- Understanding the desired impact
- Identifying the attributes that contribute significantly
- A model for selecting among the choices

Decisions and Controls

What are project control levers?

What portion of that control is in your scope of action?

- schedule
- resources
- external expenses
- requirements and quality goals
- process & tools
- technology

Outline for Problem Solving

Schedule slip example

Decision making steps

- What is happening?
- What might be causing the problem?
- What control should you try?
- What are the possible consequences of the control action?

Schedule Slip Example

The project is undergoing a management review

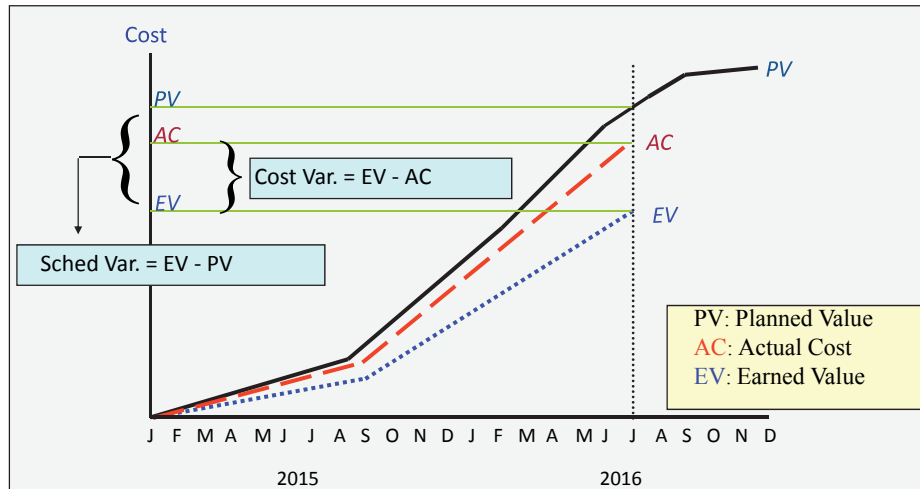
- both cost and schedule have slippage

Our goal:

- Find out **what** is happening.
- Find out **why** it is happening.
 - What questions would you ask?
 - What additional information do you need?
- What **decisions/actions** would you take based upon the data presented?
- What are the **future consequences**?

Earned Value

Both cost and schedule have slippage



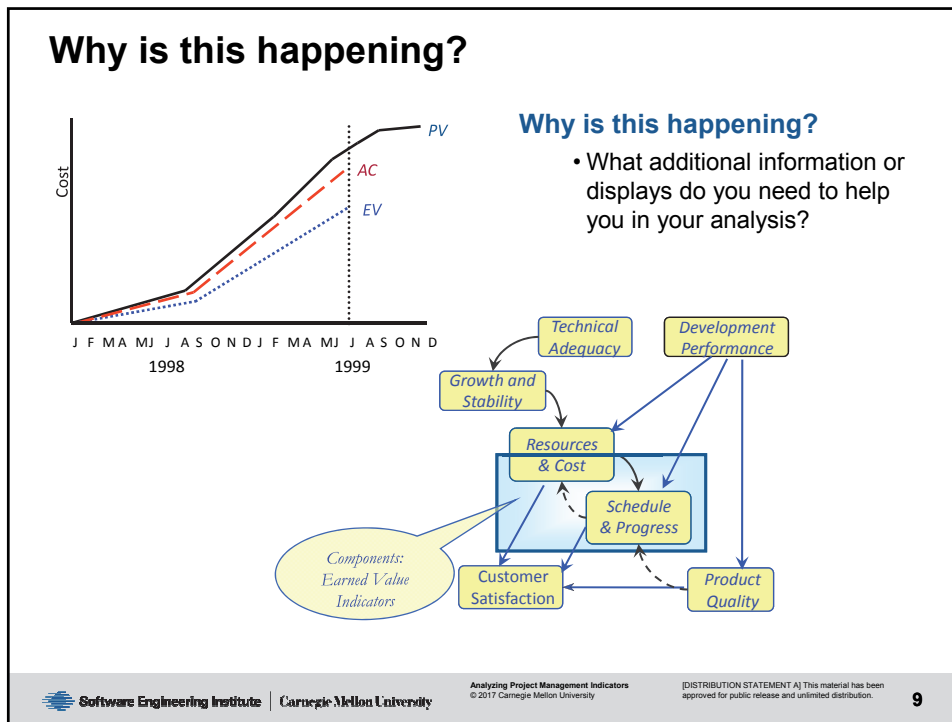
What Is Happening?

Observations:

- both cost and schedule have slippage
- cost slip is: $EV - AC$
- schedule slip is: $EV - PV$

Areas for further investigation:

- why is this happening?
- what additional information or displays do you need to help you in your analysis?



Possible Answers

Observations: project has both cost and schedule have slippage

Performance

- Development performance does not meet estimate.
- People do not have training or background
- External milestones

Resources

- People doing work against wrong account

Technology

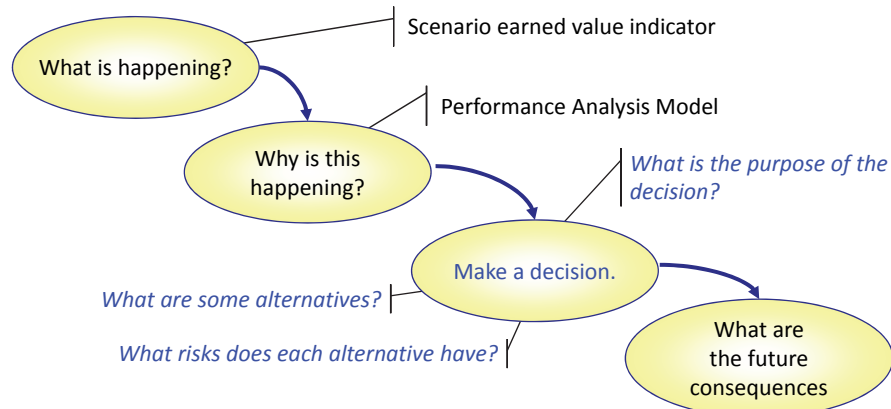
- Technology adoption is slow
- Technology doesn't fit

Size

- Requirements growth or change
- Application larger than estimate

Successful Metrics Activity

Measurement by itself does not do anything – ***action must be taken.***



Ref: Kepner 97

Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

11

Analysis

For each indicator, ask the following questions:

- What additional information does this indicator tell you?
- What questions would you ask about this indicator?
- What additional information or displays do you need to help you in your analysis?
- Does this chart negate any of your postulated reasons for the slippage?

Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

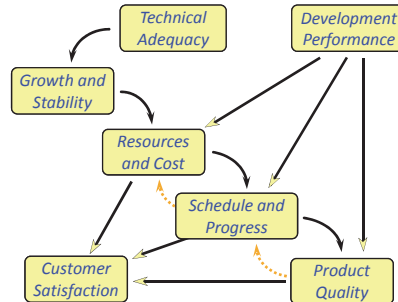
12

Finding Causes – Performance Analysis

Try this model to diagnose the problem and look for cause.

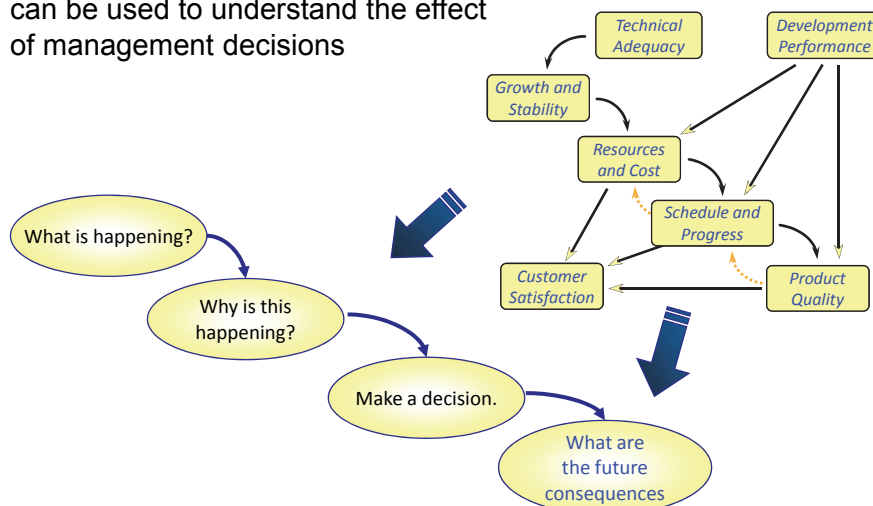
Schedule slip is the starting point.

Get additional data based on searching upstream.



Future Consequences

The [Performance Analysis Model](#) can be used to understand the effect of management decisions



Make Decisions – 1

Try “what-if” scenarios

- Determine where tradeoffs can be made to bring the project into line with project objectives.
- Example: Projected excessive cost may be mitigated by taking action in an upstream area.
 - Amount of work may be reduced by reducing requirements.
 - Productivity may be enhanced by increasing the level of automation.

Make Decisions – 2

Do not make conclusions on a single item of evidence.

Use historical and qualitative experience from similar projects.

Alternatives should be considered in the context of risk and financial status.

- Purchasing a design tool may improve future productivity but at the expense of acquiring, installing, and training.

Availability of budget and schedule may affect the project's ability to implement the proposed course of action.

Make Decisions – 3

Extrapolate actual performance out to the end of the phase.

- estimate and predict the future based on current performance
- use parametric models

Use project variance-to-date to predict future performance.

Use predicted outcome to predict the effect on performance in areas downstream in the Performance Analysis Model.

Make Decisions – 4

The measurement results must be clearly understood by the decision makers.

Make results available throughout the organization.

Action must be taken to realize any benefit from measurement.

Desired actions may not be possible - may have to optimize within project constraints.

Report Results

Use measurement analysis results to communicate facts and the need for a change.

- overall evaluation of project
- identification of specific problems, risks, and lack of information
- recommendations
- potential new issues or questions

Why is this happening? – Discussion

It is your turn to ask questions that will lead to the discovery of the cause of the problem.

In response to your questions, we will present indicators with project information.

Exercise

The project is undergoing a management review

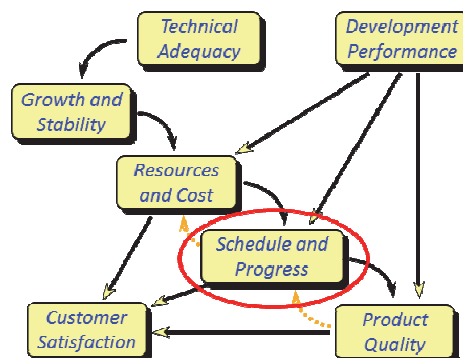
Our goal:

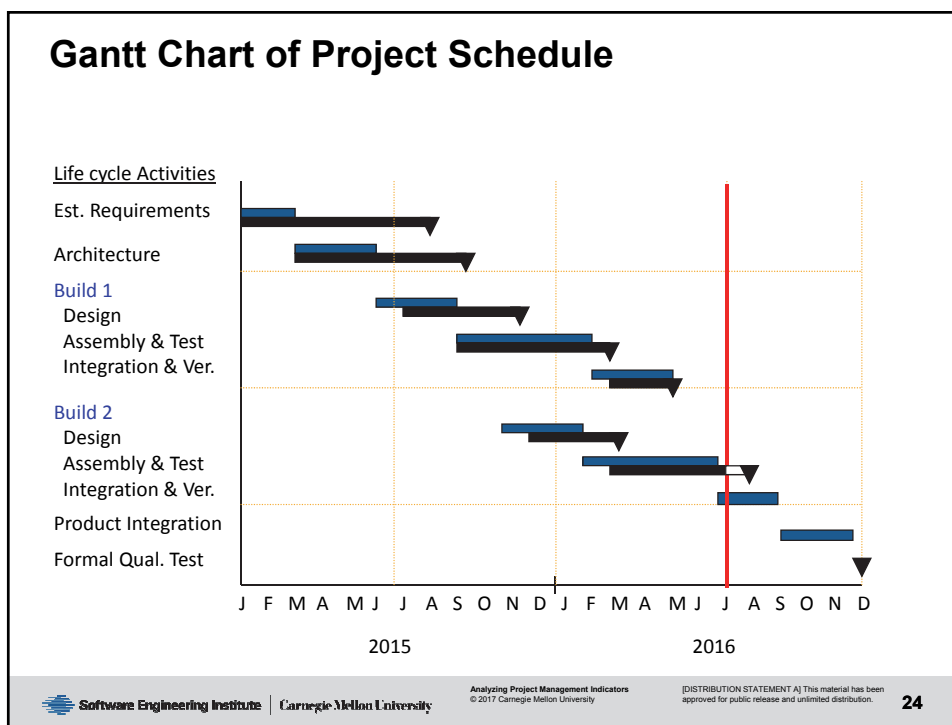
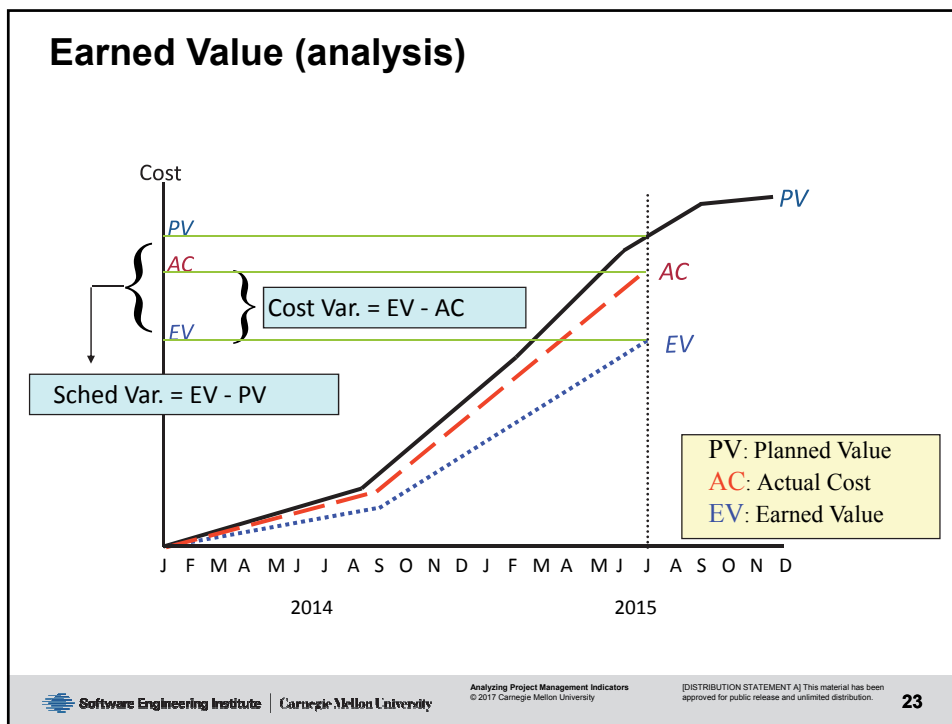
- Find out what is happening.
- Find out why it is happening.
 - What questions would you ask?
 - What additional information do you need?
- What decisions/actions would you take based upon the data presented?
- What are the future consequences?

Schedule and Progress Indicators

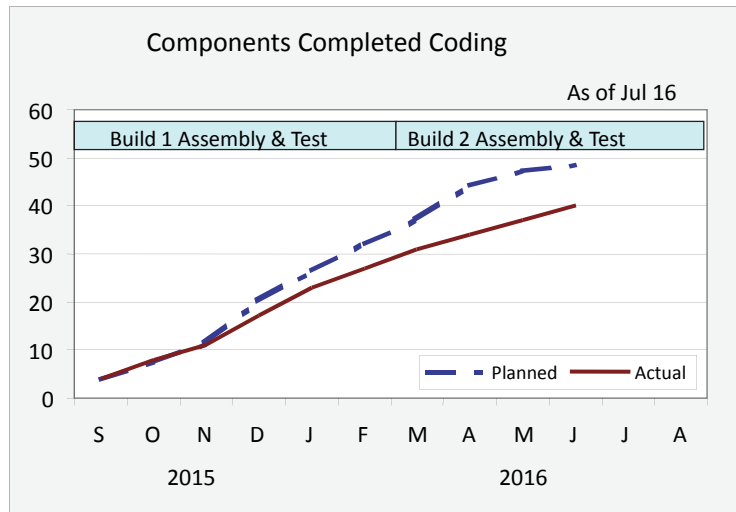
Problem: Both Cost and Schedule have slippage

WHY?





Work Unit Progress



What We Know So Far

Schedule and progress indicators

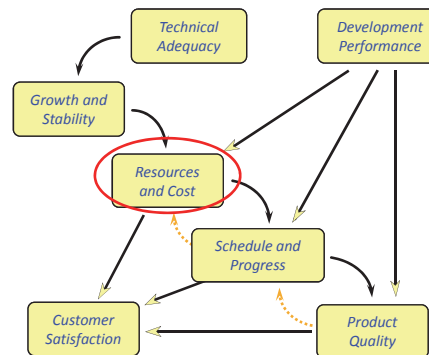
- Cost and schedule slippage (EV chart)
- Activities taking longer than planned (Gantt chart)
- Assembly and test behind schedule (Components completion chart)

Yes, we have a problem.

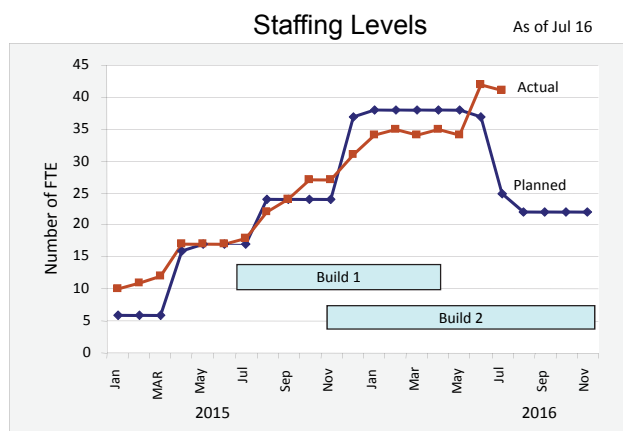
Resources and Cost Indicators

Problem: Both Cost and Schedule have slippage

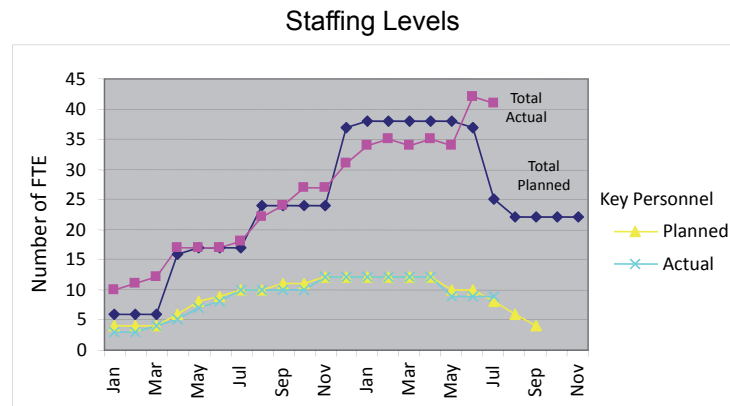
WHY?



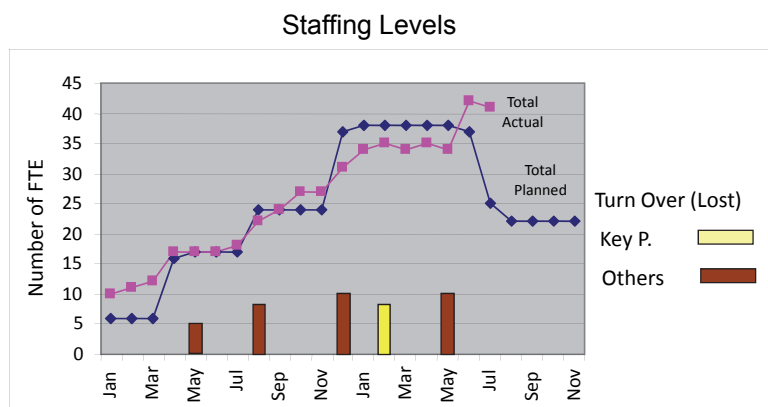
Resource and Cost



Key Personnel Availability



Personnel Turnover



Analyzing Project Management Indicators

Staff Allocation

		Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun	Jul
Mgt	Plan	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	Actual	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2	2	2	2	2
Analyst	Plan	3	3	3	7	5	5	4	3	3	2	1	1	1	1	1	1	1	1	1
	Actual	2	2	2	4	4	4	3	3	2	2	2	2	2	2	2	2	2	1	1
Design	Plan	1	1	1	5	6	6	6	6	6	5	5	5	5	5	3	3	3	3	1
	Actual		1	1	3	3	3	3	3	4	4	4	4	4	4	3	3	3	3	1
Prg	Plan				2	3	3	3	9	9	11	12	22	23	23	22	22	22	22	15
	Actual	3	3	4	5	5	5	6	9	9	11	11	15	18	19	19	20	20	30	30
Tester	Plan					1	1	1	3	3	2	2	5	5	7	8	8	8	8	5
	Actual	3	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5
Other	Plan	1	1	1	1	1	1	2	2	2	3	3	3	3	3	3	3	3	3	3
	Actual	1	1	1	1	1	1	1	2	2	3	3	3	3	3	3	3	3	3	3
Total	Plan	6	6	6	16	17	17	17	24	24	24	24	37	38	38	38	38	37	25	
	Actual	10	11	12	17	17	17	18	22	24	27	27	31	34	35	34	35	34	42	41
DELTA		4	5	6	1	0	0	1	-2	0	3	3	-6	-4	-3	-4	-3	-4	5	16



Additional Information

Resources and Cost

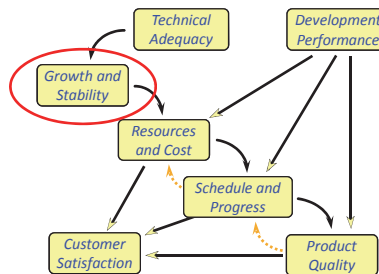
- Staffing did not follow planned level
 - Too many at beginning of project
 - Below planned level during most development
 - High turnover rate
- Cost over run due partly to staffing problems and retraining costs



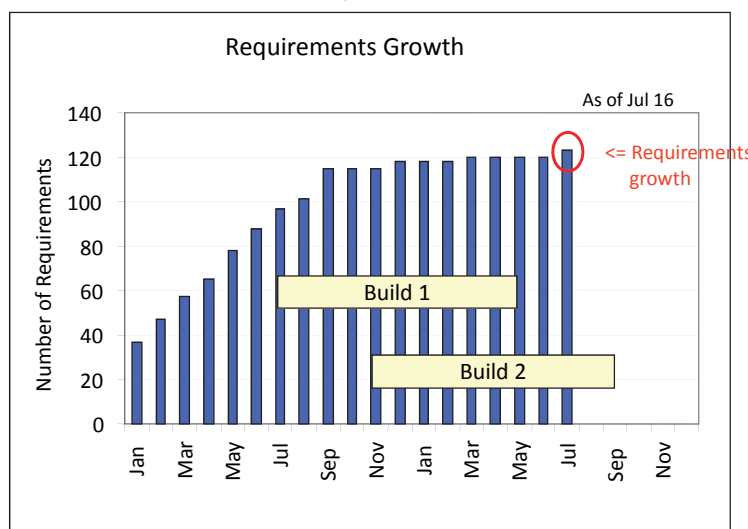
Growth and Stability Indicators

Problem: Both Cost and Schedule have slippage

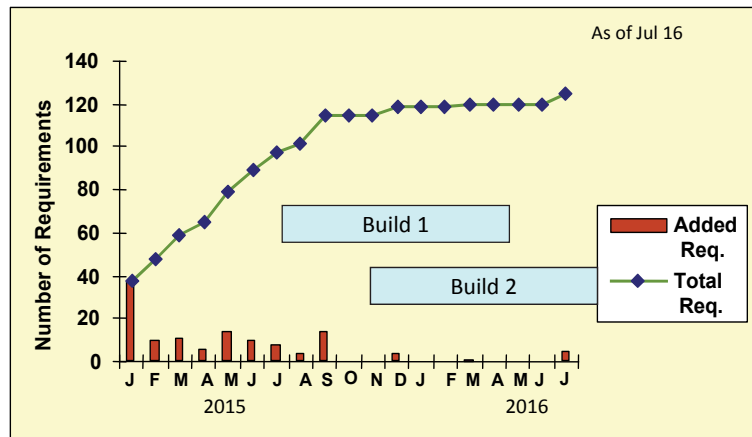
WHY?



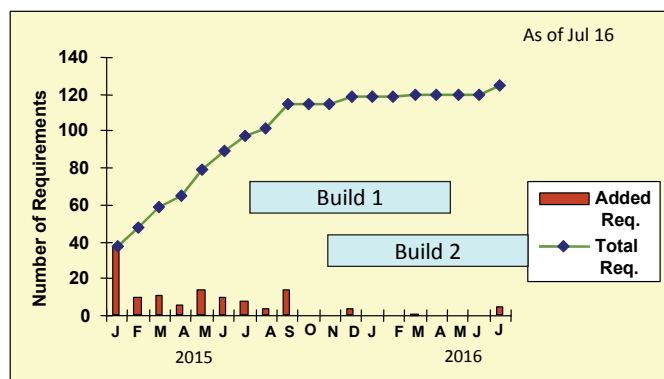
Growth and Stability



Requirement Changes

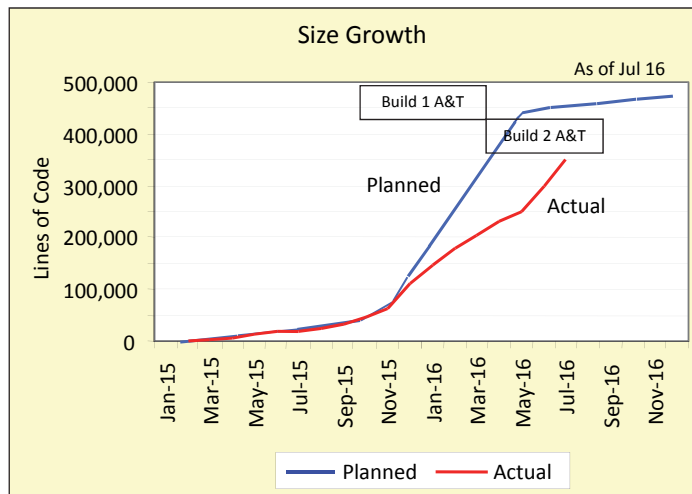


Requirement Changes



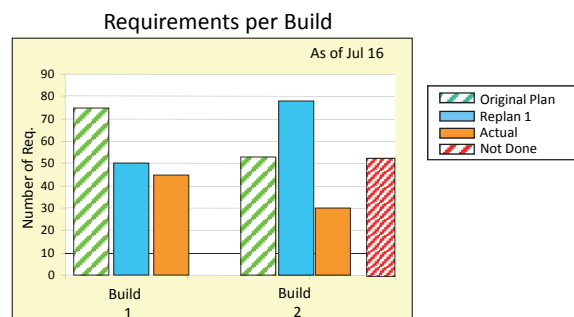
	2015								2016		
	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Dec	Mar	Jul
Req Changes	10	11	6	14	10	8	4	14	4	1	5
Complexity	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low
Resources (staff-days)	4	5	3	2	4	2	3	3	2	1	2

Code Growth



A&T: Assembly and Test

Build Contents



Observation: Functions deferred to later build because of unanticipated growth in complexity.

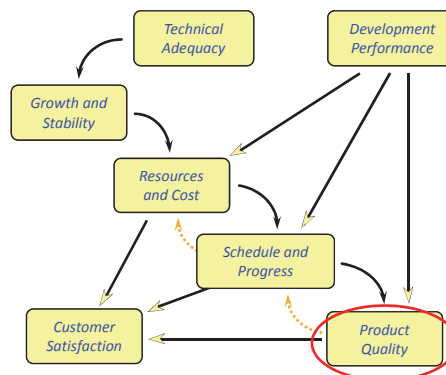
Additional Information

Growth and Stability

- Requirement changes are of low complexity and small
- Requirements changes after Build2 code and unit test
- Code production below planned value
- Functionality being deferred from build 1 to build 2 due to unanticipated growth in complexity
- Since code production below planned value, implementing all functions now in build2 may be unrealistic

Product Quality Indicators

Problem: Have cost and schedule problems caused other problems?



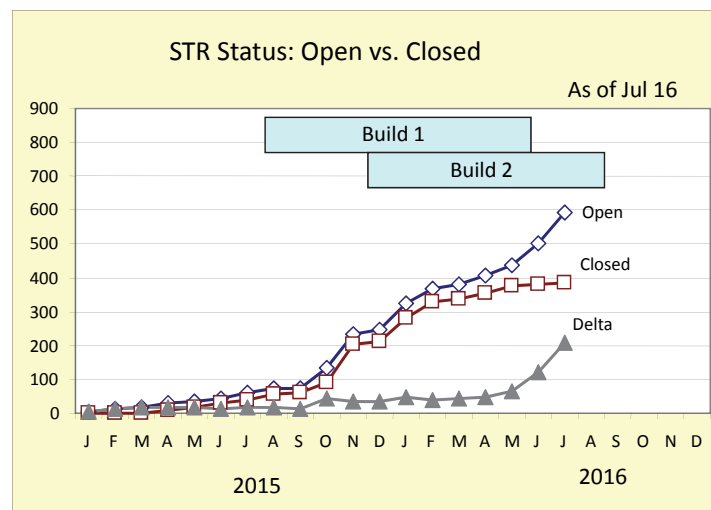
Product quality problems?

What are product quality problems?

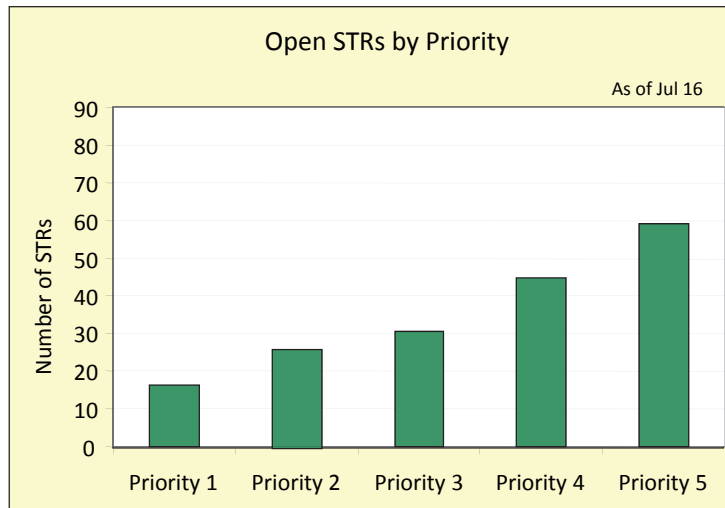
What do they mean to you?

What charts or information do you want?

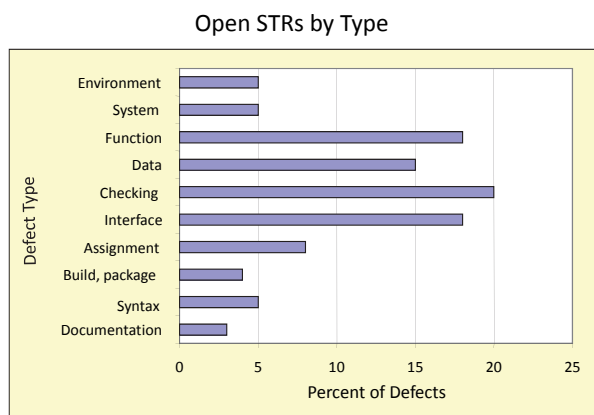
STR Status



Quality



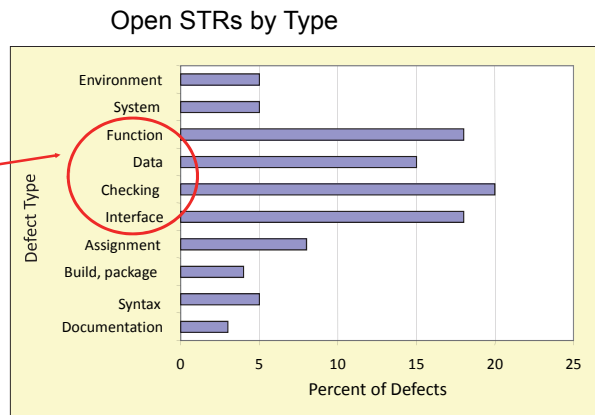
Type of Errors– 2



Type of Errors – 2

Companies Past Experience

Types predominately found during Code Inspections



Software Engineering Institute

Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

45

Additional Information

Product Quality

- STRs are being opened faster than being closed
- Code inspections could have found type of errors identified



Software Engineering Institute

Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

46

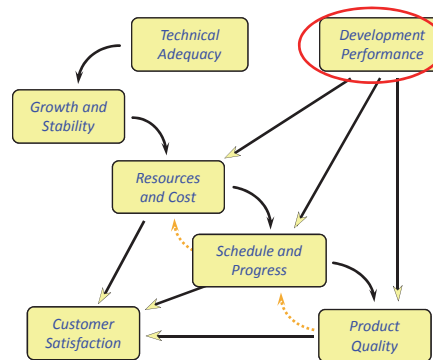


Development Performance Indicators

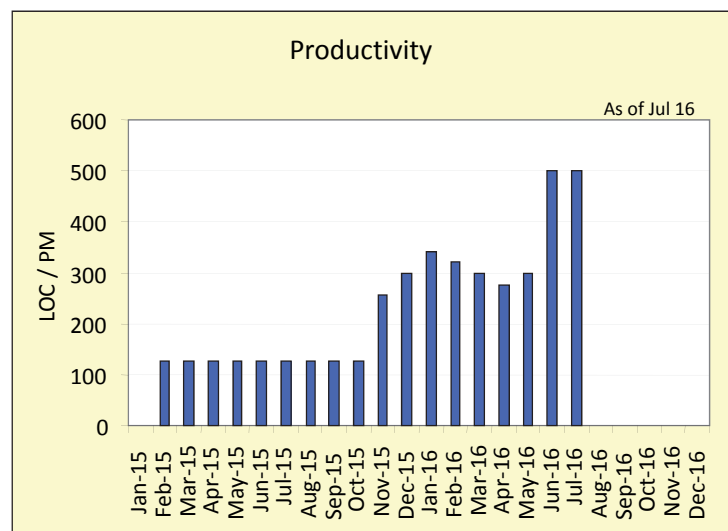
Problem: We see product quality problems

WHY?

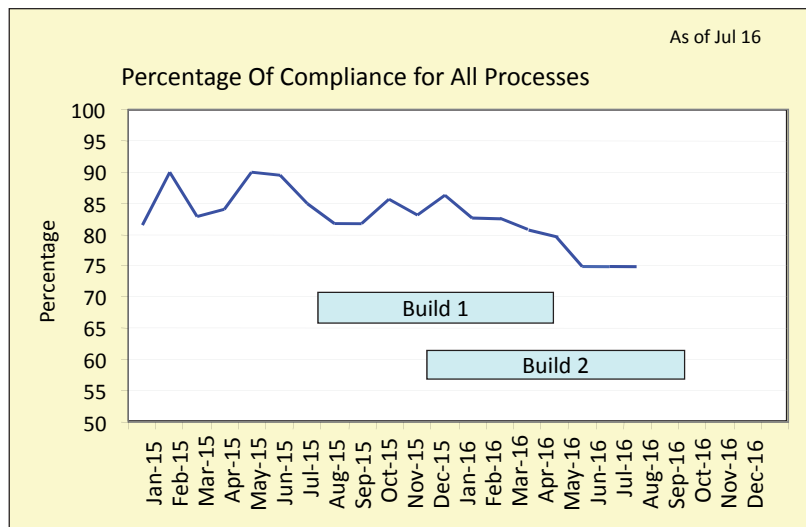
What information do you have available to explore development performance?



Performance



Process Maturity



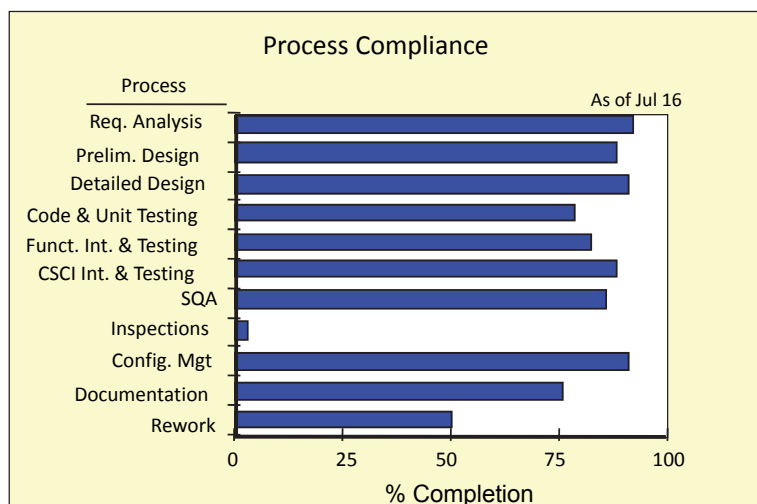
Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

49

Process Compliance



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

50



Additional Information

Development Performance

- Stopped doing inspections

Summary of Information – 1

Schedule and Progress Indicators

- Confirm we have a problem

Growth and Stability

- Requirement changes are low complexity and small
- Code production below planned value
- Functionality being deferred from build 1 to build 2 due to unanticipated growth in complexity
- Since code production below planned value, implementing all functions now in build2 may be unrealistic

Summary of Information – 2

Resources and Cost

- Staffing did not follow planned level
 - Too many at beginning of project
 - Below planned level during most development
 - High turnover rate
- Cost over run due partly to staffing problems

Product Quality

- STRs are being opened faster than being closed
- Code inspections could have found type of errors identified

Summary of Information – 3

Development Performance

- Stopped doing inspections

Reasons for Schedule Slippage

Staffing did not follow planned level

- too many at beginning of project
- below planned level during most of development
- high turnover rate

Process compliance: stopped doing inspections

Requirements changes after Build 2 code and unit test

What Actions Would You Take?

-
-
-
-
-
-
-
-
-

Possible Actions

Replan based on current performance

Use contract labor

Deliver smaller size - less functionality

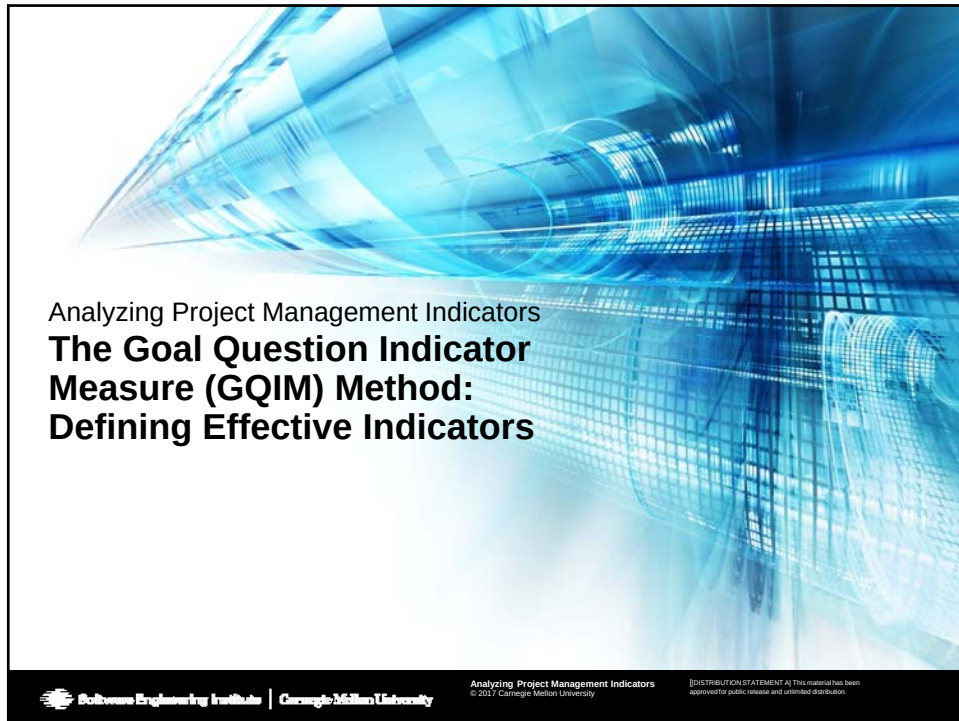
Restart inspections

- code
- test cases

Do not decrease staffing to conform to “planned” staffing

What Actions Could Have Been Taken Earlier to Prevent Slip?

-
-
-
-
-
-
-



Problem

Does your organization collect a multitude of measures that have little impact or value?

Are you wondering how to provide the information that management, engineering, and operations want and need?

Solution

Design and develop measures that meet identified information needs.

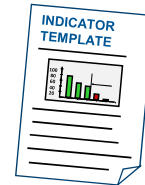
Don't just measure what's easy to measure.

Use the SEI Indicator Template

Defining Effective Measures

The template is

- the cornerstone of the Goal-Question-Indicator-Measure method
- used to align collected measures with the goals and information needs of management and engineers
- designed to map to the specific practices in the CMMI Measurement and Analysis process area and serves as a practical tool for implementing and improving your measurement processes



We will learn how the SEI indicator template:

- assists with implementing measurement and analysis processes
- serves as an effective tool for facilitating the development and definition of effective measures, analyses, and indicators
- provides measurement, analysis, and reporting specifications critical for measurement and analysis improvement and planning



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been
approved for public release and unlimited distribution.

3

Module Outline

Background



Description of indicator template

- initial
- example of use

Chart examples

Measurement handbook



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been
approved for public release and unlimited distribution.

4



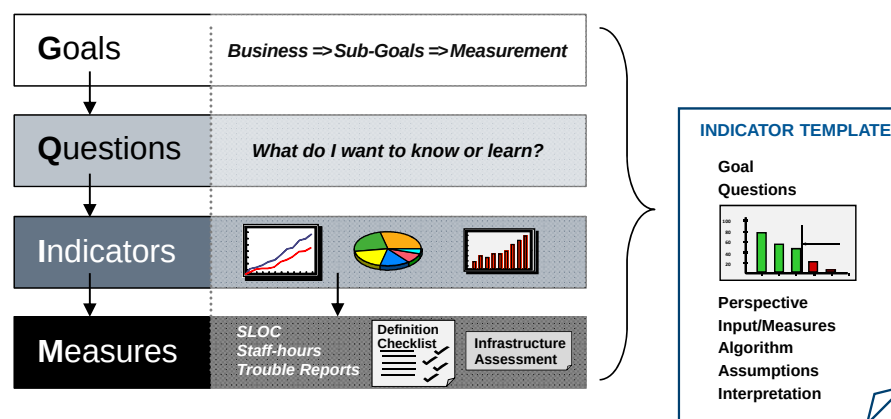
Background

Many of the potential benefits that an organization can derive from a sound measurement program are often not achieved due to inconsistent construction and interpretation of indicators derived from measurement data.

The indicator template is a

- tool an organization can use to direct its data collection and measurement and analysis processes
- comprehensive template that provides guidance for the development and precise description of an indicator

Goal-Driven Measurement Workshop



Module Outline

Background

Description of indicator template

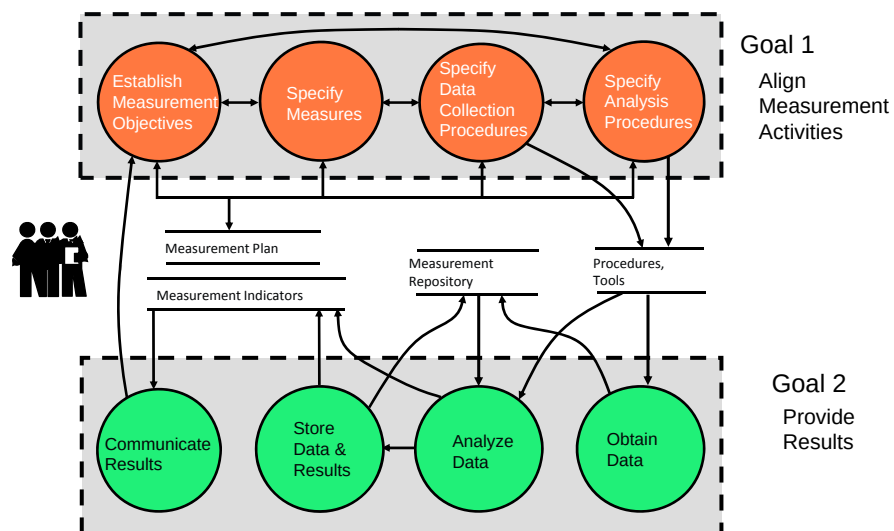
- initial
- example of use

Chart examples

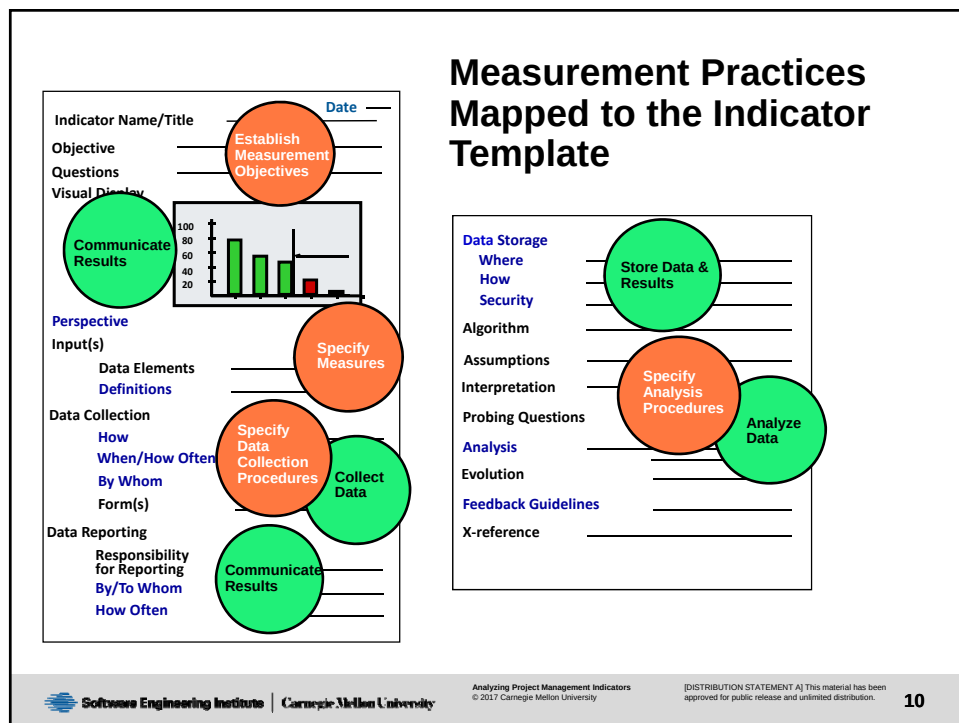
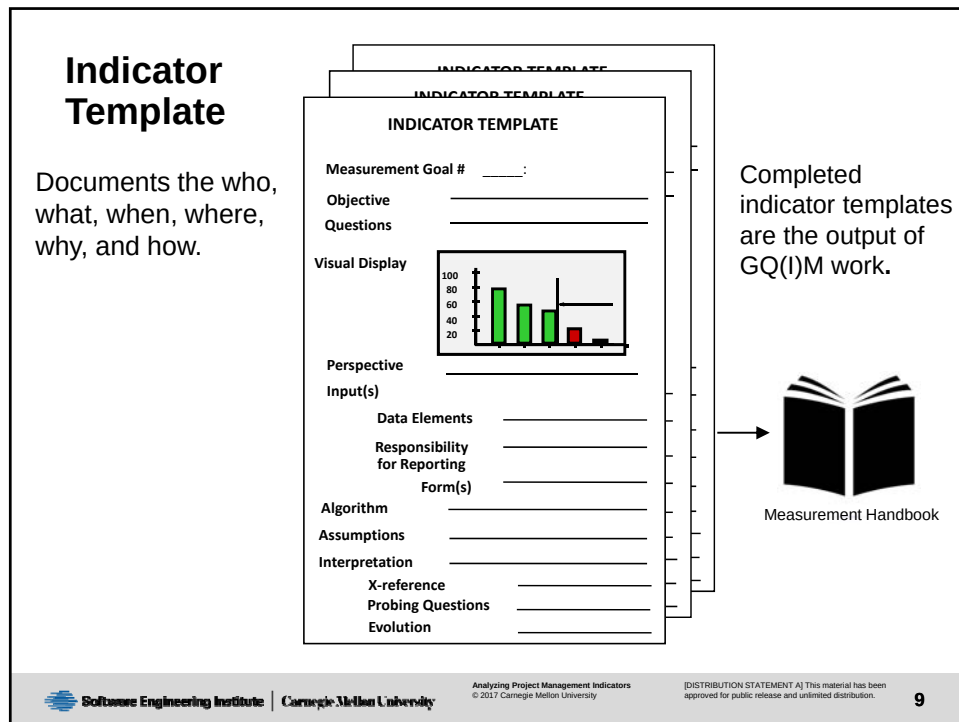
Measurement handbook



CMMI Measurement Activities



Analyzing Project Management Indicators



Contents of Indicator Template – 1

Indicator objective	Objective or purpose of the indicator
Questions	List of questions that the indicator is trying to answer
Perspective or viewpoint	Description of the audience for whom this indicator is intended
Visual display	Graphical view of the indicator
Data Elements	List of data elements (measures) required to construct the display/indicator
Data Collection Information	Information pertaining to how, when, how often, and by whom for the data elements required to construct the indicator. Also identify any specific forms or tools to be used



Contents of Indicator Template – 2

Data Reporting Information	Information about who is responsible for reporting the data, to whom, and how often
Data Storage	Information on the storage, management, access, and security of the data and reports
Algorithm	Description of the algorithm used to construct the display/indicator
Assumptions	List of assumptions about the organization, its processes, life-cycle model, and so on that are important conditions for understanding this indicator
Analysis and Interpretation of Results	Guidance on the analysis and interpretation of the indicator. Often useful to note common misinterpretations or misuses. Related fields include “probing questions” and “cross-references.”
Evolution and Feedback	A place to capture ideas for improving the indicator over time, especially as more data become available and processes mature





How to Complete the Indicator Template – 1

Indicator objective	Strategic plan, project goals, stakeholder workshop, interview intended users
Questions	Stakeholder workshop or interviews. Ask what specific questions are important.
Perspective or viewpoint	Elicit as part of identifying the objective
Visual display	Have stakeholders sketch what they want
Data elements	Derive from sketched display and confirm with those producing the data
Data collection information	Look at existing process descriptions for opportunities to instrument the process. Find out from those generating and providing the data how it is really done, what tools and forms are used, what triggers the collection of the data, and so forth. Don't rely solely on documentation.



How to Complete the Indicator Template – 2

Data Reporting Information	Collect requirements about format, timing, and content from the intended recipients. Provide samples and collect feedback. Allow for a change request procedure.
Data Storage	Work with IT staff to establish the infrastructure including storage, configuration management, access management, and archiving of data and reports. Also, be clear about what data are stored in raw format and what will be summarized and stored.
Algorithm	Confirm this with stakeholders. Does it align with the questions and objective? Are there any hidden assumptions?





How to Complete the Indicator Template – 3

Assumptions

Review and confirm with stakeholders and subject matter experts. Be sure to get understanding about the details associated with data collection. Are there important gaps in the data collection?

Analysis and Interpretation of Results

Draw different patterns of results on the indicator and describe what they mean. For line charts, what does it mean if lines are parallel, converging, or diverging? Often useful to note common misinterpretations or misuses. Related fields include “probing questions” and “cross-references.” Consider the PSM Analysis Model as a tool.

Evolution and Feedback

Capture ideas for improving the indicator over time, especially as more data become available and processes mature. Allow for a change request process. Note comments and feedback.



Module Outline

Background

Description of indicator template

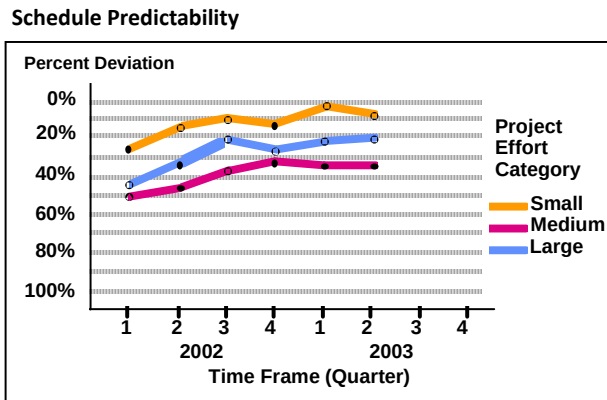
- initial
- **example of use**

Chart examples

Measurement handbook



Example: Schedule Predictability



What do you need to know to interpret this indicator?



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

17

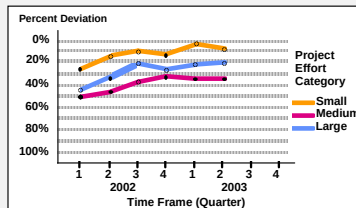
Schedule Predictability – 1

Indicator Name: Schedule Predictability

Objective: To monitor trends in the predictability of meeting schedules as input toward improvements at the technical unit level and across the enterprise.

- Questions:**
- Are we improving our schedule estimates in small, medium, and large projects?
 - How far are our schedule plans from actual effort, cost, & dates?

Visual Display:



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

18



Schedule Predictability – 2

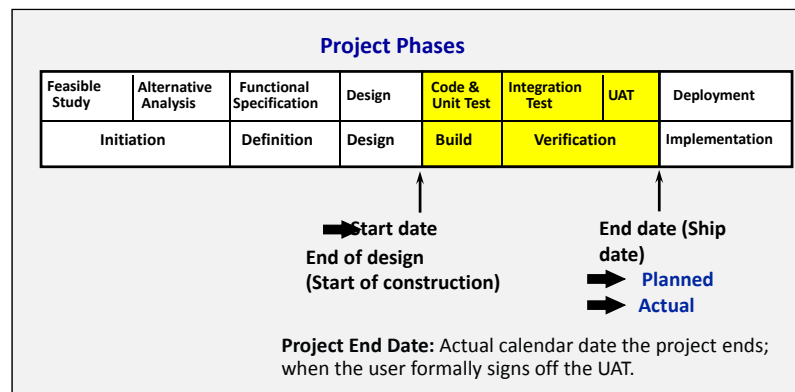
Input: Data is to be segregated into three project effort categories (small, medium, and large) and only submitted for projects completed during the quarter.

Data Elements:

There are two types of input data:

1. **Organizational reference information**, which includes
 - name of organization
 - reporting period
 - contact person
 - contact phone number
2. **Schedule predictability metric data** for each project completed during the period, which includes
 - actual date of the end of the design phase
 - planned ship date
 - project end date
 - effort category (small, medium, or large)

Schedule Predictability – 3



Graphic included to ensure no misunderstanding.

Schedule Predictability – 4

Responsibility for Reporting:

The project manager is responsible for collecting and submitting the input.

Forms

Forms to record the required data can be designed and maintained at the organization level.

Algorithm: The deviation from the planned schedule is calculated based on the number of calendar days the project end date deviates from the planned ship date, expressed as a percentage of the planned duration.

The percent deviation is calculated for each effort category according to the following formula:

$$\text{Percent Deviation} = \frac{\text{absolute value (project end date - planned end date)}}{(\text{Planned end date - start date})} * 100$$



Schedule Predictability – 5

Algorithm: (continued)

The average percent deviation for each effort grouping is plotted for each quarter.

Assumptions:

Schedule deviation is undesirable regardless of whether it is a slip in delivery date or a shipment earlier than planned. The goal of project schedule estimations is accuracy so that others may plan their associated tasks with a high degree of confidence. (A shipment of software a month early may just sit for a month until UAT personnel are free to begin testing.)

- Measurements are based on elapsed calendar days **without adjustment** for weekends or holidays.
- The value reported for planned ship date is the **estimate** of planned ship date **made at the end of the design phase** (start of construction).



Schedule Predictability – 6

Probing Questions:

- Is there a documented process that specifies how to calculate the planned ship date?
- Does the planning process take into account historic data on similar projects?
- Has the customer successfully exerted pressure to generate an unrealistic plan?
- How stable have the requirements been on projects that have large deviation?
- Do delivered projects have the full functionality anticipated or has functionality been reduced to stay within budget?



Schedule Predictability – 7

Evolution: The **breakdown** based on project effort (small, medium, or large) can be modified to look at projects **based on planned duration** (e.g., all projects whose planned duration lies within a specified range). This may lead to optimization of project parameters based on scheduling rules.

Historical data can be used in the future to identify local cost drivers and to fine tune estimation models in order to improve accuracy. **Confidence limits** can be placed around estimates, and root cause analysis can be performed on estimates falling outside these limits in order to remove defects from the estimation process.



Schedule Predictability – 8

Definitions: **Project Effort Categorization:** The completed projects are grouped into the three effort categories (small, medium, large) according to the criteria described in the table below.

Categories	SMALL	MEDIUM	LARGE
Development Effort (hours)	< 200 hrs	200 – 1800 hrs	> 1800 hrs



Operational Definitions

Start & End Date Milestone Definition Checklist

Project Start Date

- ☒ Sign-off of user requirements that are detailed enough to start functional specification
- ☒ Kick-off meeting

Project End Date

- ☒ Actual UAT sign-off by customer

Estimation Start Date

- ☒ Start of code construction



Code Complexity – 1

Indicator objective	Identify components that should be re-engineered or require extra inspection. Re-engineering could involve partitioning or layering to reduce complexity																												
Questions	Which components have the highest complexity values? How many components have complexity values higher than the threshold? List the question(s) the indicator user is trying to answer. Examples: Is the project on schedule? Is the product ready to ship? Should we invest in moving more software organizations to CMMI maturity level 3? (Refer to the Goal Achievement Questions section of the Operationalized Goal Worksheet)																												
Perspective	•Development and project managers •Testers and QA Describe the audience for whom this visual display is intended. (Refer to the name of the perspective in the heading of the Operationalized Goal Worksheet)																												
Visual display	Provide a graphical view of the indicator. (Refer to the Visual Display section of the Operationalized Goal Worksheet). Note: The visual display shown is an example. Component Complexity <table><thead><tr><th>Component</th><th>Coupling</th><th>Function</th><th>McCabe</th></tr></thead><tbody><tr><td>F</td><td>10</td><td>15</td><td>45</td></tr><tr><td>B</td><td>10</td><td>15</td><td>35</td></tr><tr><td>D</td><td>10</td><td>15</td><td>30</td></tr><tr><td>C</td><td>10</td><td>15</td><td>25</td></tr><tr><td>A</td><td>10</td><td>15</td><td>20</td></tr><tr><td>E</td><td>10</td><td>15</td><td>15</td></tr></tbody></table>	Component	Coupling	Function	McCabe	F	10	15	45	B	10	15	35	D	10	15	30	C	10	15	25	A	10	15	20	E	10	15	15
Component	Coupling	Function	McCabe																										
F	10	15	45																										
B	10	15	35																										
D	10	15	30																										
C	10	15	25																										
A	10	15	20																										
E	10	15	15																										



Code Complexity – 2

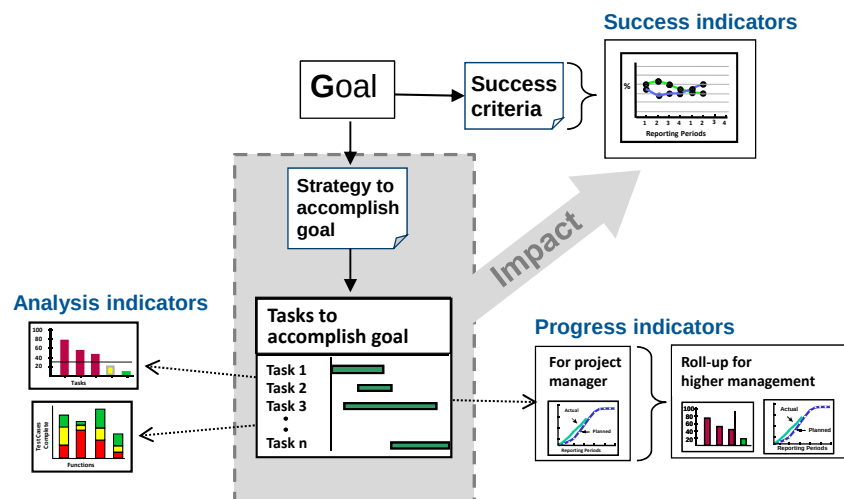
Data elements	Component ID <i>List each data element in the production of the indicator.</i>	(A-F shown above, but names should be used)
	Coupling Complexity	$(Calls_In * Calls_Out) + (Message_In * Message_Out)$
	Functional Size	$Sum(Calls + I/O + DataReferences)$
	McCabe	Directed graph count of paths
How collected	Static code analysis or UML Diagram analysis	
When and how often collected	As part of a design review or code check-in. (Refer to the Source column in the Indicator Availability and Source Worksheet).	
Interpretation	Top 10-20% of components with highest complexity may be candidates for re-engineering if they exceed the threshold <i>Describe what different values of the indicator mean. Make it clear how the indicator answers the "Questions" section above. Provide any important cautions about how the data could be misinterpreted and measures to take to avoid misinterpretation.</i>	



Code Complexity – 3

Probing questions	Is there a design rationale for the high complexity component? Has the component been modified? <i>List questions that delve into the possible reasons for the value of an indicator, whether performance is meeting expectations or whether appropriate action is being taken.</i>
Evolution	A weighted measure of complexity dependent on maturity of the software could be developed. Thresholds could be established based on defect frequency. <i>Specify how the indicator can be improved over time, especially as more historical data accumulates (e.g., by comparison of projects using new processes, tools, environments with a baseline; using baseline data to establish control limits around some anticipated value based on project characteristics.)</i>

Different Types of Measurement Indicators



Indicator Classification

Success

- Show that improvement efforts or product or service has the desired impact from the stakeholder's viewpoint.
- Product achieves revenue or market share
- Improvement reduces cost, rework, cycle-time

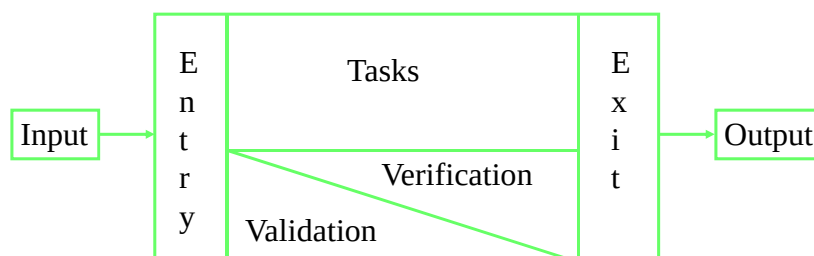
Progress

- Tasks are performed and deliverables produced
- Code growth or test cases passed
- Used to prove plan is viable and direct work efforts.

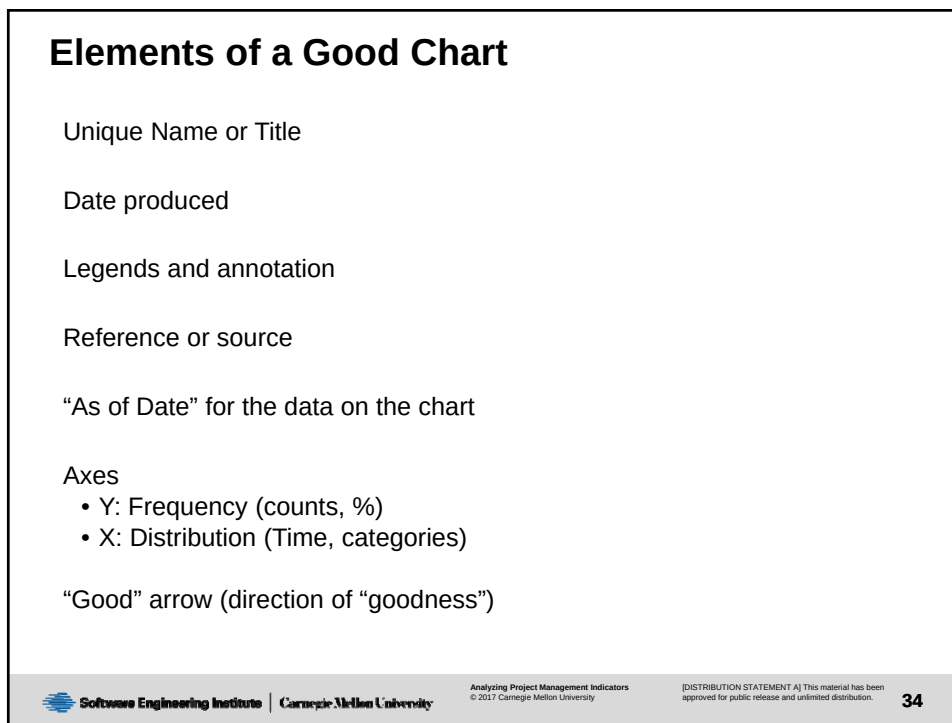
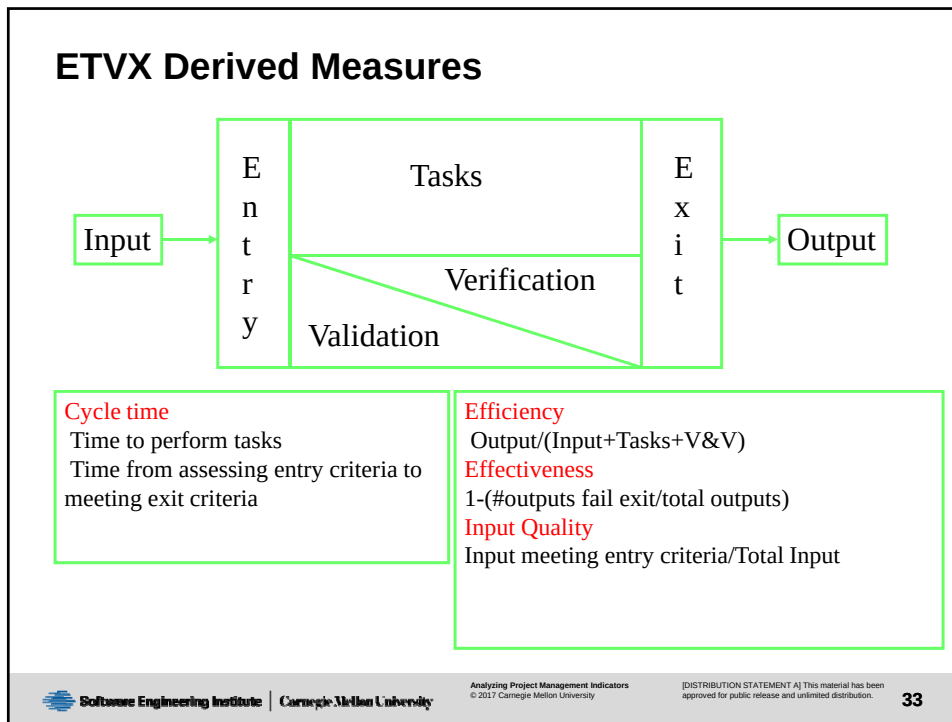
Analysis

- Resources are properly allocated
- Open vs. closed defects
- Used for problem identification and problem solving.

Integrating Measurement and Analysis into Processes



Where are the data generated and how?
Who needs to use the information and when?



Accessible Complexity: the Friendly Data Graphic

Friendly

- Words spelled out
- Words run left to right
- Little messages explaining data
- Labels on the graphic, no legend needed
- Graphic provokes curiosity
- Blue contrasted with others
- Clear, precise, modest type
- Type is mixed case, with serifs

Unfriendly

- Abbreviations abound
- Vertical words or different directions
- Graphics requires references to text
- Encoded shading, colors, cross hatches requires legend
- Chartjunk is repellent
- Red and green contrasted
- Clotted overbearing type
- Type is ALL CAPS SANS SERIF

"Communicating Measurement Results with Graphics", by Doug Smith, can be downloaded at the following URL:
<<http://www.asq0511.org/Presentations/1298/sld001.htm>>



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been
approved for public release and unlimited distribution.

35

Knowledge for Action

1. Valid Information

What's the source of the data?
Has the process for collecting or reporting changed?
What's different this time?

2. Actionable Choices

What's the significance of the information?
Does it represent a risk or problem?
Is a decision required?

3. Follow-Through

What will you monitor after the action? How often?
Can you get the monitoring data you need?
Who is responsible?



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been
approved for public release and unlimited distribution.

36



Module Outline

Background

Description of indicator template

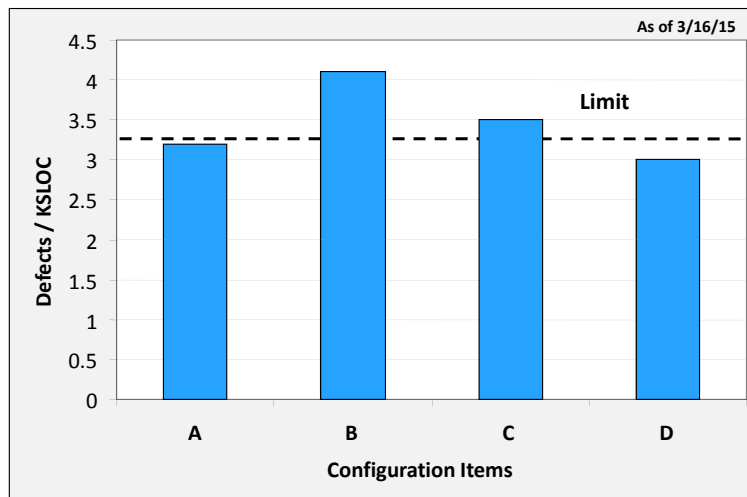
- initial
- example of use

Chart examples

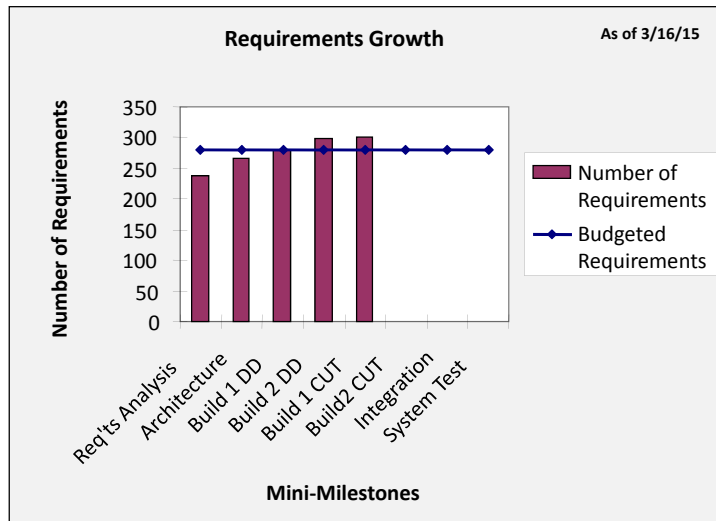
Measurement handbook



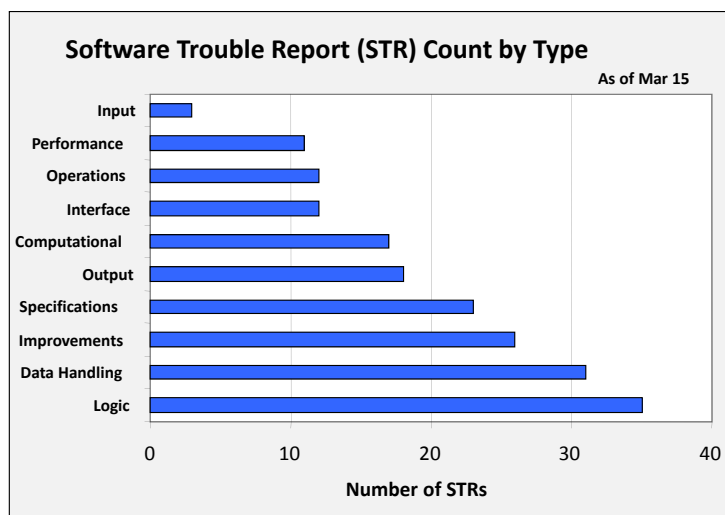
Status Indicator with Limit



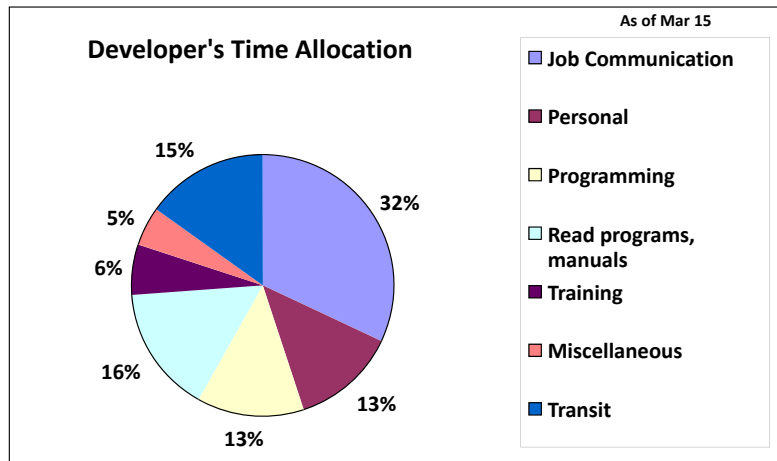
Column Indicator Example



Bar Indicator Example



Pie Chart Example



Stoplights & Gauges

SOFTWARE MANAGEMENT INDICATOR SUMMARY CHART		
Indicator	Stoplight	Trend
Quality Metrics		
Entry and exit points		↘
% Comment lines		→
Span of control		→
Software supportability		→
Complexity		→
Program strength		↗
Release Criteria		
% of tests passed		→
Requirements stability		→
Critical defects		↘
Important defects		→
Build release		→
Computer resources		↗



Trend Indicators

Trend indicators usually show change over time

- planned versus actual lines
- expected differences between two actuals
- actual versus a limit, average, rule-of-thumb
- rate of growth
- growth over time

Issue Management

Issue#	Open-date	Date-close	Days
1	1Mar16	11Mar16	11
2	4Mar16		

Issues and action items are a queue of work. Failure to close them usually represents a missing decision maker or communications link.

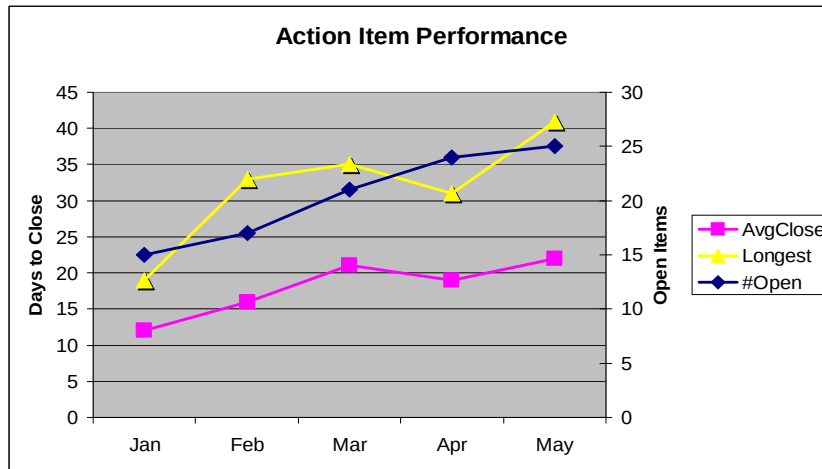
Queue = # open items

Time-to-close = average days open for all items closed this month

Longest item open can also be plotted

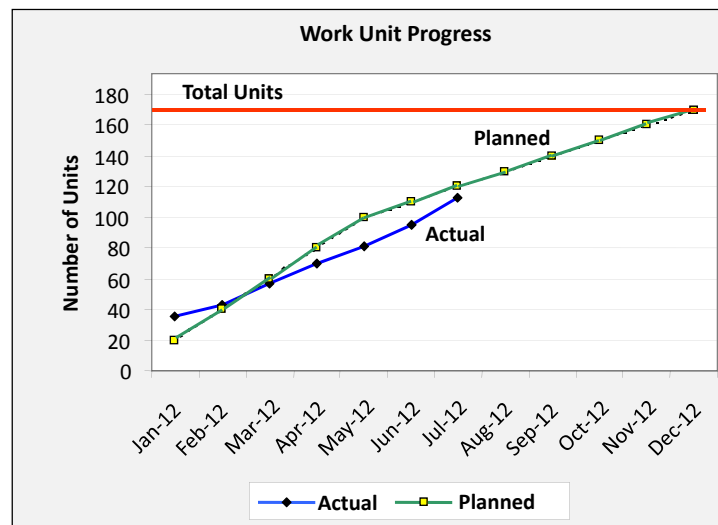
May also measure Days-late instead of Time-to-close

Issue Management

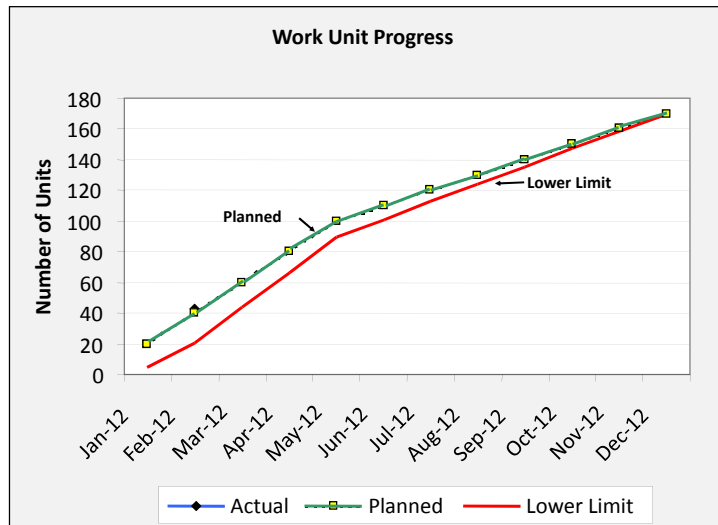


A growing queue is a bad sign

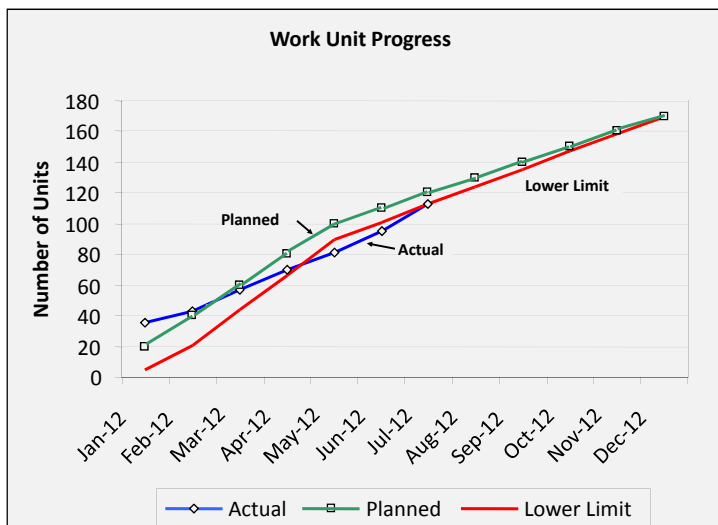
Trend Indicator with Baseline



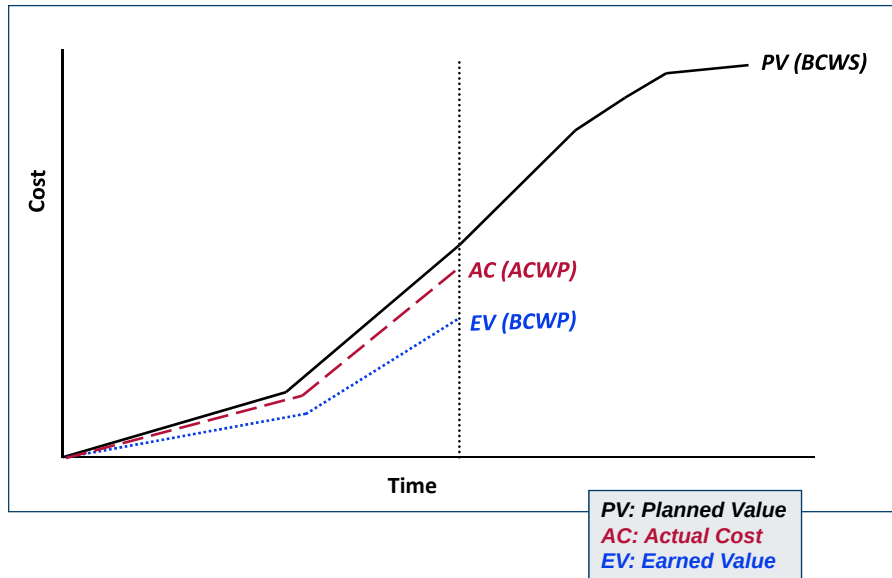
Trend Indicator with Threshold – 1



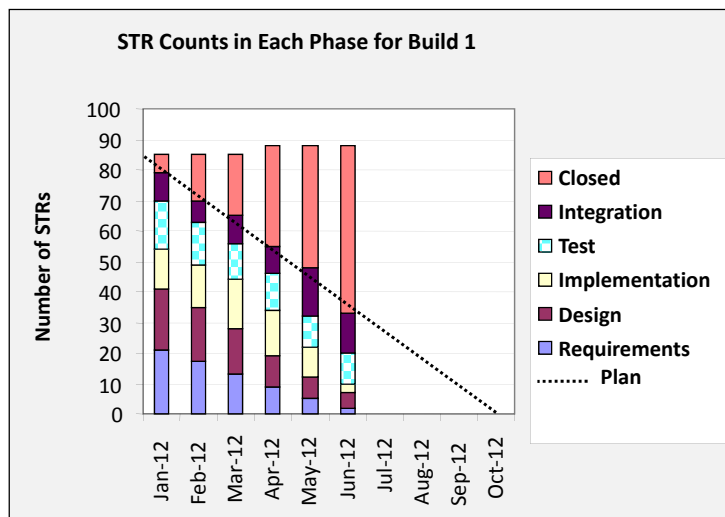
Trend Indicator with Threshold – 2



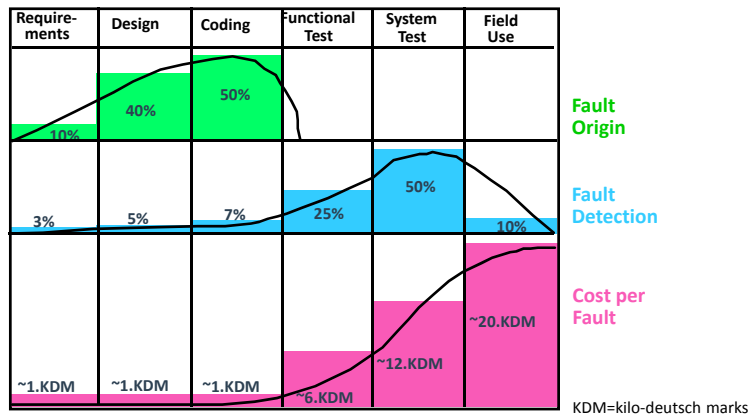
Earned Value Example



Stacked Column Indicator



Combining Information into One Indicator



Data provided by Dr. Paulish of Siemens from "Engineering an Effective Software Measurement Program"



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

51

Phase Containment Chart

Phase Found	Req'ts	Design	Code	Unit Test	Integration Test	System Test	Customer	Total Found by Phase
Phase Originated								
Req'ts	25	8	3	1	2	1	1	41
Design		55	30	10	15	4	1	115
Code			250	67	25	15	7	364
Unit Test				22	4	2	0	28
Integration Test					35	3	0	38
System Test						14	0	14
Customer							0	0
Total by Type	25	63	283	100	81	39	9	600

Risk color based on cell value exceeding historical data model prediction at this point in time



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.

52



Statistical Charts

Statistics can be used to describe data. Common statistical descriptions:

- average
- median
- standard deviation
- percentiles

Statistics can also be used to analyze data:

- relationships between measures
- differences between groups
- explanatory and predictive models

Relating Indicators & Statistics

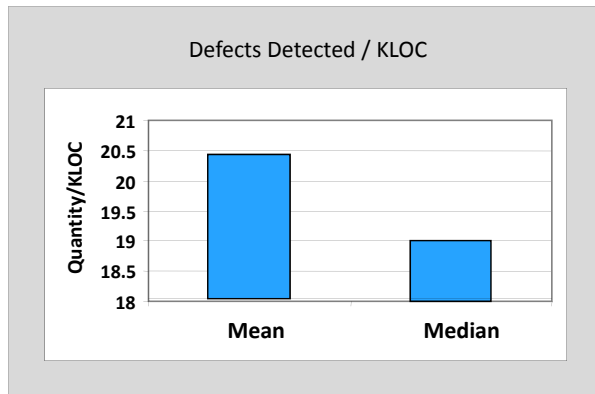
The mean and median statistic can be treated as status indicators when they are compared to:

- quality goals
- limits
- rules of thumb
- completion criteria

If you track the mean or median over time they become a trend indicator.

Descriptive Statistics – Mean/Median

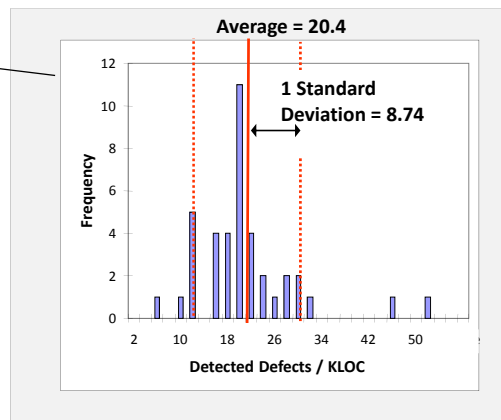
Sorted	
Defects Detected / KLOC	Defects Detected / KLOC
12	52
11	22
52	20
19	19
20	19
15	19
15	15
19	15
19	12
22	11



Descriptive Statistics – Std Deviation

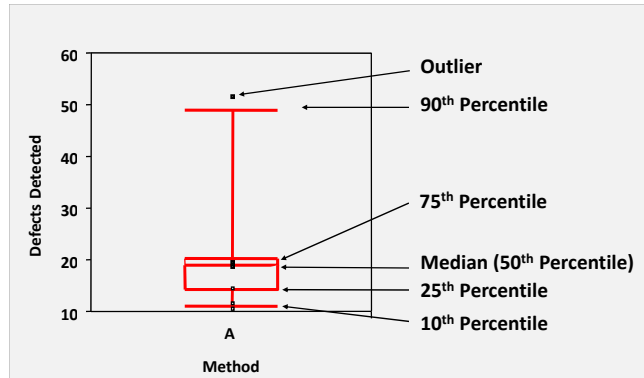
- Measure of the spread of data
- Typically used with average
- May be shown as a number or in histogram form
- With enough data, can characterize the distribution

- Same data from previous slide + 30 additional inspection points
- Average & Median remain the same



Descriptive Statistics – Percentiles

- Alternative way to describe the spread of data
- Typically used with “Median”
- Box plot presentation very useful for comparing data



Ref: Goethert et. al. 98

Using Descriptive Statistics – 2

Data and Tabular Statistics:

Ten comparable modules with same expected defect rates and same inspector skill level

Defects Detected /KLOC			
	Method A	Method B	Method C
	12	15	12
	11	12	11
	52	32	17
	19	20	26
	20	21	30
	15	17	10
	15	19	45
	19	19	21
	19	19	5
	22	30	27
Mean	20.4	20.4	20.4
Median	19	19	19
Standard Deviation	11.66	6.19	11.96

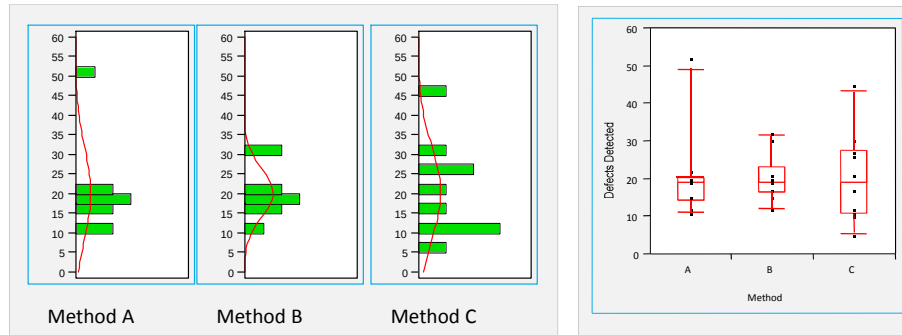
Mean and Median offer no distinction between the data sets

But, there is a difference in the spread of the data

What conclusion would you make at this point?

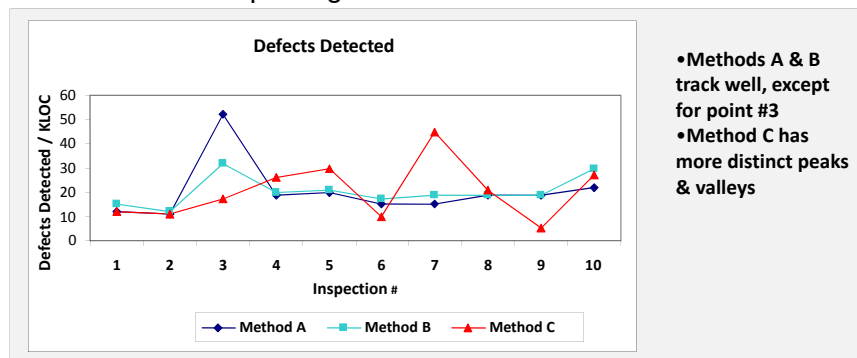
Using Descriptive Statistics – 3

Two Graphical Views of Data Spread



Using Descriptive Statistics – 4

Trend Indicator – placing data into “time” context



What would you like to know about Inspection #3?
About method C?
How would this view of the data influence your decision?

Descriptive Statistics – Key Points

Always represent data so that it preserves the evidence for the predictions that might be made from these data.

Summary data should not mislead the user into taking any action that the user would not take if the data were presented in a time series. Summary data is an average, range or histogram, etc.

Always view the data in multiple (& simple) ways

Statistical Modeling

A statistical model can be employed to explain what has been observed, and/or to predict what will occur based on the past.

Most parametric estimation models are statistical in nature, and the 'parameters' of the model need to be calibrated to local conditions.

While statistical models can be quite powerful, they are only as good as the data you use to calibrate them.

Using Regression Analysis

A scatter plot can be used to look for a relationship.

Correlation or Regression analysis can be used to quantify the relationship.

Linear Regression models have the general form:

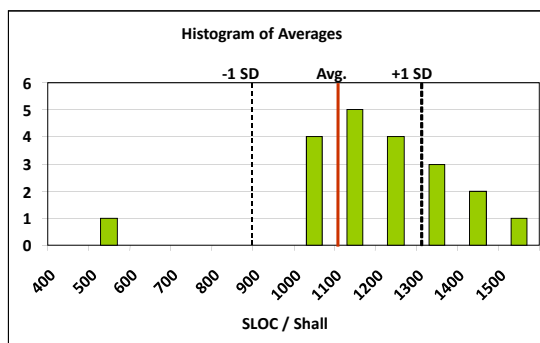
$$Y = a + bX + \dots$$

The value R^2 , is a measure that tells us how well the predictor variables (the X's) in the regression model can predict the response variable (the Y).

- R^2 varies between 0 and 1.
- The closer to 1 or the better the model.

Using an Expansion Factor for SLOC Prediction

- An Expansion Factor for predicting SLOC is 1106 SLOC/Shall.
- The standard deviation is 212.
- What do you think?



Shall's	SLOC	SLOC/Shall
31	14276	458
120	124833	1036
92	90672	982
66	90074	1371
72	64987	902
75	74411	986
27	31099	1163
32	40357	1253
108	135090	1248
103	121974	1179
122	131575	1077
33	48075	1463
114	133671	1174
73	75562	1037
76	89754	1177
70	92343	1315
69	74226	1073
46	47009	1024
61	58626	953
93	116022	1246

Average: 1106

Modeling SLOC Prediction

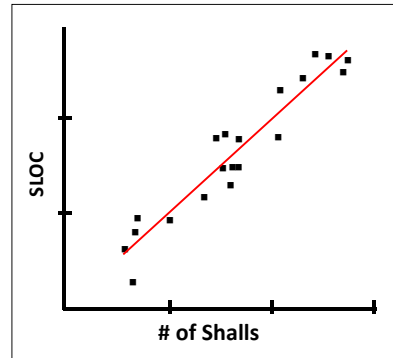
This is a model using the number of "Shall" statements in the requirements (horizontal axis) to predict the software size in SLOC (vertical axis).

The predictor variable is the number of requirement Shalls and the response variable is source lines of code.

$$\text{SLOC} = -2658 + 1152 * (\text{Shall's})$$

$$R^2 = 0.91$$

What do you think of this model?

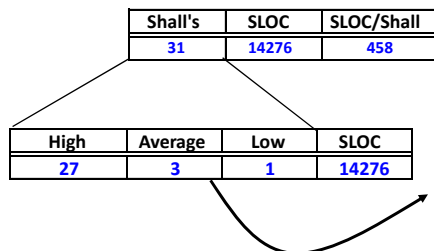


Refining Our Model

The additional information may produce a better model

Classify the requirements by their **precedentedness**:

- high
- average
- low



Shall's			SLOC
High	Average	Low	
27	3	1	14276
25	94	1	124833
32	55	5	90672
5	53	8	90074
31	38	4	64987
23	50	2	74411
9	14	4	31099
2	30	1	40357
13	86	9	135090
19	77	8	121974
21	100	2	131575
6	20	7	48075
24	80	10	133671
29	36	8	75562
21	46	9	89754
3	62	5	92343
27	33	9	74226
28	9	9	47009
23	35	3	58626
2	89	2	116022

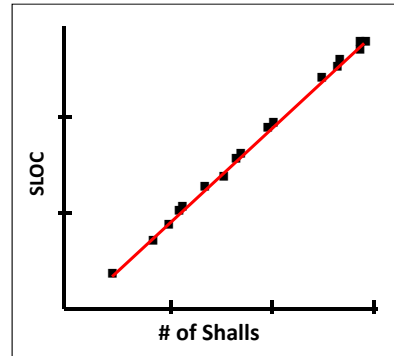
Results of Refined Model of SLOC Prediction

The refined model is:

$$\text{SLOC} = 1137 + 244 * (\text{High}) + 1208 * (\text{Average}) + 3005 * (\text{Low})$$

$$R^2 = 0.99$$

This model predicts SLOC more accurately



Analyzing the Model

In addition to predicting SLOC, we can obtain additional information by analyzing our expanded model:

When a new requirement is given to the project (this is pretty much of a given), it is possible to quickly determine the impact on effort.

- High precedence requirements generally result in an additional 244 SLOC
- Average and Low precedence requirements result in 1208 and 3005 SLOC respectively

The constant, 1137, generally occurs in all of these systems. Is there a way to make the code reusable so not as much time is spent on producing this overhead code?

Uses of Statistical Analysis

Relationships between measures

- size and effort
- code complexity and defect density

Differences between groups

- performance of different inspection methods
- performance of different subcontractors

Explanatory and predictive models

- cost estimation models
- product quality models

Module Outline

Background

Description of indicator template

- initial
- example of use

Chart examples

Measurement handbook

Measurement / Metric Handbook

Most often an indicator is part of a collection of related indicators.

Recommendation

Collect and cross-reference the following indicators in a measurement handbook:

- success indicators
- progress indicators
- analysis indicators



Measurement Handbook

Business goals

- success criteria
- success indicators

Subgoals (perspective 1-n)

- success criteria
- success indicators

Progress indicators

Analysis indicators



Appendix

Success Indicators

Business Goal

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Subgoals

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Progress Indicators

Completed Indicator Templates

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Analysis Indicators

Completed Indicator Templates

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Goal Questions

Indicator Template

Table of Contents



Section 1: Introduction

- Purpose of the SPI Indicators
- Overview
- Lifecycle Phase Coverage
- Software Process Improvement Goals
- Software Measurement Framework
- Software Measurement Process Architecture

Section 2: Indicator Templates for the SPI Indicators

Section 3: Data Element Definitions Checklists

- Defects, Effort, Features

Section 4: Appendix

- Lifecycle Coverage (indicator and data elements)
- Mapping of Data Elements to Indicator
- Mapping of Effort to Size Characterization
- Sample Data Forms
- References
- Glossary and Acronyms



Implementation

Then Just Do It.

Follow the workshop steps laid out earlier.

Integrate with and leverage existing processes.



Summary



The indicator template is used to precisely describe, document, and report the who, what, when, where, why, and how of an organization's indicators.



An organization can amass its completed templates in a measurement or indicator reference handbook.

Ref:

Application of the Indicator Template for Measurement and Analysis (CMU/SEI-2004-TN-024)

<http://www.sei.cmu.edu/publications/documents/04.reports/04tn024.html>



Software Engineering Institute | Carnegie Mellon University

Analyzing Project Management Indicators
© 2017 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] This material has been
approved for public release and unlimited distribution.

75



Analyzing Project Management Indicators



Schedule

Day 1	Day 2	Day 3
1 – Introduction	6 – Progress and Change	9 – Diagnosing Problems
2 – Project Management Dashboard	7 – Quality and Defects	10 – Goal-Driven Indicators and Charting Data
3 – Schedule	8 – Basic Estimation and Planning	11 – Course Summary
4 – Resources and Cost		
5 – Size		

The Project Management Role

As a project manager you

- Use data-based indicators to
 - Monitor and control the project
 - Confirm completeness and meeting objectives
 - Predict the future
 - Identify risks and problems
 - Ask for help

Sources of knowledge

- Project Management Institute (www.pmi.org)
- Practical Systems and Software (www.psmc.com)
- Various SEI papers (<http://resources.sei.cmu.edu/library/>)

Project Management Indicators

Are most successful when they

- have clear objectives
- are interpretable and actionable
- involve the technical staff in deciding what to collect and how
- identify the criteria for evaluating their effectiveness
- support decision making and action based on results

Concluding Thoughts – 1

Measurement is not a strategy unto itself.

- If measurements are not answering your concerns then change the process or change the measures to get what you need.

Measurement indicators have meaning within the context of the issue or question they address.

- Plan v actuals (schedule, cost, staffing, ...)
- Scope/requirements changes
- Quality

Verify the measurement system.

- Know the operational definitions and insist that an Indicator Template be filled out for each indicator.
- Measures may have a different meaning or data collection than you expect.

Concluding Thoughts – 2

Use the **Dashboard** or **Performance Analysis Model** to:

- find the source of problems
- create and interpret leading indicators
- understand the ripple affect of management decisions

Completion criteria must be satisfied before an activity or phase is declared complete.

- Completion criteria must be measurable and observable

Understand the estimation and planning process and the inputs

- Be sure the assumptions and inputs are well documented (the basis of estimate)

Concluding Thoughts – 3

Build your measurement program in small steps.

- Address one or two questions / issues.
- Present and confirm the results of the measures with the developers.
- Get concurrence on the analysis.
- Use the Dashboard or Performance Analysis Model to expand the measures to collect and analyze.

Get contractor buy-in

- “You are a world class developer. You should have this already.”

Don’t collect “nice-to-have,” “just-in-case” indicators.

- It makes measurement appear ineffective because measures are not providing insight into anything of interest.

After the Class – 1

Do these next steps:

1. List what you as a Project Manager are responsible for.
2. Circle the items that need the most immediate attention.
3. For the highest priority items, sketch out an indicator that will give you insight into the status of that item
 - indicator can be a chart, table or list
 - Improve understanding by showing trends, projections and thresholds

After the Class – 2

4. Make sure you know what you want the indicator to accomplish. Be realistic.
5. Try currently available data to build your indicators.
 - what is the source of the data
 - what data is not readily available that you need
 - how hard would it be to get data that you need
6. If the data is not available and too hard to get, then repeat steps 3 through 5.
7. If you are ready, call a meeting with the project team and present your need for more information



For Additional Information

Name	SEMA
Telephone	412 / 268-5856
FAX	412 / 268-5758
Email	SEMA@sei.cmu.edu , dz@sei.cmu.edu rwf@sei.cmu.edu
Online	http://www.sei.cmu.edu
U.S. Mail	Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213-3890 USA