Monitoring Progress

Dr. Mark C. Paulk

SE 4381, Software Project Planning and Management

Management Topics

1. Modern project management

PMBOK

2. Organization strategy and project selection

3. Organization: structure and culture

4. Defining the project

5. Estimating times and costs

6. Developing a project plan

7. Managing risk

8. Scheduling resources and cost

9. Reducing project duration

10. Leadership

11. Teams

12. Outsourcing

→ 13. Monitoring progress

14. Project closure

15. International projects

16. Oversight

17. Agile PM

Critical chain project management

Project Control

“How does a project get one year late?

… One day at a time.”

- **Fred Brooks, The Mythical Man Month**

Control

- **holds people accountable**
- **prevents small problems from mushrooming into large problems**
- **keeps focus**

Project Monitoring System

Involves determining what data to collect

How, when, and who will collect the data

Analysis of the data

Reporting current progress

Typical Questions

What is the current status of the project in terms of schedule and cost?

How much will it cost to complete the project?

When will the project be completed?

Are there potential problems that need to be addressed now?

What, who, and where are the causes for cost or schedule overruns?

What did we get for the dollars spent?

If there is a cost/schedule overrun midway in the project, can we forecast the overrun at completion?

Important Status Information

Start with a summary of what the audience should be aware of

Based on the current WBS and activity network

- **include prior plan info as appropriate in reporting progress**

Includes a graphic describing progress

- **earned value chart**
- **burndown chart**
- **milestone / tracking charts**
- **Gantt chart**

Includes status against identified risks

- **status is NOT the same as impact / probability**
- **risks evolve as the project progresses**

Key Measurement Questions

Are we measuring the right thing?

- **Goal / Question / Metric (GQM)**
- **business objectives ⇔ data**
    - **cost (dollars, effort)**
    - **schedule (duration, effort)**
    - **functionality (size)**
    - **quality (defects)**

Are we measuring it right?

- **operational definitions**

Goal-Driven Measurement

Goal / Question / Metric (GQM) paradigm

- *V.R. Basili and D.M. Weiss, "A Methodology for Collecting Valid Software Engineering Data," IEEE Transactions on Software Engineering, November 1984.*

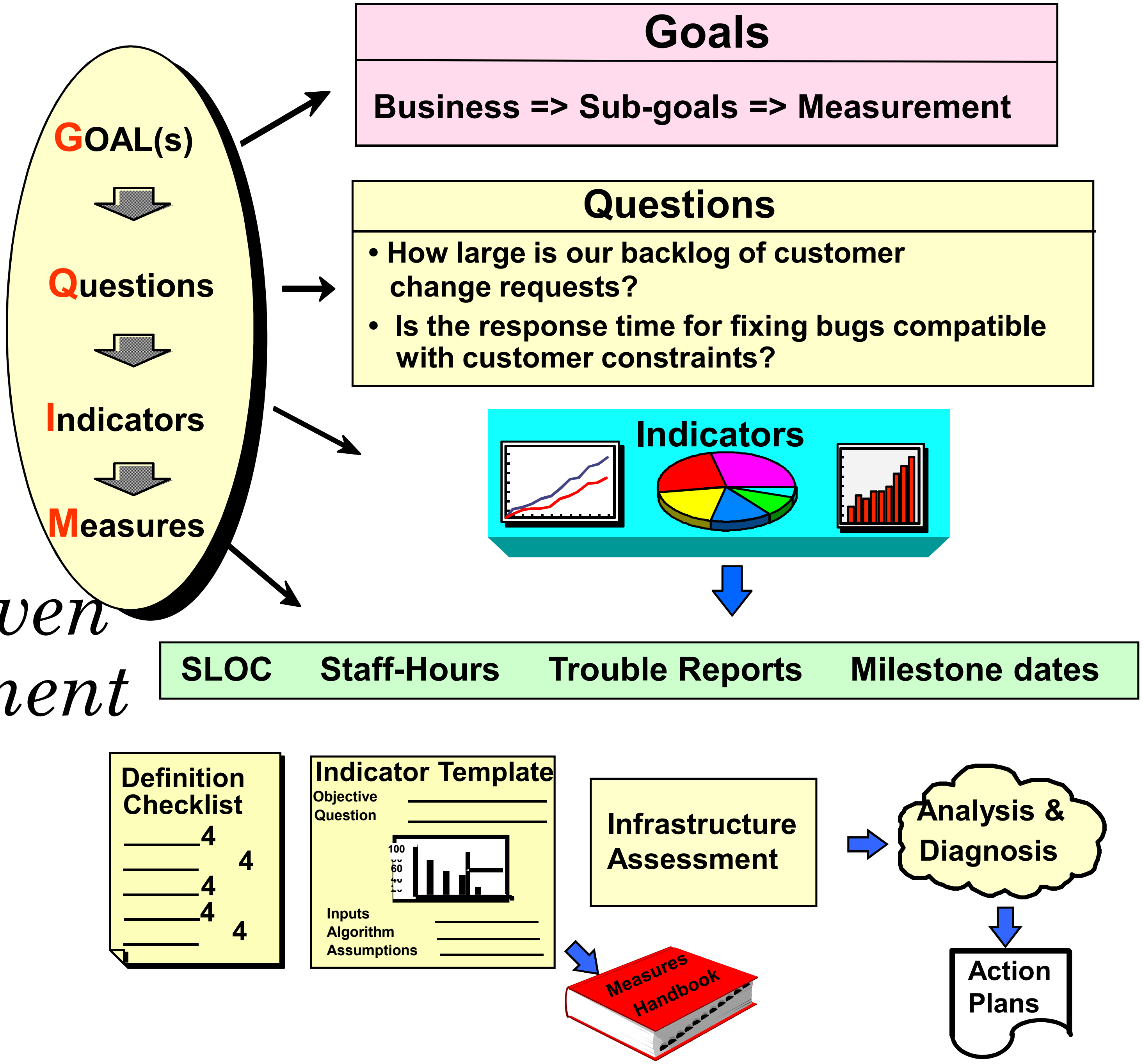
SEI variant: goal-driven measurement

- *Robert E. Park, Wolfhart B. Goethert, and William A. Florac, "Goal-Driven Software Measurement – A Guidebook," CMU/SEI-96-HB-002, August 1996.*

ISO 15939 and PSM variant: measurement information model

- *John McGarry, David Card, et al., <u>Practical Software Measurement: Objective Information for Decision Makers</u>, Addison-Wesley, Boston, MA, 2002.*

Goal-Driven Measurement



GOAL(s)

Questions

Indicators

Measures

Goals

Business => Sub-goals => Measurement

Questions

- How large is our backlog of customer change requests?
- Is the response time for fixing bugs compatible with customer constraints?

Indicators

SLOC Staff-Hours Trouble Reports Milestone dates

Definition Checklist

Indicator Template

Objective

Question

Inputs

Algorithm

Assumptions

Measures Handbook

Infrastructure Assessment

Analysis & Diagnosis

Action Plans

SEI Core Measures

Dovetails with SEI's adaptation of goal-driven software measurement

Checklist-based approach with strong emphasis on operational definitions

Measurement areas where checklists have already been developed include:

- **effort (person hours)**
- **size (SLOC)**
- **schedule**
- **quality (defects)**

See http://www.sei.cmu.edu/measurement/index.cfm

MITRE Metrics for Software Management

Software size

Software volatility

Design complexity

Design progress

Testing progress

Software personnel

Computer resources utilization

Schedule progress

Computer software unit development progress

Incremental release content

H. Schultz, "Software Management Metrics," MITRE, ESD-TR-88-001, May 1988.

Hewlett Packard's FURPS

Functionality

- **capabilities, security**

Usability

- **human factors, aesthetics, consistency, documentation**

Reliability

- **recoverability, MTTF, frequency/severity of failure, predictability, accuracy**

Performance

- **speed, efficiency, resource consumption, throughput, response time**

Supportability

- **testability, extensibility, adaptability, maintainability, compatibility, configurability, servicability, installability, localizability**

Putnam and Myers' Five Core Metrics

Size

- **quantity of function, usually in SLOC or function points**

Productivity

- **functionality produced for the time and effort expended**

Time

- **duration of the project in calendar months**

Effort

- **amount of work expended in person-months**

Reliability

- **defect rate (or mean time to defect)**

Operational Definitions

The rules and procedures used to capture and record data

What the reported values include and exclude

Operational definitions should meet two criteria

- ***Communication*** **– will others know what has been measured and what has been included and excluded?**
- ***Repeatability*** **– would others be able to repeat the measurements and get the same results?**

SLOC Definition Considerations

Whether to include or exclude

- **executable and/or non-executable code statements**
- **code produced by programming, copying without change, automatic generation, and/or translation**
- **newly developed code and/or previously existing code**
- **product-only statements or also include support code**
- **counts of delivered and/or non-delivered code**
- **counts of operative code or include dead code**
- **replicated code**

When the code gets counted

- **at estimation, at design, at coding, at unit testing, at integration, at test readiness review, at system test complete**

Example of Schedule Considerations

Critical Design Review (CDR) was completed 15 June 2013

What does this mean?

- **the internal review was completed?**
- **the review was held?**
- **high-priority action items were closed?**
- **all action items were closed?**
- **sign-off from the customer was obtained?**

Technical Performance Measurement

Typically nonfunctional requirements (quality attributes)

- **availability**
- **interoperability**
- **modifiability**
- **performance**
- **security**
- **testability**
- **usability**

Need to be specified (negotiated) with the customer

Dysfunctional Behavior

Austin's Measuring and Managing Performance in Organizations

- **motivational versus information measurement**

Deming strongly opposed performance measurement, merit ratings, management by objectives, etc.

Dysfunctional behavior resulting from organizational measurement is inevitable unless

- **measures are made "perfect"**
- **motivational use impossible**

Control

Control is the process of

- comparing actual performance against plan to identify deviations
- evaluating possible alternative courses of actions
- taking appropriate corrective action

Project control steps

- set up a baseline plan
  - based on WBS, activity network, resource scheduling
- measure progress and performance
  - use earned value to obtain a realistic estimate of performance
- compare plan against actual
- take action

Project Status Tools

Tracking Gantt Chart

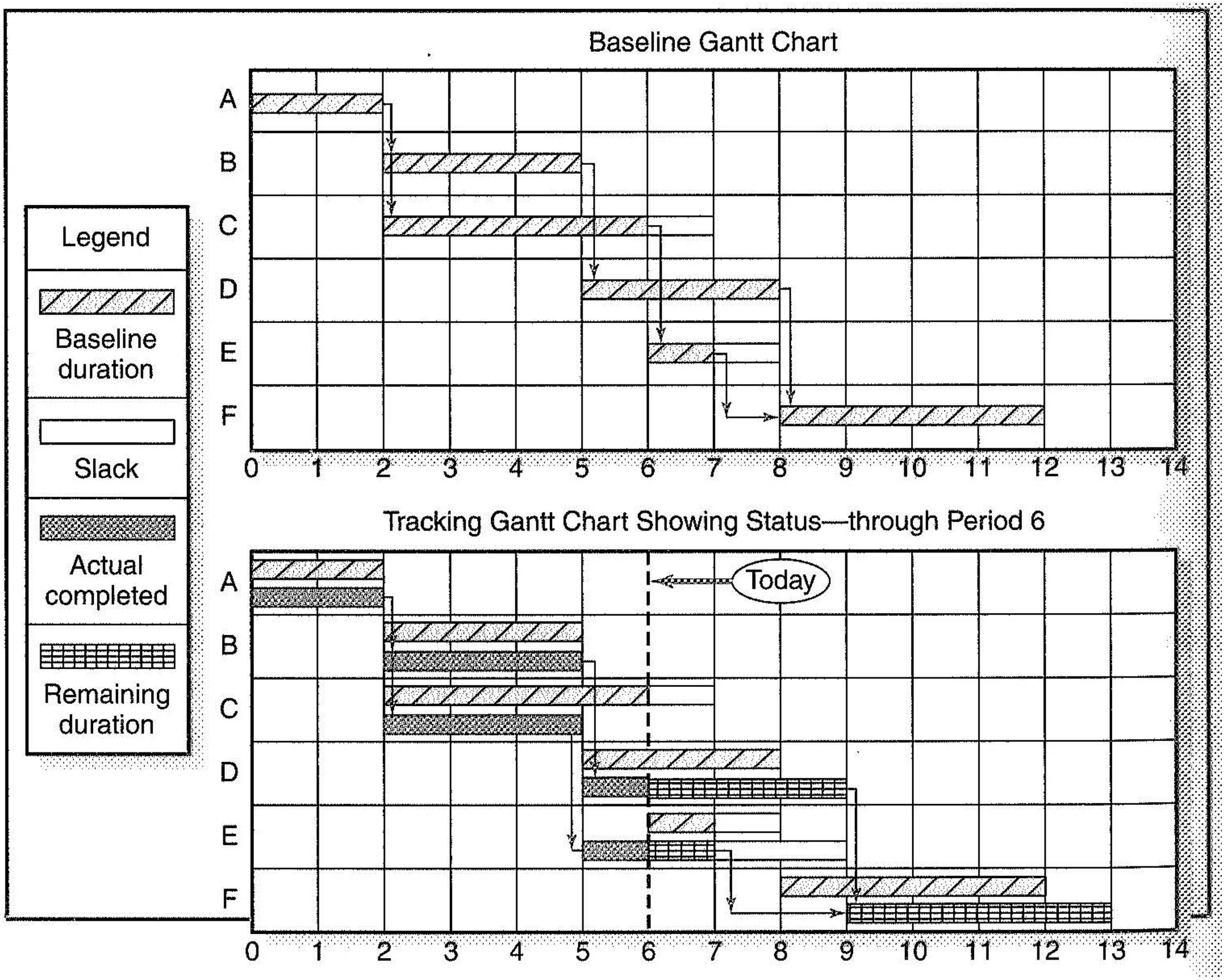
- **baseline Gantt chart shows original plan**
- **modern Gantt charts shows precedence network relations**

Control Chart

- **not Shewart’s control chart for statistical process control!**
- **plots difference between scheduled and actual time on critical path**

Gantt Chart

Larson and Gray, Figure 13.1



Gantt Chart Key

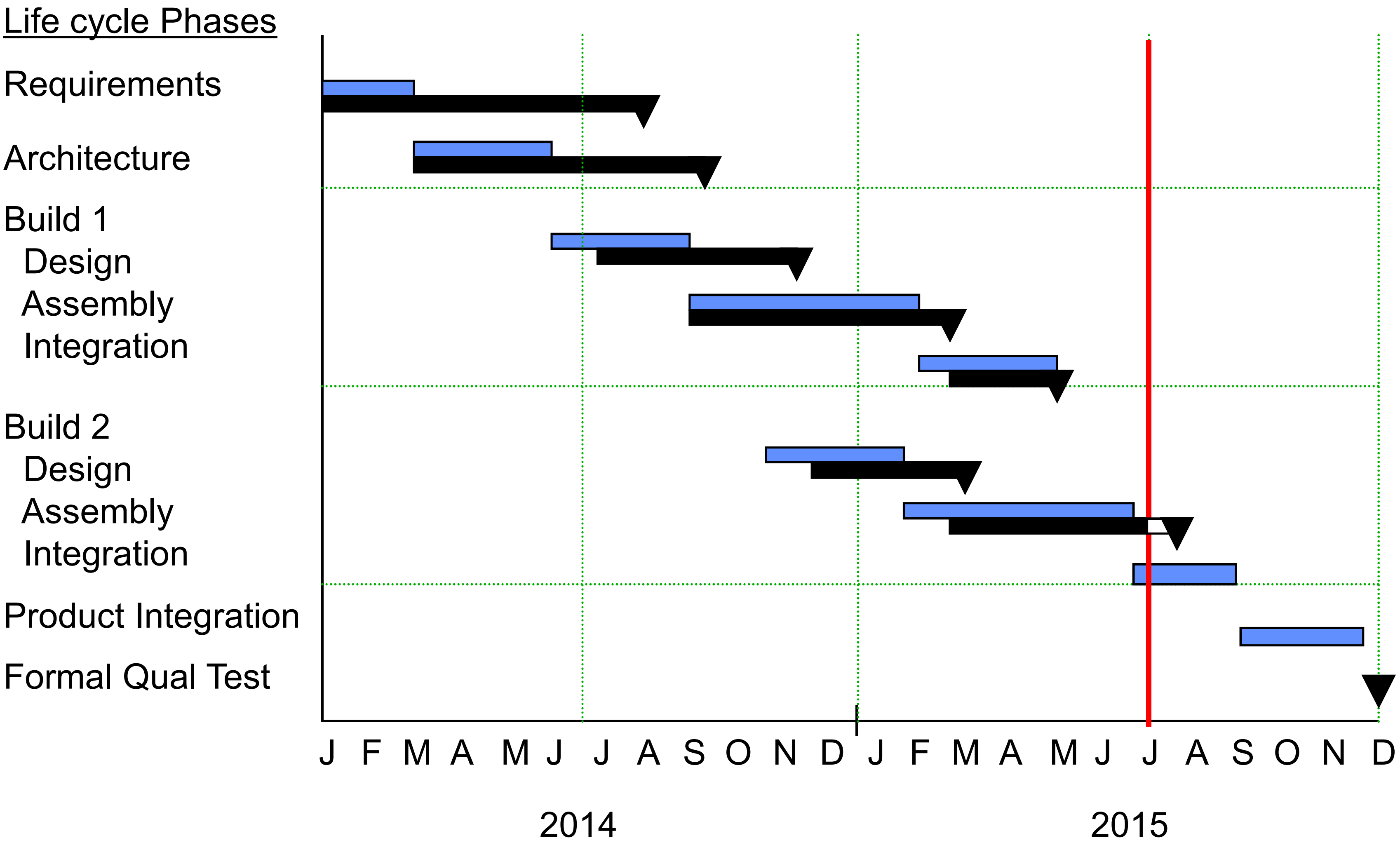
Baseline Gantt chart shows original schedule

Tracking Gantt chart

- **solid bar below original schedule bar – actual start and finish times for completed activities**
- **activities in process show actual start time**
- **extended bar shows expected remaining duration (re-estimated)**

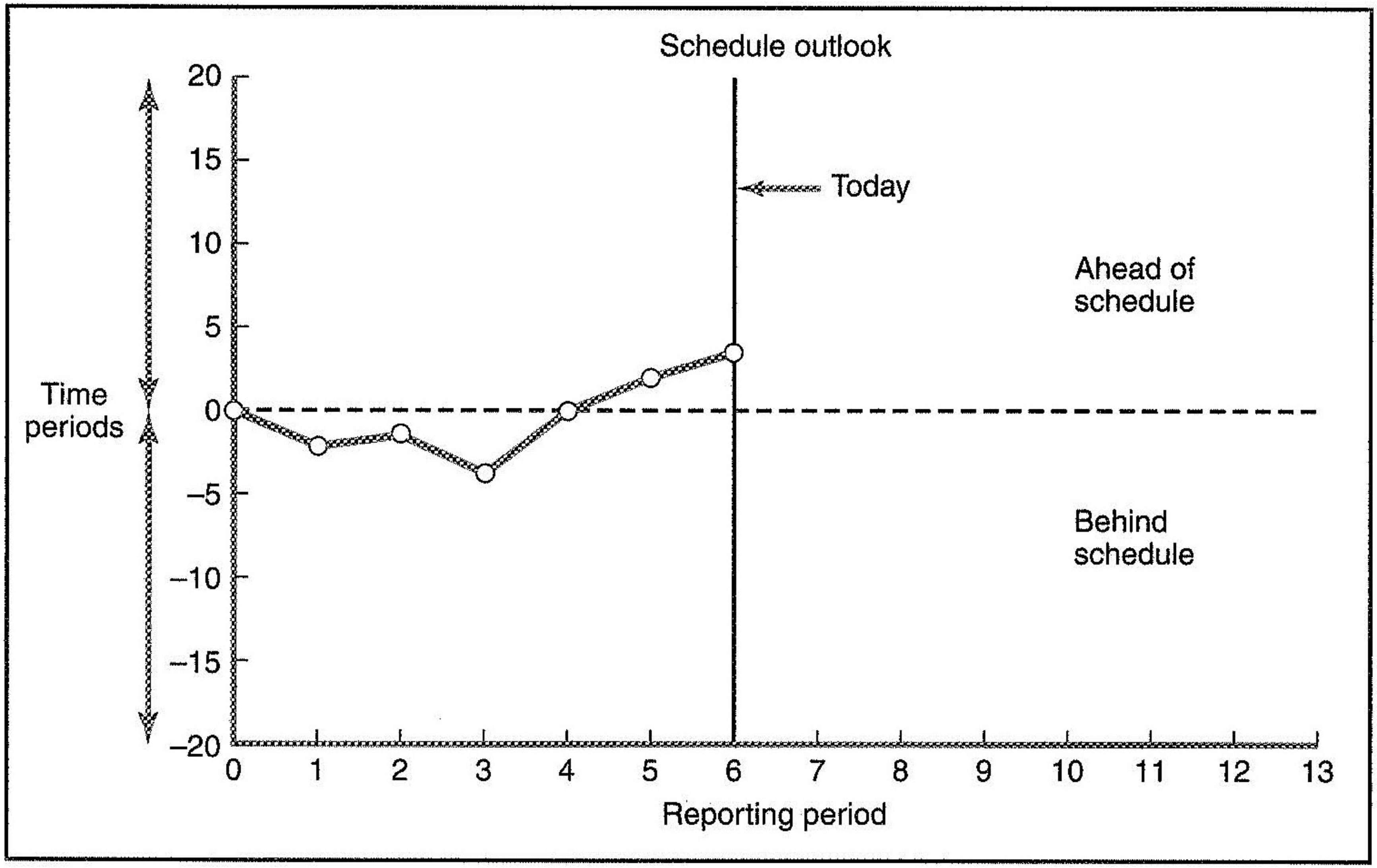
Another Example of a Gantt Chart

Early start… Late start… SLACK!



“Control” Chart

Larson and Gray, Figure 13.2



Tracking with CCPM Buffers

Put the "project buffer" at the end of the critical path.

Put a "feeding buffer" where a path merges with the critical path.

Progress is measured only on the critical path.

Monitor all the buffers.

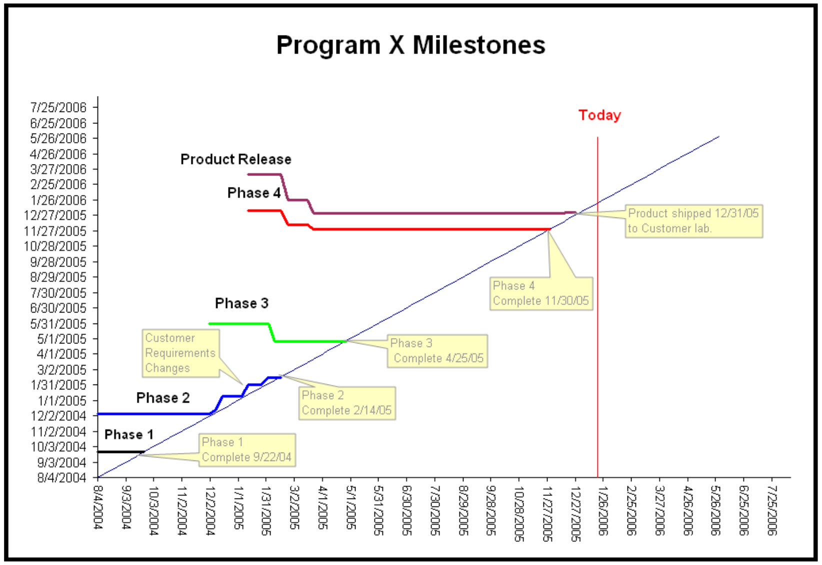
- **priority list: 1) project buffer; 2) feeding buffer**

If there is resource contention, the critical chain may not be the same as the critical path…

Example: Two Simple Tracking Measures

(example from Pete Fuenfhausen, "Sr. Management Reviews: Simple Metrics for Monitoring and Motivating Project Teams," ASEE, Dallas, 2006)

Milestone Chart



Program X Milestones

Customer Requirements Changes

Phase 1 Complete 9/22/04

Phase 2 Complete 2/14/05

Phase 3 Complete 4/25/05

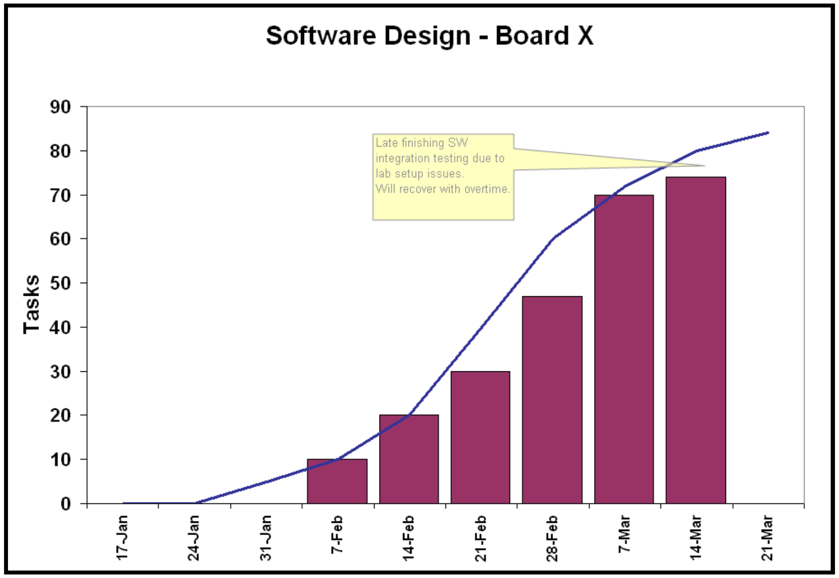
Phase 4 Complete 11/30/05

Product shipped 12/31/05 to Customer lab.

Used to monitor milestones and commitments

Used at multiple levels in programs

Tracking Chart



Software Design - Board X

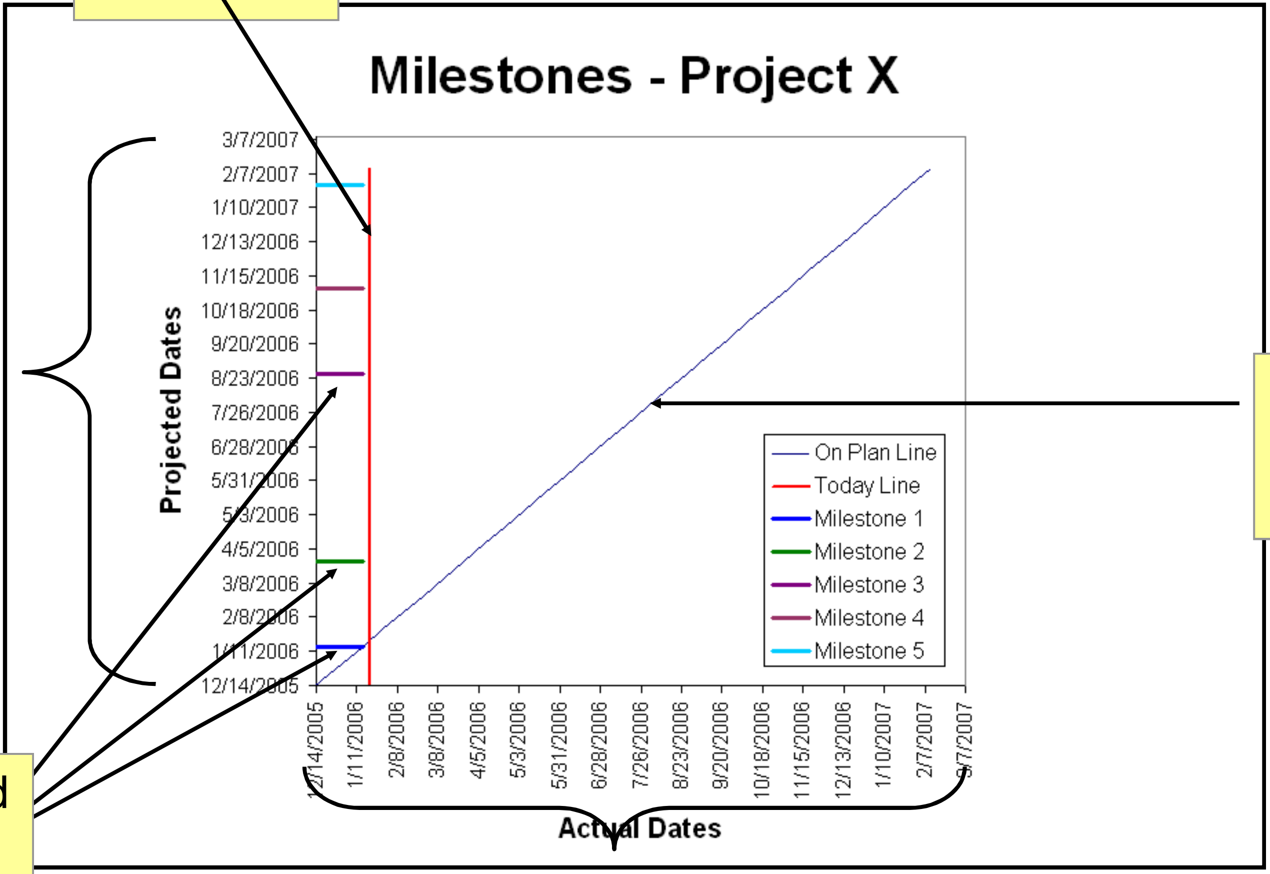
Late finishing SW integration testing due to lab setup issues.
Will recover with overtime.

Used to monitor anything that can be planned or projected vs actuals

Multiple variants depending on what is being tracked

The Milestone Chart - Basics

Today Line



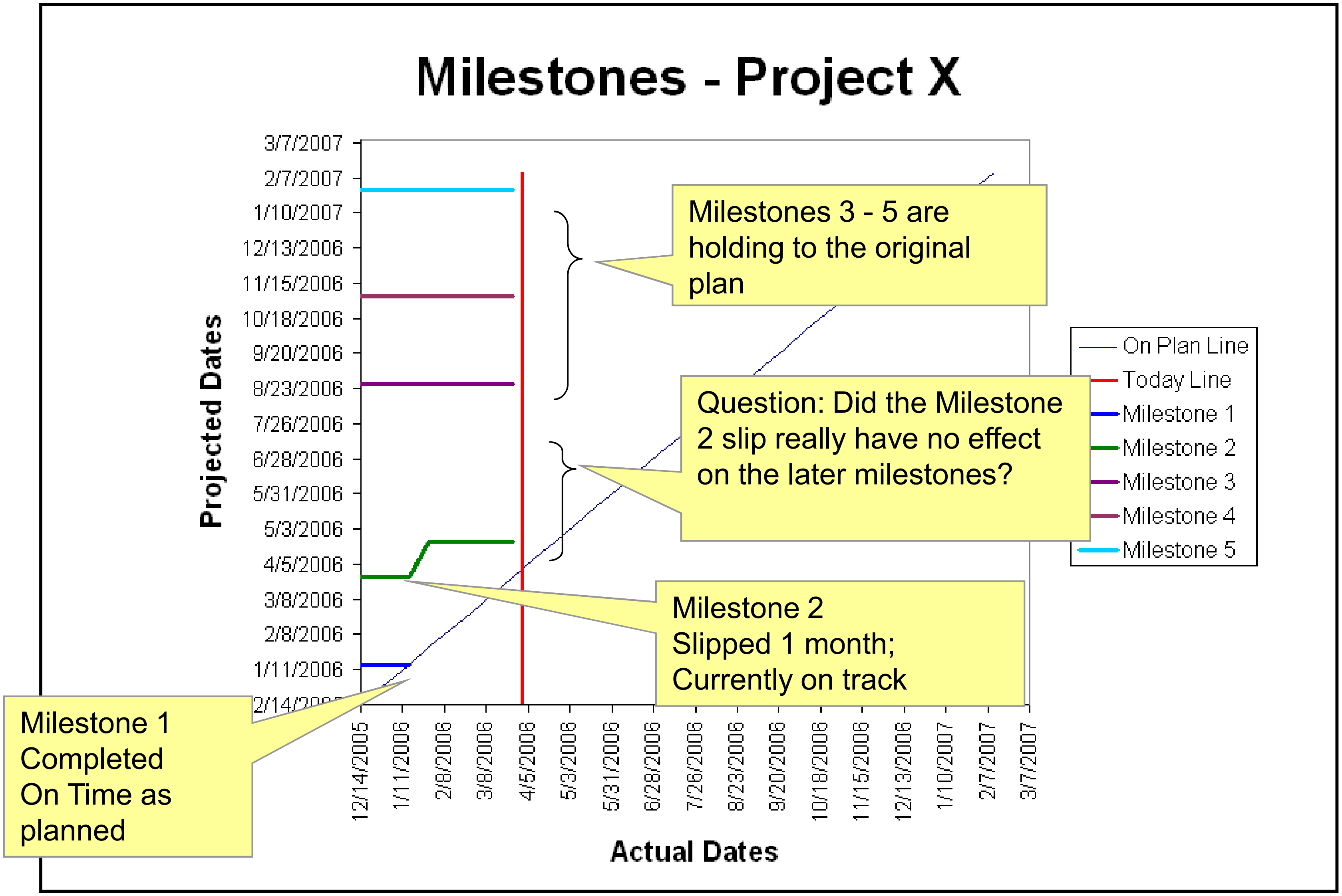
Y Axis: Projected Dates

On Plan Line

Committed Milestone Date Lines

X Axis: Actual Dates

Tracking with the Milestone Chart

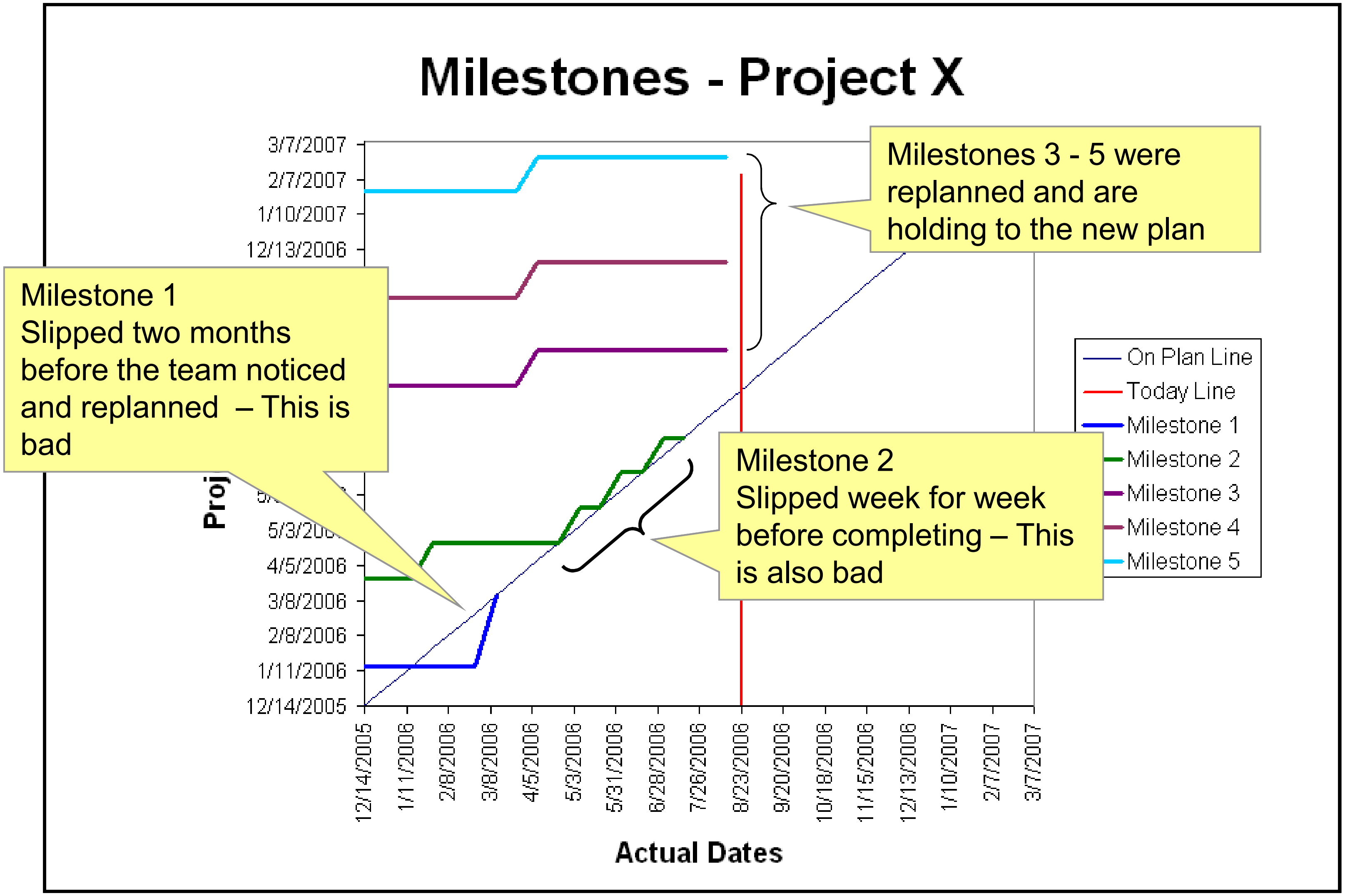


Milestones 3 - 5 are holding to the original plan

Question: Did the Milestone 2 slip really have no effect on the later milestones?

Milestone 2 Slipped 1 month; Currently on track

Milestone 1 Completed On Time as planned

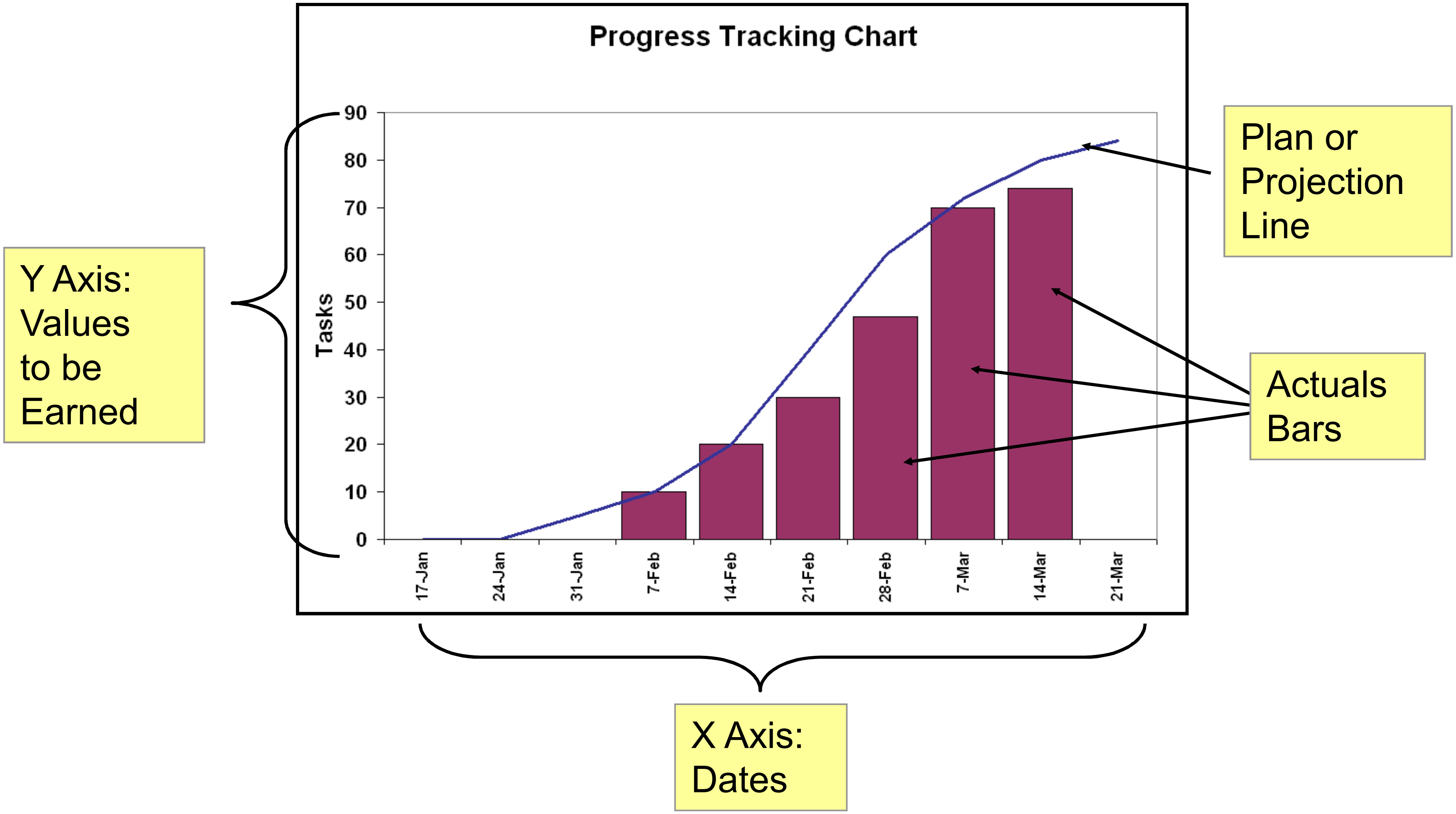


Milestones 3 - 5 were replanned and are holding to the new plan

Milestone 1
Slipped two months before the team noticed and replanned – This is bad

Milestone 2
Slipped week for week before completing – This is also bad

The Tracking Chart - Basics



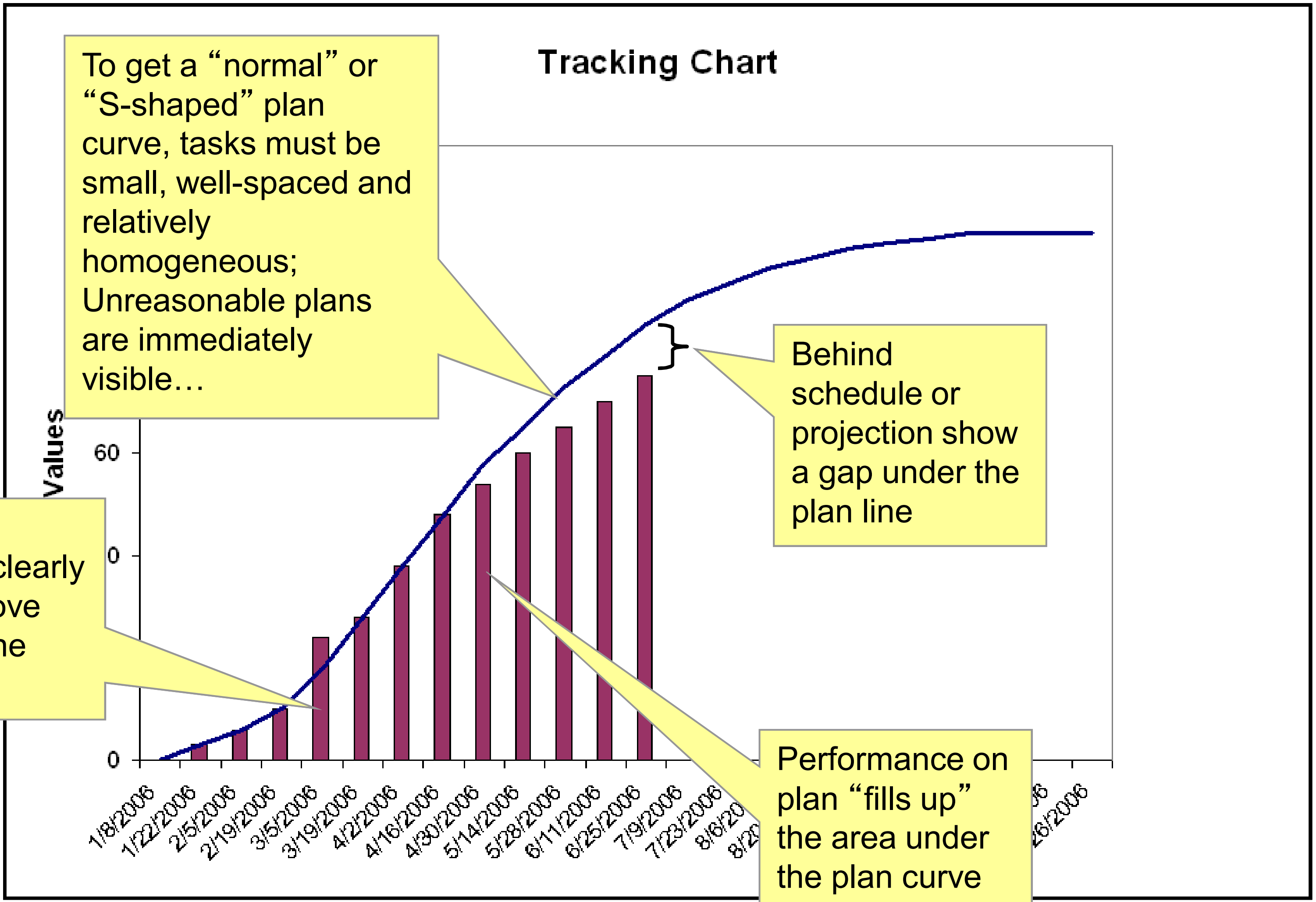
Plan or Projection Line

Y Axis: Values to be Earned

Actuals Bars

X Axis: Dates

Monitoring with the Tracking Chart



To get a “normal” or “S-shaped” plan curve, tasks must be small, well-spaced and relatively homogeneous; Unreasonable plans are immediately visible…

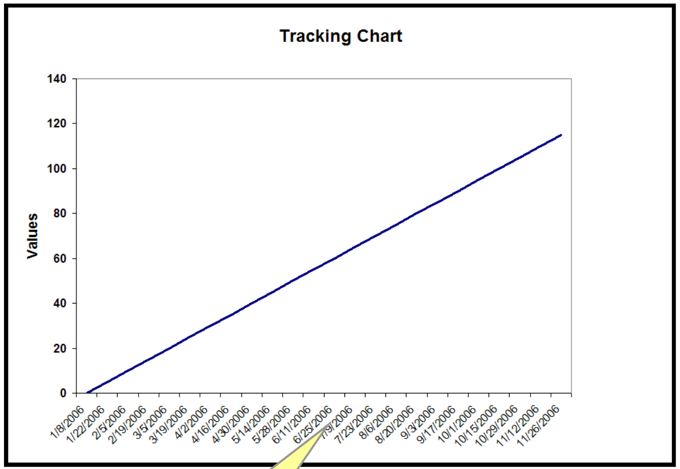
Behind schedule or projection show a gap under the plan line

Ahead of schedule clearly shows above the plan line

Performance on plan “fills up” the area under the plan curve

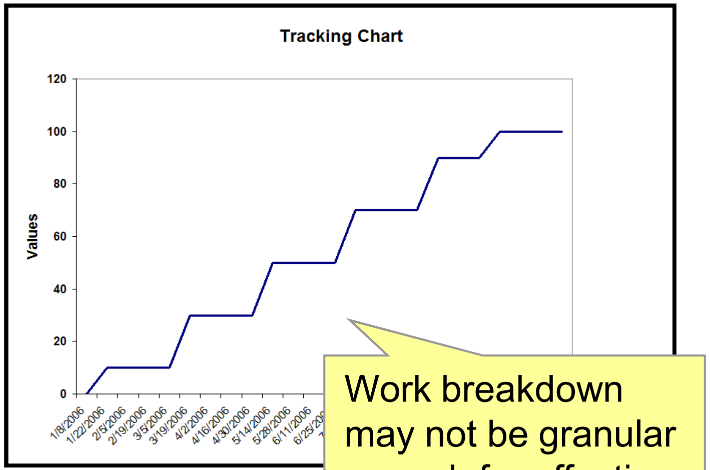
Suspect Plans

The “Straight Line”



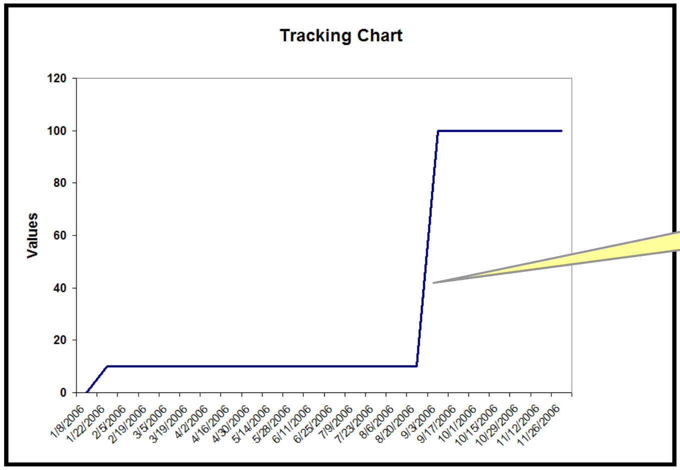
Usually indicates that not a lot of thought went into the planning

The “Step Function”



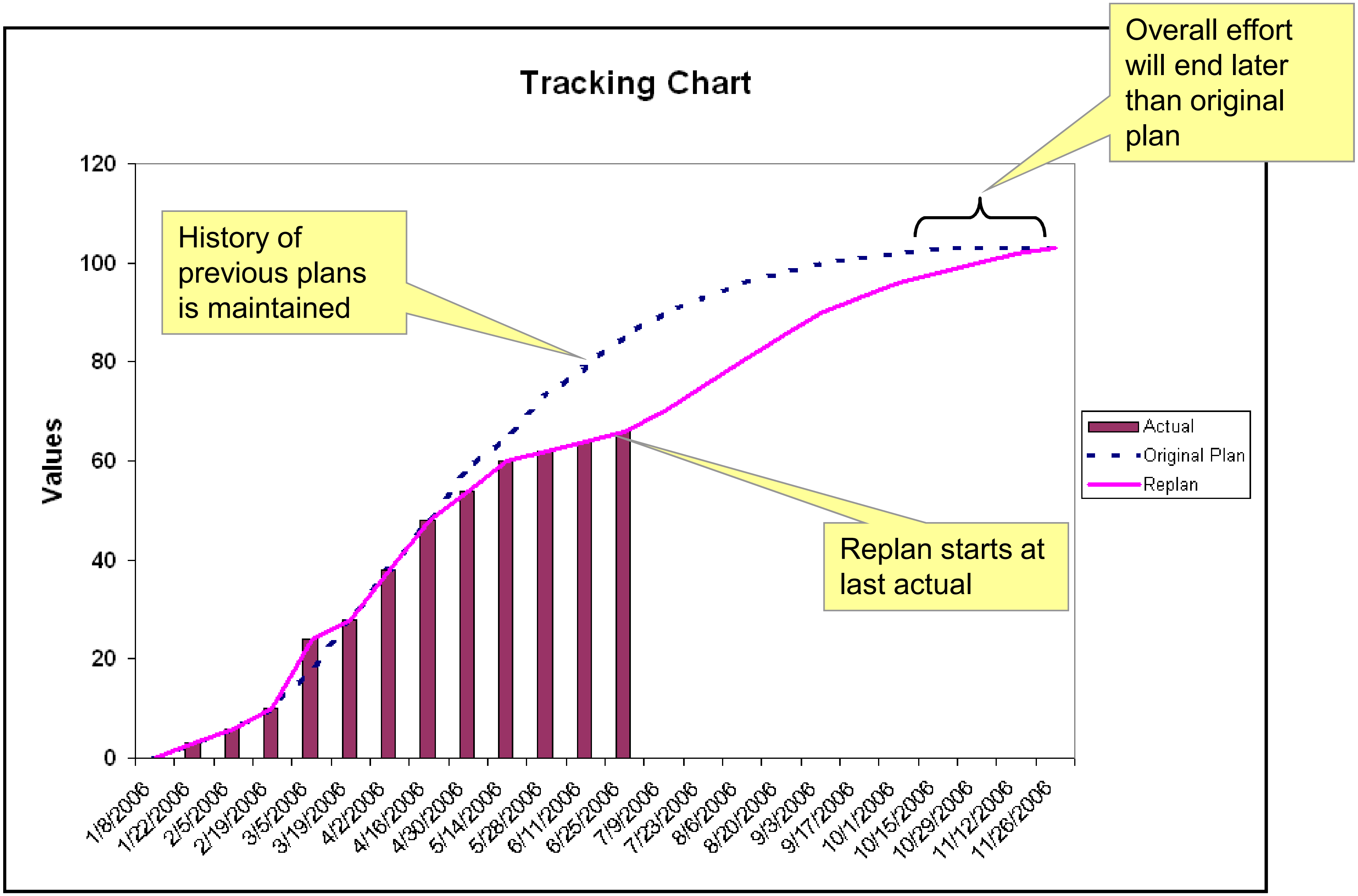
Work breakdown may not be granular enough for effective tracking

The “Cliff”



More of a “prayer” than a plan

Replanning with the Tracking Chart



Overall effort will end later than original plan

History of previous plans is maintained

Replan starts at last actual

Estimate at Completion (EAC)

Forecast of most likely total project costs based on project performance and risk quantification.

EAC_{re} = actuals to date plus a new estimate for all remaining work

- **original estimating assumptions were fundamentally flawed**

EAC_f = actuals to date plus remaining budget modified by a performance factor

- **current variances are seen as typical of future variances**
- **EAC_f = (work remaining) / CPI = (BAC – EV) / (EV / AC)**

TCPI = actuals to date plus remaining budget

- **current variances are seen as atypical**
- **To Complete Performance Index = (BAC – EV) / (BAC – AC)**

Managing Requirements Volatility

Scope creep

- **“minor” refinements**
  - **“Schedules slip one day at a time.” F. Brooks**
- **goldplating**

The key to managing scope creep is change management.

- **Requirements Management in CMM(I) is basically putting the requirements under configuration management**

Tracking Chart Variants

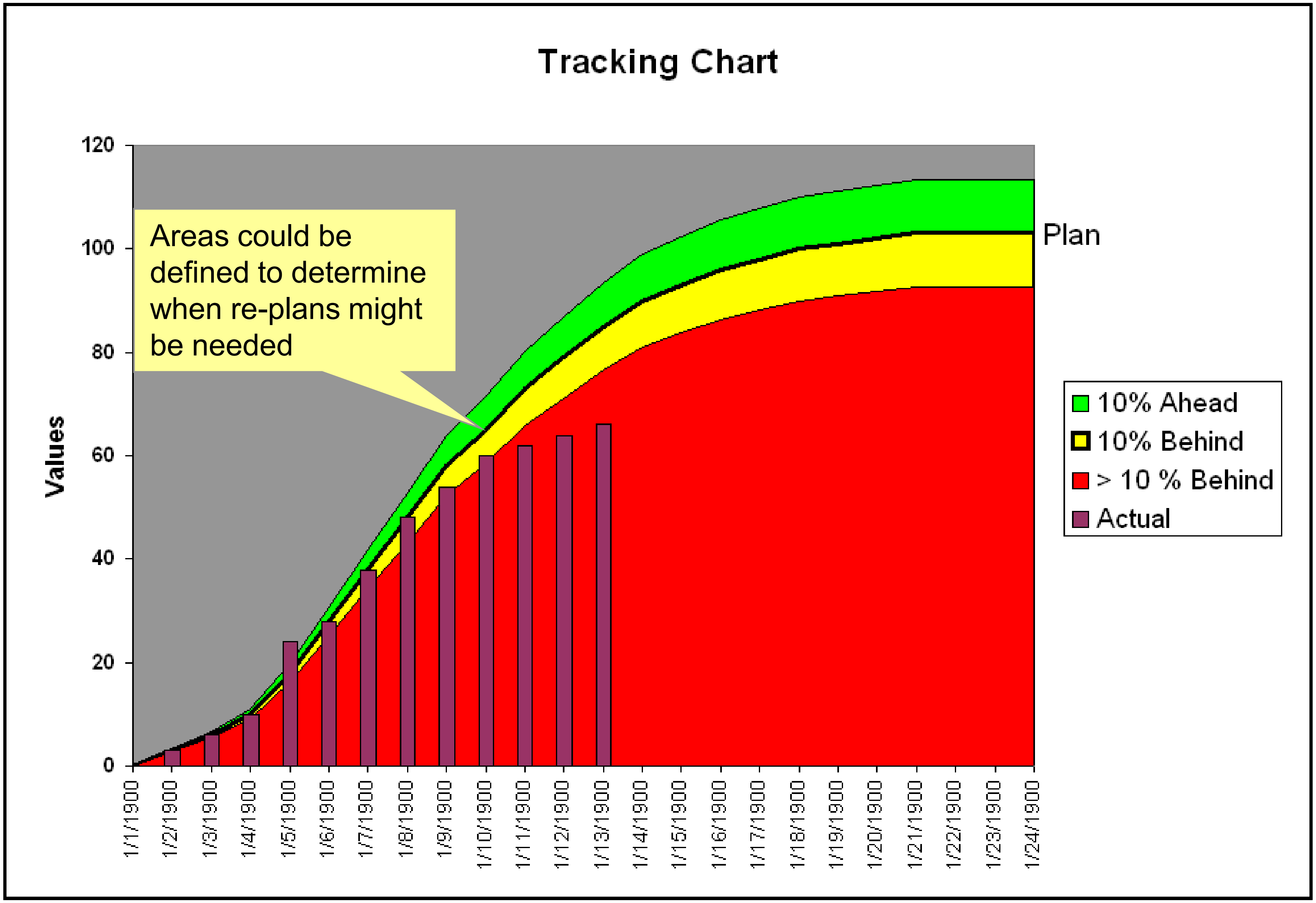
Most tracking charts measure tasks completed against the plans, but…

Virtually any project attribute can be tracked in a consistent manner using a variant of the basic tracking chart format:

- **financial data (plan vs actual spent)**
- **labor and staffing**
- **requirements developed**
- **requirements changes – arrivals and fixes**
- **source code produced**
- **defects detected / fixed**
- **test cases executed**
- **technical measures (memory usage, processor throughput, etc.)**
- **audits performed**

Advanced Uses of the Tracking Chart

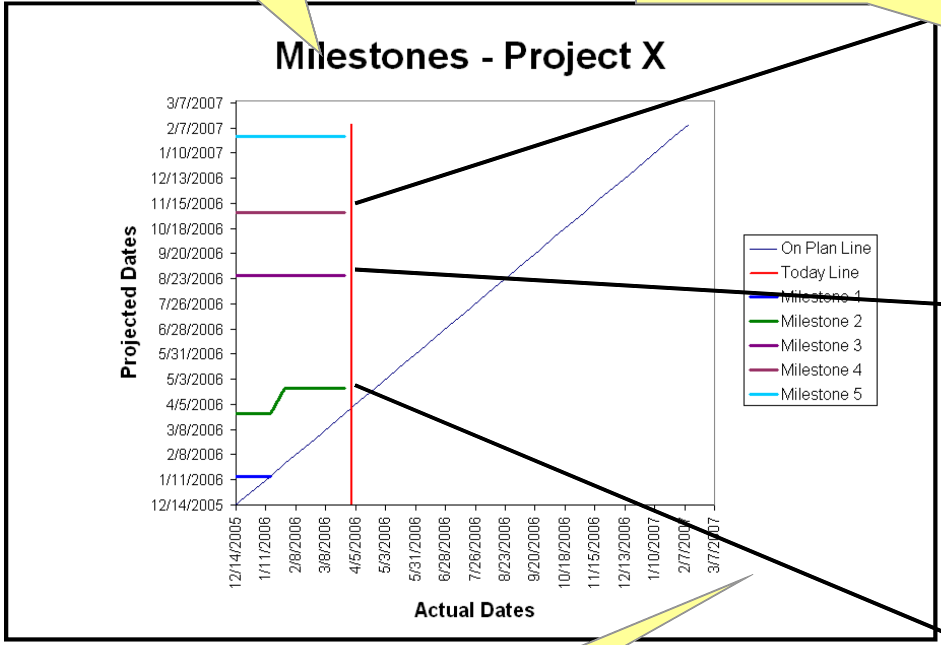
Setting Management Thresholds



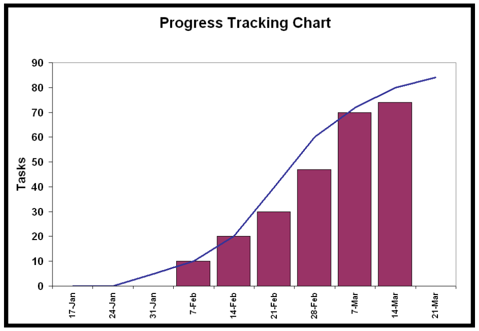
Areas could be defined to determine when re-plans might be needed

Milestone and Tracking Charts Play Well Together

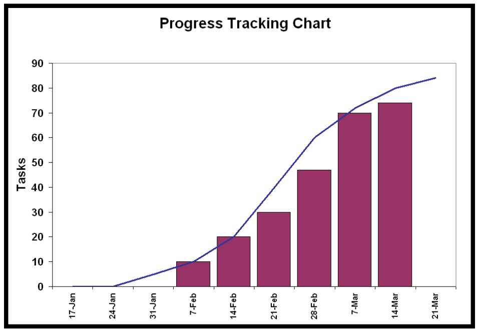
Milestone charts for the entire program are reviewed each time (biweekly)

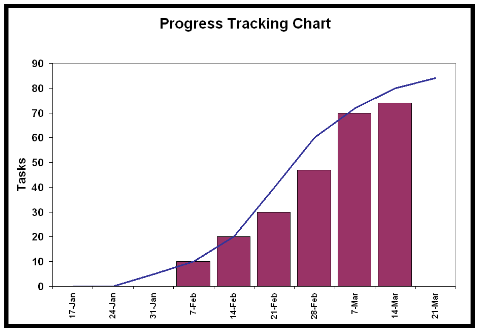


Inconsistencies between milestone and tracking charts are identified and questioned



Sliding Window





Tracking charts are produced for each milestone

Only the tracking charts with current relevance are reviewed

Putting It Together for a Project

A typical program review runs as follows

- **top program risks**
- **program-level Milestones Chart**
  - **milestones are the major program phases and deliveries**
- **financial Tracking Charts**
  - **capital and expense spending**
- **staffing profile Tracking Chart**
- **each functional group briefs their metrics in turn:**
  - **Milestone Chart first, followed by**
  - **Tracking Charts relevant for the review period**

The only chart in the stack that is not a milestone or tracking chart is the list of risks.

Why It Works – Some Psychology

Repetition

- **biweekly progress reviews with senior management**
- **metrics are always in the same format, so interpreting them becomes natural over time**

Simplicity and consistency

- **lines always indicate plans or projections**
- **bars always indicate actuals**
- **programs and groups are all measured in the same way**
- **consistency makes automation easy**

Graphical representation should match expectations

- **up is good**
- **filled-in is good**
- **straight across is good**

Focus on current data

- **keep milestone charts in view for past history and long-term outlook**
- **review tracking charts just for the work that is in progress or starting soon**

Conclusions from the Example

Simple, easy-to-read metrics can be an extremely powerful tool for monitoring and motivating projects.

The combination of the milestone and tracking charts

- **promotes effective team behaviors**
- **motivates teams to create reasonable plans and stick to them**

A robust metrics and senior management review process can be fully implemented in a very short period of time.

Earned Value Management (EVM)

- Define the work
- Schedule and budget
- Measure performance

Relates three independent variables

- *Planned Value (PV)* – estimated cost planned for an activity during a given period
- *Actual Cost (AC)* – costs incurred in accomplishing work on an activity during the given period
- *Earned Value (EV)* – value of the work completed

Variances and Performance Indices

Schedule Variance: SV = EV – PV

Cost Variance: CV = EV – AC

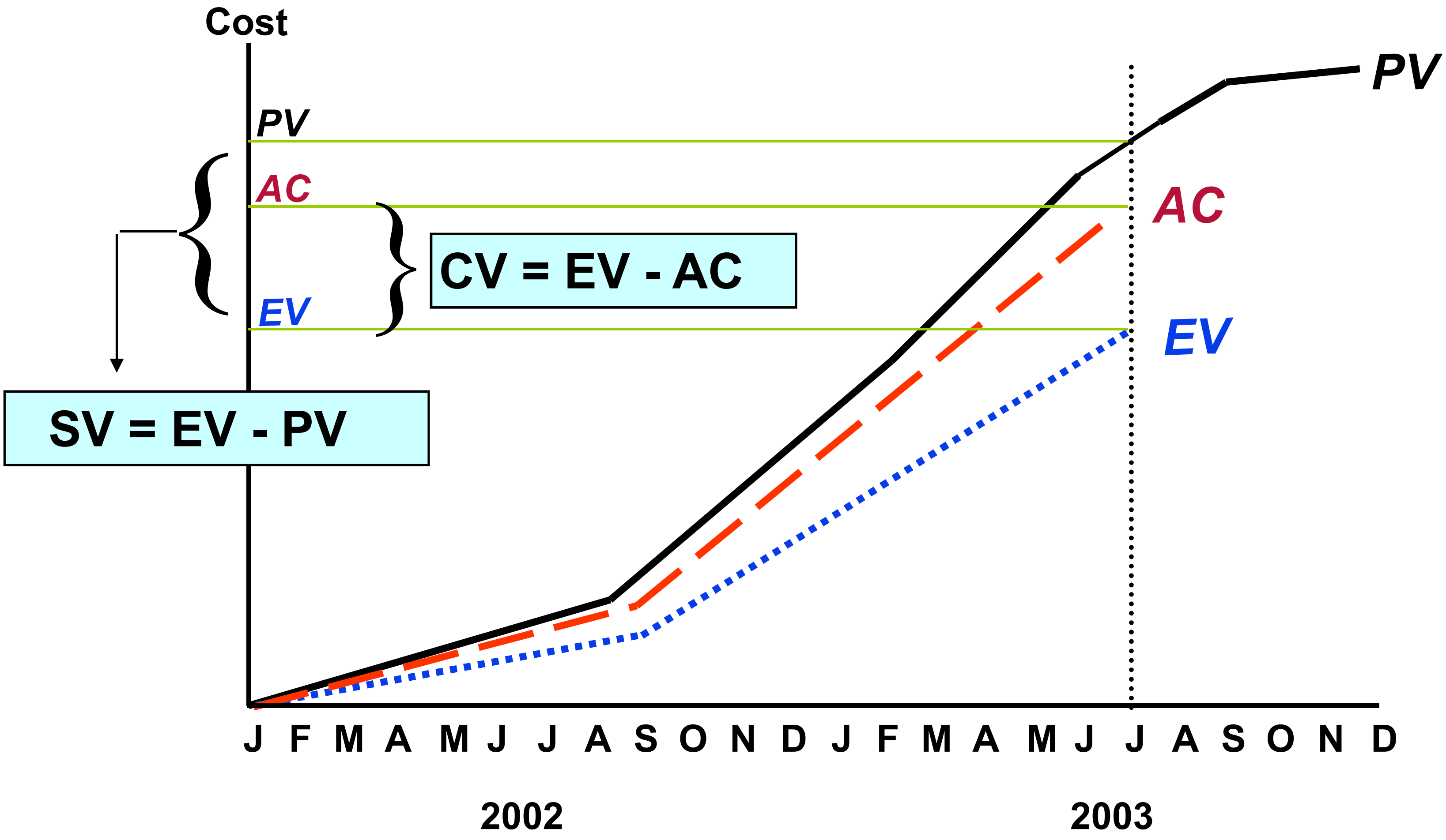
Negative SV and CV are over-runs.

Schedule Performance Index: SPI = EV / PV

Cost Performance Index: CPI = EV / AC

SPI and CPI less than 1 are over-runs.

Earned Value Analysis



CV = EV - AC

SV = EV - PV

Interpreting Performance Indices

Index	CPI (cost)	SPI (schedule)
>1	under cost	ahead of schedule
=1	on cost	on schedule
<1	over cost	behind schedule

EVM Example

Project budget of $300K

12 reporting periods

$97K allocated for first four periods

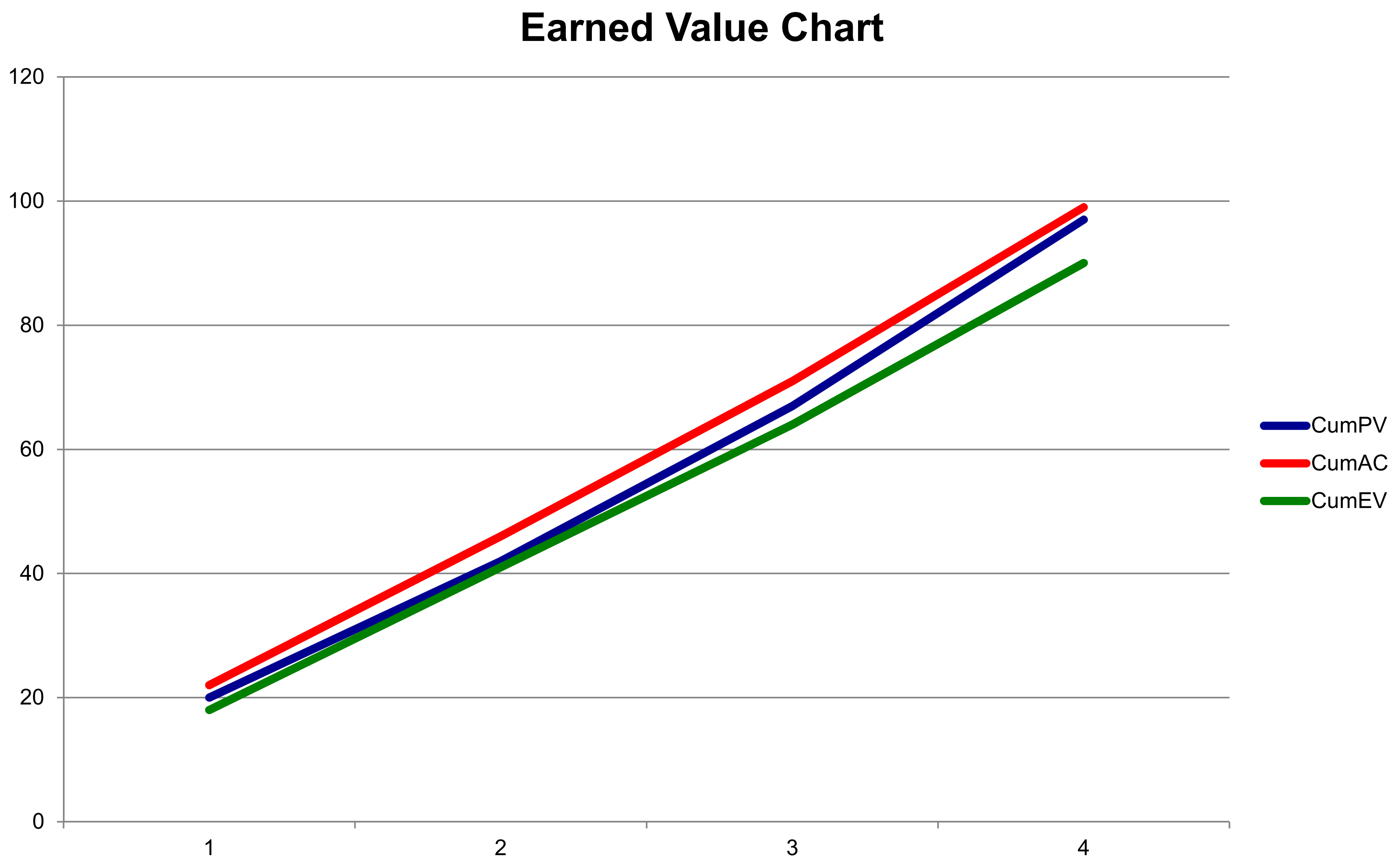
- now is end of period 4

By period	Period 1	Period 2	Period 3	Period 4
Work planned	20	22	25	30
Actual cost	22	24	25	28
Earned value	18	23	23	26

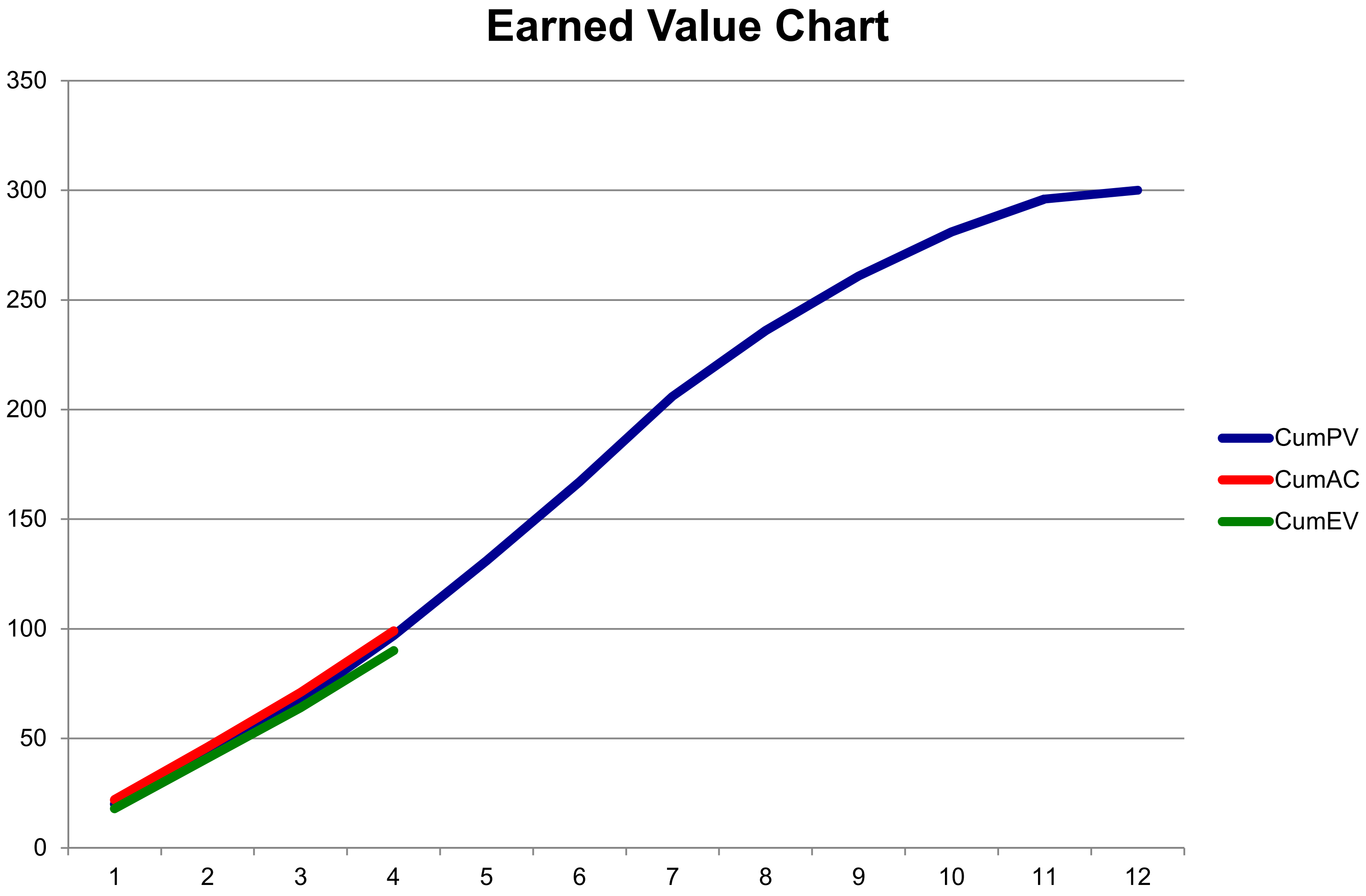
SV and CV
CPI and SPI

Cumulative	Period 1	Period 2	Period 3	Period 4
PV	20	42	67	97
AC	22	46	71	99
EV	18	41	64	90
SV =EV-PV	-2	-1	-3	-7
CV =EV-AC	-4	-5	-7	-9
SPI =EV/PV	18/20 =90%	41/42 =98%	64/67 =96%	90/97 =93%
CPI =EV/AC	18/22 =82%	41/46 =89%	64/71 =90%	90/99 =91%

Earned Value Chart (to now)



Earned Value Chart (to end)



An Earned Value Example

Project budget of $500K, 12 reporting periods

$300K allocated for first six periods

- **now is end of period 6**

By period	1	2	3	4	5	6
Planned value	10	20	30	30	40	50
Actual cost	8	12	32	40	48	53
Earned value	7	11	22	25	36	41

Cumulative PV, AC, EV

	1	2	3	4	5	6
PV	10	30	60	90	130	180
AC	8	20	52	92	140	193
EV	7	18	40	65	101	142

SV and CV

	1	2	3	4	5	6
PV	10	30	60	90	130	180
AC	8	20	52	92	140	193
EV	7	18	40	65	101	142
SV =EV-PV	-3	-12	-20	-25	-29	-38
CV =EV-AC	-1	-2	-12	-27	-39	-51

CPI and SPI

	1	2	3	4	5	6
PV	10	30	60	90	130	180
AC	8	20	52	92	140	193
EV	7	18	40	65	101	142
SV =EV-PV	-3	-12	-20	-25	-29	-38
CV =EV-AC	-1	-2	-12	-27	-39	-51
SPI =EV/PV	7/10 70%	18/30 60%	40/60 67%	65/90 72%	101/130 78%	142/180 79%
CPI =EV/AC	7/8 88%	18/20 90%	40/52 77%	65/92 71%	101/140 72%	142/193 74%

EVM Terminology

EV – Earned Value

- **aka BCWP budgeted cost of the work performed**

PV – Planned Value

- **aka BCWS budgeted cost of the work scheduled**

AC – Actual Cost

- **aka ACWP actual cost of the work performed**

BAC – Budgeted Cost at Completion

- total budgeted cost of the baseline or project cost accounts

EAC – Estimated Cost at Completion

ETC – Estimated Cost to Complete Remaining Work

VAC – Cost Variance at Completion

- indicates expected actual over- or under-run cost at completion

Measuring Progress

0/100% rule aka binary inch-pebbles

- either done or not done
- used for work packages with very short durations

50/50 rule aka ternary rule

- not started = 0%
- started = 50%
- done = 100%
- used for work packages of short duration and small total costs

Percent complete with weighted monitoring gates

- subjective estimated percent complete with hard, tangible monitoring points
- used on long-duration activities that can be broken into short, discrete work packages

What Does "Done" Mean?

For software modules

- **analysis and design completed**
- **design inspected**
- **coded**
- **code inspected**
- **unit tested**
- **integration tested**
- **regression tested**
- **documented**
  - **user, maintenance, deployment**
- **…**

For activities

- **"overhead"**
- **project management processes**
- **documentation support**
- **architectural design**
- **programming environment support**
- **test environment support**
- **…**

Does your budget (plan) include all of these activities?

Questions and Answers