SS ZG622: Software Project Management (Lecture #8)

T V Rao, BITS-Pilani Off-campus Centre, Hyderabad

BITS Pilani

Pilani|Dubai|Goa|Hyderabad

Text Books

5th edition
SOFTWARE PROJECT MANAGEMENT
Bob Hughes | Mike Cotterell | Rajib Mall

T1: Bob Hughes, Mike Cotterel, and Rajib Mall, Software Project Management, 5th Edition, McGraw Hill, 2011

T2: Pressman, R.S. Software Engineering : A Practitioner's Approach, 7th Edition, McGraw Hill, 2010

R1: Sommerville, I., Software Engineering, Pearson Education, 9th Ed., 2010

R2: Project Management: The Managerial Process by Erik W. Larson, Clifford F. Gray, McGraw Hill ©2011

R3: Capers Jones., Software Engineering Best Practices, TMH ©2010

R4: George Stepanek, Software Project Secrets : Why Software Projects Fail, Apress ©2012

R5: A Guide to the Project Management Body of Knowledge (PMBOK® Guide), Fifth Edition by Project Management Institute Project Management Institute © 2013

R6: Jake Kouns and Daniel Minoli, Information Technology Risk Management in Enterprise Environments. John Wiley & Sons © 2010

L8: Sotware Project Planning –

Project Monitoring & Control, Earned Value Analysis

Source Courtesy: Some of the contents of this PPT are sourced from materials provided by publishers of prescribed books

Software Project Planning (review)

- Software project planning encompasses five major activities
  - Estimation, scheduling, risk analysis, quality management planning, and change management planning
- Estimation determines how much money, effort, resources, and time it will take to build a specific system or product
- The software team first estimates
  - The work to be done
  - The resources required
  - The time that will elapse from start to finish
- Then they establish a project schedule that
  - Defines tasks and milestones
  - Identifies who is responsible for conducting each task
  - Specifies the inter-task dependencies

Estimation/Scheduling problems (review)

- Estimating the difficulty of problems and hence the cost of developing a solution is hard.
- Productivity is not proportional to the number of people working on a task.
- Adding people to a late project makes it later because of communication overheads.
- The unexpected always happens. Always allow contingency in planning.

– Sommerville

Basic Principles for Project Scheduling (review)

- Compartmentalization
  - The project must be compartmentalized into a number of manageable activities, actions, and tasks; both the product and the process are decomposed
- Interdependency
  - The interdependency of each compartmentalized activity, action, or task must be determined
  - Some tasks must occur in sequence while others can occur in parallel
  - Some actions or activities cannot commence until the work product produced by another is available
- Time allocation
  - Each task to be scheduled must be allocated some number of work units
  - In addition, each task must be assigned a start date and a completion date that are a function of the interdependencies
  - Start and stop dates are also established based on whether work will be conducted on a full-time or part-time basis

(More on next slide)

Basic Principles for Project Scheduling (continued)

- Effort validation
  - Every project has a defined number of people on the team
  - As time allocation occurs, the project manager must ensure that no more than the allocated number of people have been scheduled at any given time
- Defined responsibilities
  - Every task that is scheduled should be assigned to a specific team member
- Defined outcomes
  - Every task that is scheduled should have a defined outcome for software projects such as a work product or part of a work product
  - Work products are often combined in deliverables
- Defined milestones
  - Every task or group of tasks should be associated with a project milestone
  - A milestone is accomplished when one or more work products has been reviewed for quality and has been approved

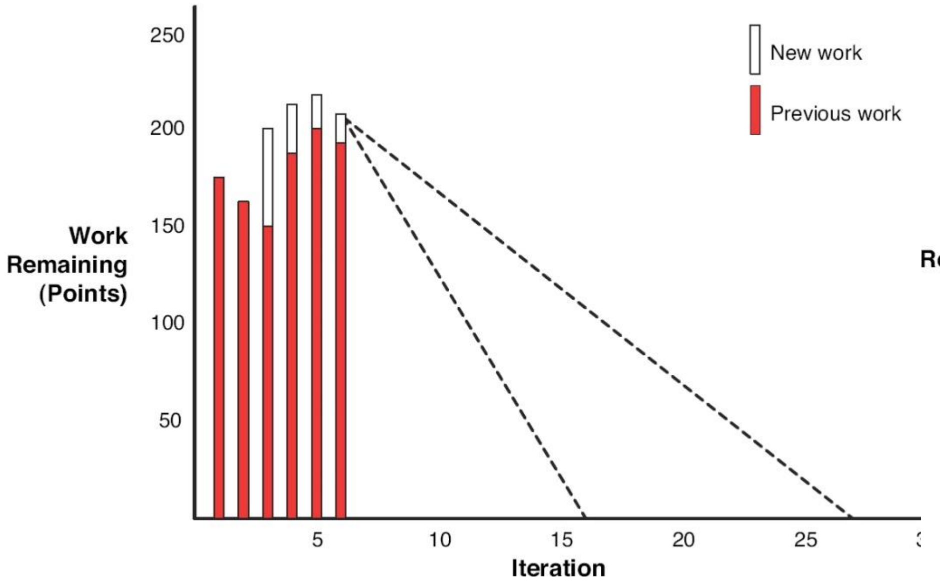
Distribution of Effort (review)

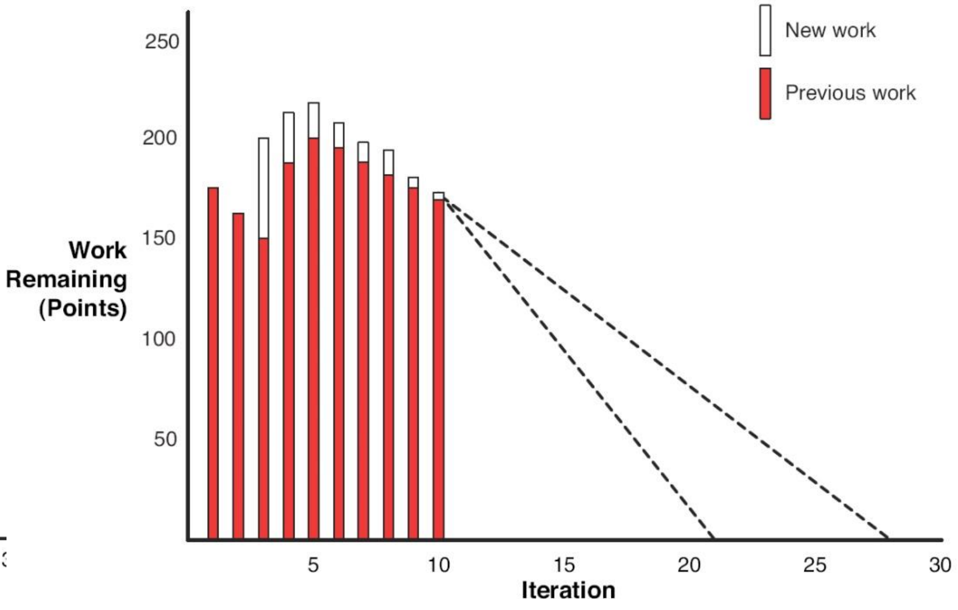
- A recommended distribution of effort across the software process is 40% (analysis and design), 20% (coding), and 40% (testing)
- Work expended on project planning rarely accounts for more than 2 - 3% of the total effort
- Requirements analysis may comprise 10 - 25%
  - Effort spent on prototyping and project complexity may increase this
- Software design normally needs 20 – 25%
- Coding should need only 15 - 20% based on the effort applied to software design
- Testing and subsequent debugging can account for 30 - 40%
  - Safety or security-related software requires more time for testing

Time Boxing (review)

- Severe deadline pressure may require the use of time boxing
  - An incremental software process is applied to the project
  - The tasks associated with each increment are "time-boxed" (i.e., given a specific start and stop time) by working backward from the delivery date
  - The project is not allowed to get "stuck" on a task
  - When the work on a task hits the stop time of its box, then work ceases on that task and the next task begins
  - This approach succeeds based on the premise that when the time-box boundary is encountered, it is likely that 90% of the work is complete
  - The remaining 10% of the work can be
    - Delayed until the next increment
    - Completed later if required
- Time-boxing is commonly used in agile processes

Convergence in Burndown Chart (review)





Gross velocity is the best-case situation and overly optimistic.
Net velocity is the worst-case scenario and overly pessimistic.
Estimates for completion should converge over iterations

Time to Market

The Software Equation (review)

A dynamic multivariable model

$$E = [LOC \times B^{0.333}/P]^3 \times (1/t^4)$$

[A 10% compression in timeline(t) leads to 52% increase in Effort/Cost(E)]

E= effort in person-months or person-years

t = project duration in months or years

B = “special skills factor”

– for small programs(5 to 15 KLOC), B=0.16

– for large programs(> KLOC), B=0.39

P = productivity parameter reflecting overall process maturity, skills, experience etc. Needs to be customized for local conditions.

– for real-time embedded software P=2000

– for system software P=10000

– for business applications P=28000

- Putnam et al.

Effort Applied vs. Delivery Time

- There is a nonlinear relationship between effort applied and delivery time (Ref: Putnam-Norden-Rayleigh software equation)
  - Effort increases rapidly as the delivery time is reduced
- Also, delaying (within limits) project delivery can reduce costs significantly as shown in the equation **$E \propto L^3/(P^3 t^4)$**
  - E = development effort in person-months
  - L = source lines of code delivered
  - P = productivity parameter
  - t = project duration in calendar months

Reducing Project Duration

- Time Is Money: Cost-Time Tradeoffs
  - Reducing the time of a critical activity usually incurs additional direct costs.
    - Cost-time solutions focus on reducing (crashing) activities on the critical path to shorten overall duration of the project.
  - Reasons for imposed project duration dates:
    - Time-to-market pressures
    - Unforeseen delays
    - Incentive contracts (bonuses for early completion)
    - Imposed deadlines and contract commitments
    - Overhead and public goodwill costs
    - Pressure to move resources to other projects

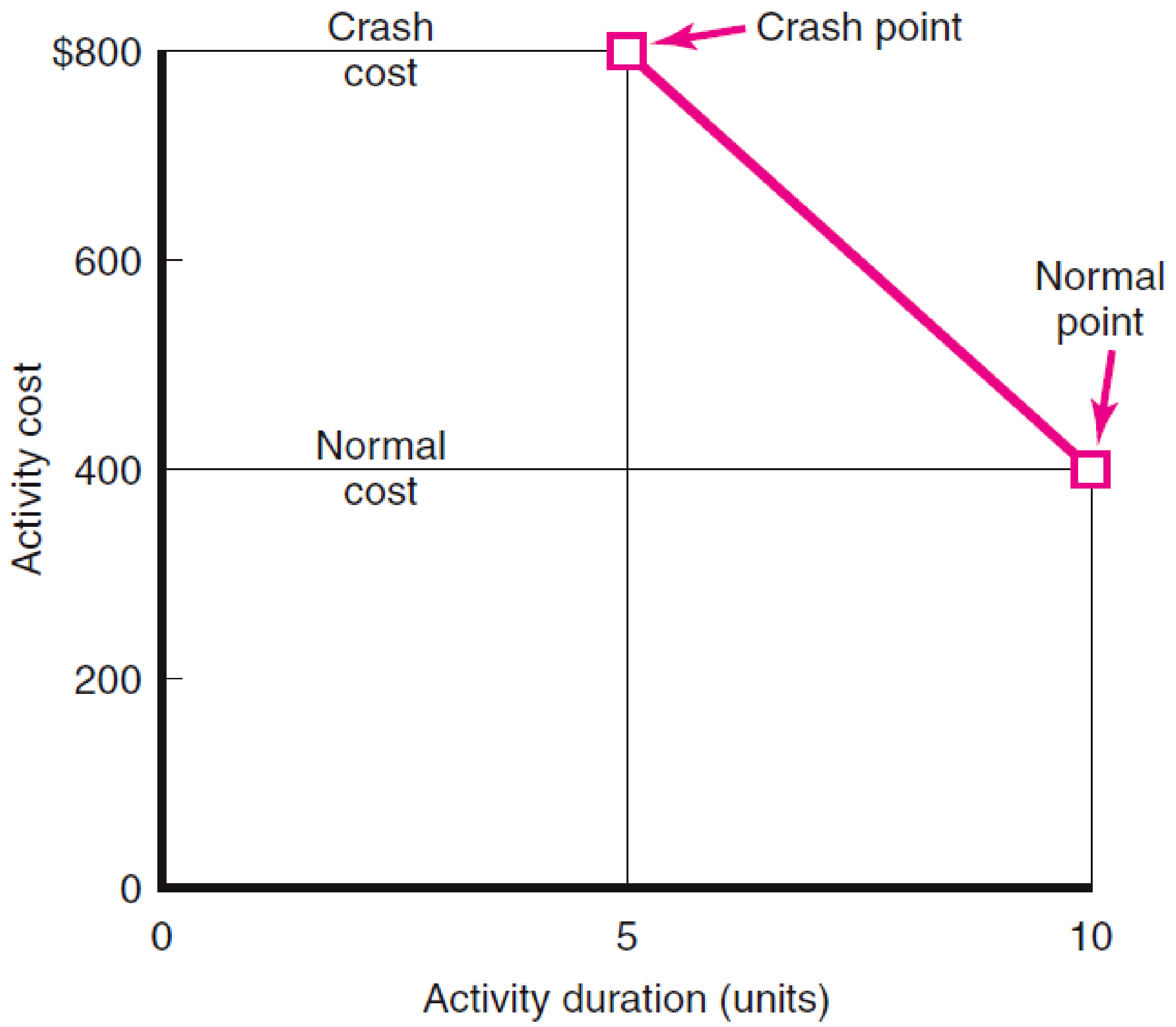
Options for Accelerating Project

- Resources *Not* Constrained
  - Adding resources
  - Outsourcing project work
  - Scheduling overtime
  - Establishing a core project team
  - Do it twice—fast and then correctly
- Resources Constrained
  - Fast-tracking
  - Critical-chain (Eliyahu Goldratt's technique)
  - Reducing project scope
  - Compromise quality

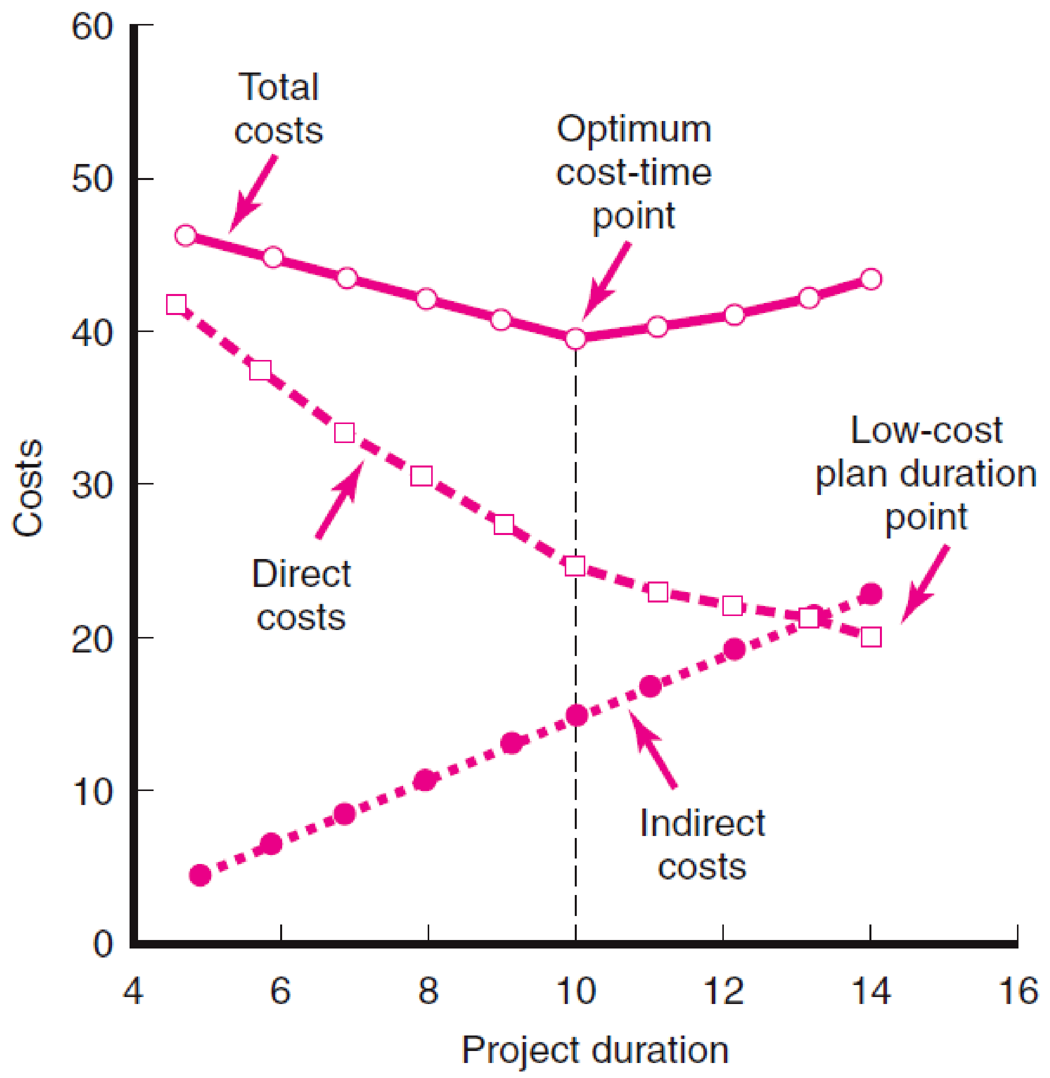
Project Costs

- Project Indirect Costs
  - Costs that cannot be associated with any particular work package or project activity.
    - Supervision, administration, consultants, and interest
  - Costs that vary (increase) with time.
    - Reducing project time directly reduces indirect costs.
- Project Direct Costs
  - Normal costs that can be assigned directly to a specific work package or project activity.
    - Labor, materials, equipment, and subcontractors
  - Crashing activities increases direct costs.

Activity Cost Graph

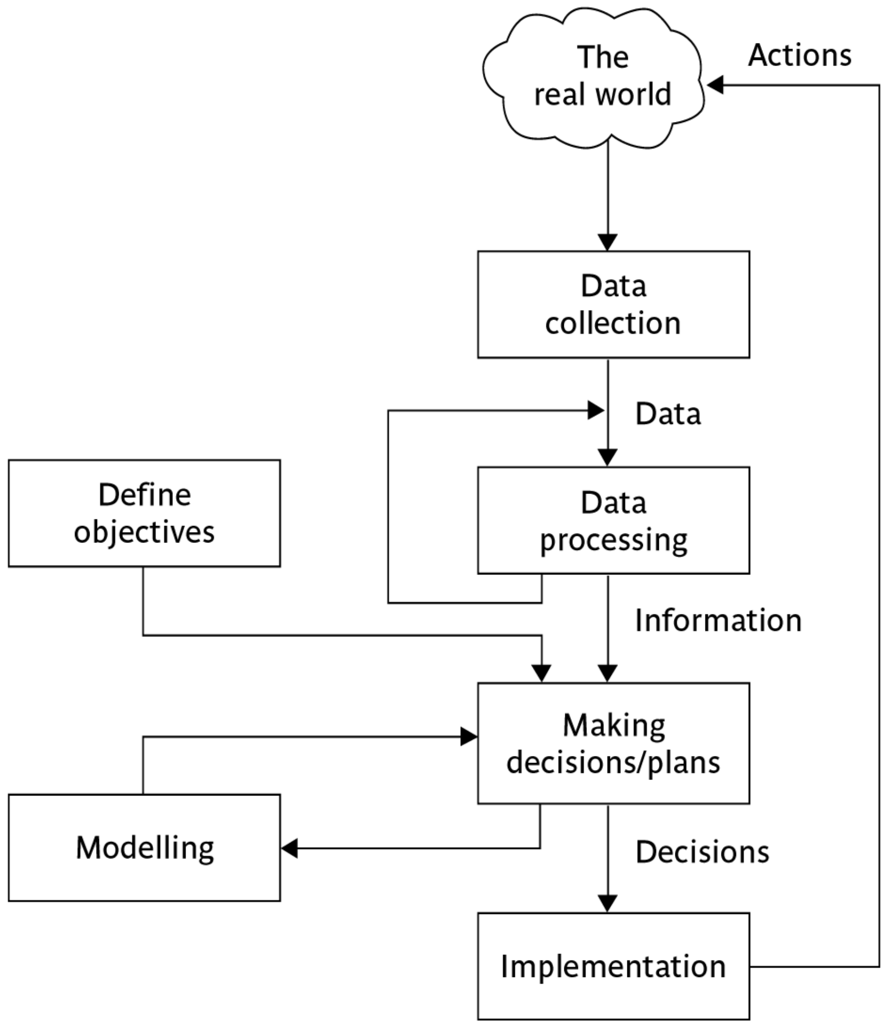


Project Cost–Duration Graph

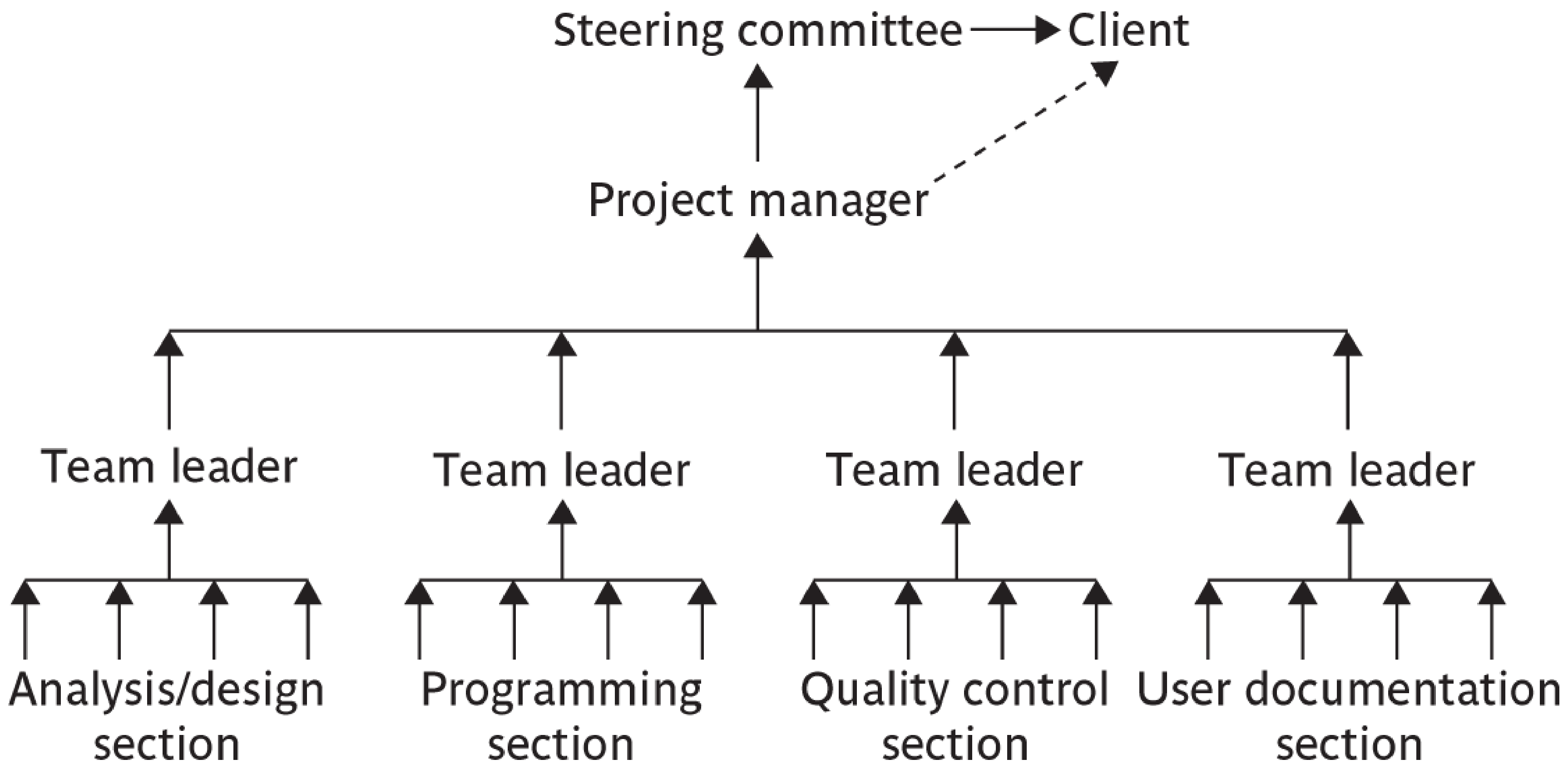


Monitoring & Control

Control Cycle



Software Project Reporting Lines (medium to large)



Progress Assessment

Checkpoints – predetermined times when progress is checked

- Event driven: check takes place when a particular event has been achieved
- Time driven: date of the check is pre-determined

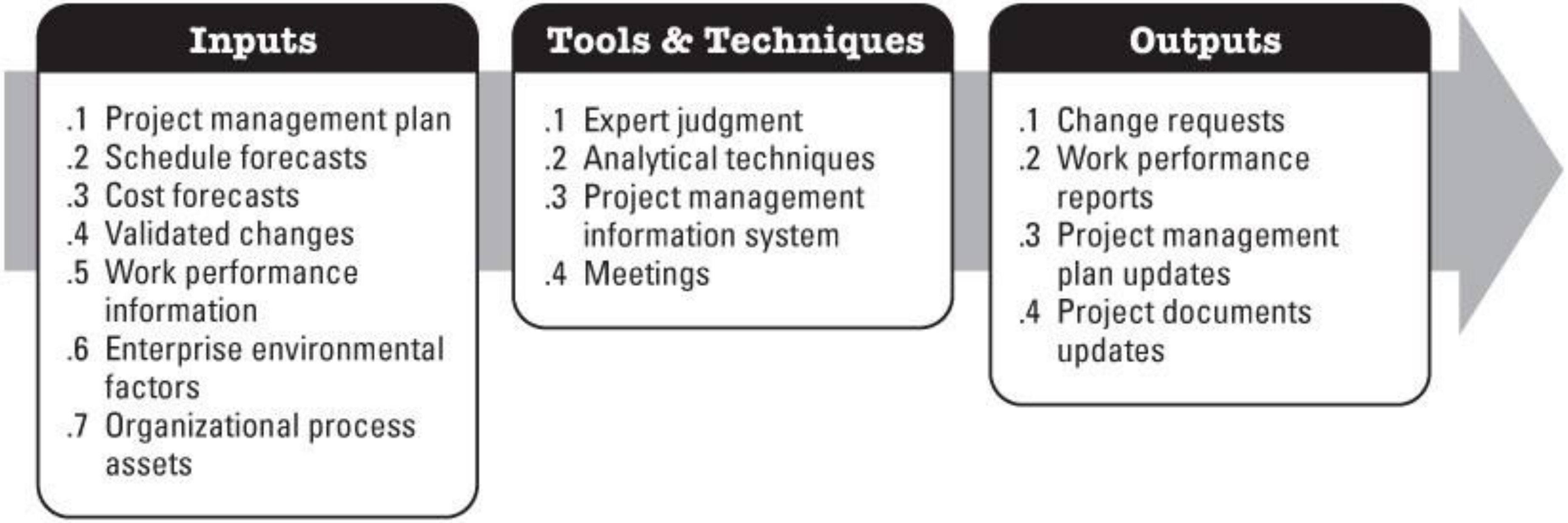
Frequency of reporting

The higher the management level then generally the longer the gaps between checkpoints

Project Control

- The project manager applies control to administer project resources, cope with problems, and direct project staff
- If things are going well (i.e., schedule, budget, progress, milestones) then control should be light
- When problems occur, the project manager must apply tight control to reconcile the problems as quickly as possible. For example:
  - Staff may be redeployed
  - The project schedule may be redefined

Monitoring & Control (PMI view)



Inputs

.1 Project management plan
.2 Schedule forecasts
.3 Cost forecasts
.4 Validated changes
.5 Work performance information
.6 Enterprise environmental factors
.7 Organizational process assets

Tools & Techniques

.1 Expert judgment
.2 Analytical techniques
.3 Project management information system
.4 Meetings

Outputs

.1 Change requests
.2 Work performance reports
.3 Project management plan updates
.4 Project documents updates

Monitoring & Control (PMI view)

The Monitor and Control Project Work process is concerned with

- Comparing actual project performance against the project management plan
- Assessing performance to determine whether any corrective or preventive actions are indicated, and then recommending those actions as necessary
- Identifying new risks and analyzing, tracking, and monitoring existing project risks to make sure the risks are identified, their status is reported, and that appropriate risk response plans are being executed
- Maintaining an accurate, timely information base concerning the project's product(s) and their associated documentation through project completion
- Providing information to support status reporting, progress measurement, and forecasting
- Providing forecasts to update current cost and current schedule information
- Monitoring implementation of approved changes as they occur
- Providing appropriate reporting on project progress and status to program management when the project is part of an overall program

Prioritizing monitoring

We might focus more on monitoring certain types of activity e.g.

- Critical path activities
- Activities with no free float – if delayed later dependent activities are delayed
- Activities with less than a specified float
- High risk activities
- Activities using critical resources

How to get back on track?

- Renegotiate the deadline – if not possible then
- Try to shorten critical path e.g.
  - Work overtime
  - Re-allocate staff from less pressing work
  - Buy in more staff
- Reconsider activity dependencies
  - Over-lap the activities so that the start of one activity does not have to wait for completion of another
  - Split activities

Exception planning

- Some changes could affect
  - Users
  - The business case (e.g. costs increase reducing the potential profits of delivered software product)
- These changes could be to
  - Delivery date
  - Scope
  - Cost
- In these cases an **exception report** is needed

Exception planning - continued

- First stage
  - Write an **exception report** for sponsors (perhaps through project board)
    - Explaining problems
    - Setting out options for resolution
- Second stage
  - Sponsor selects an option (or identifies another option)
  - Project manager produces an **exception plan** implementing selected option
  - Exception plan is reviewed and accepted/rejected by sponsors/Project Board

Methods for Tracking the Schedule

- Qualitative approaches
  - Conduct periodic project status meetings in which each team member reports progress and problems
  - Evaluate the results of all reviews conducted throughout the software engineering process
  - Determine whether formal project milestones (i.e., diamonds) have been accomplished by the scheduled date
  - Compare actual start date to planned start date for each project task listed in the timeline chart
  - Meet informally with the software engineering team to obtain their subjective assessment of progress to date and problems on the horizon
- Quantitative approach
  - Use earned value analysis to assess progress quantitatively

"The basic rule of software status reporting can be summarized in a single phrase: No surprises." Capers Jones

Earned Value Analysis

How is project health measured?

- Project Manager evaluates project's triple constraint of scope, time and cost
- Key Questions
  - Is the project performing to budget?
  - Is the project on schedule to deliver the agreed scope?
- Typically Summarize Project Health using Green, Yellow, Red Status (Traffic Light Reporting)

Problems with the Traffic Light status approach

- Subjective to interpretation and influence
- No objective measurement to guide Project Health
- A project can report green and suddenly turn red before a few days before the launch
- A project can report red and be rationalized to green or yellow without any objective data
- No prior indicators to problems

Earned Value Analysis

- Earned Value Analysis (EVA)
  - Earned Value Analysis is an objective method to measure project performance in terms of scope, time and cost
  - Use EVA metrics are used to measure project health and project performance

- EVA tidbits
  - EVA used by US DOD. In 1991, Secretary of Defense canceled the Navy A-12 Avenger II Program because of performance problems detected by EVA. It was adopted by the National Aeronautics and Space Administration, United States Department of Energy and other technology-related agencies, and also many industrialized nations. (Wikipedia)

Earned Value Characteristics

- Point in Time Evaluation
- How much work did you PLAN to complete? (*Planned Value*)
- How much work did you ACTUALLY complete? (*Earned Value*)
- How much did you spend to complete the work? (*Actual Cost*)

Description of Earned Value Analysis

- Earned value analysis is a measure of progress by assessing the percent of completeness for a project
- It gives accurate and reliable readings of performance very early into a project
- It provides a common value scale (e.g., time) for every project task, regardless of the type of work being performed
- The total hours to do the whole project are estimated, and every task is given an earned value based on its estimated percentage of the total

Accounting conventions

- Work completed allocated on the basis
  - *50/50* half allocated at start, the other half on completion. These proportions can vary e.g. 75/25 etc
  - 0/100 allocate only after completion (safe option for software)
  - *Milestone* current value depends on the milestones achieved
  - *Units (LOC) completed*
- Can use money values, or staff effort as a surrogate

Determining Earned Value

- Compute the budgeted cost of work scheduled (BCWS) for each work task *i* in the schedule
  - The BCWS is the effort planned; work is estimated in person-hours or person-days for each task
  - To determine progress at a given point along the project schedule, the value of BCWS is the sum of the BCWS$_i$ values of all the work tasks that should have been completed by that point of time in the project schedule
  - BCWS is also referred as **PV** (Planned Value)
- Sum up the BCWS values for all work tasks to derive the budget at completion (BAC)

$$BAC = \sum (BCWS_k) \text{ for all tasks } k$$

Computing Earned Value-II

- Next, the value for *budgeted cost of work performed* (BCWP) is computed.
  - The value for BCWP is the sum of the BCWS values for all work tasks that have actually been completed by a point in time on the project schedule. Referred by many authors as **EV** (Earned Value)
- “the distinction between the BCWS and the BCWP is that the former represents the budget of the activities that were planned to be completed and the latter represents the budget of the activities that actually were completed.”
- Given values for BCWS, BAC, and BCWP, important progress indicators can be computed:
  - Schedule performance index, SPI = BCWP/BCWS
  - Schedule variance, SV = BCWP – BCWS
  - SPI is an indication of the efficiency with which the project is utilizing scheduled resources.

Computing Earned Value-III

- Percent scheduled for completion = BCWS/BAC
  - provides an indication of the percentage of work that should have been completed by time *t.*
- Percent complete = BCWP/BAC
  - provides a quantitative indication of the percent of completeness of the project at a given point in time, *t.*
- *Actual cost of work performed,* ACWP, is the sum of the effort actually expended on work tasks that have been completed by a point in time on the project schedule. It is then possible to compute
  - Cost performance index, CPI = BCWP/ACWP
  - Cost variance, CV = BCWP – ACWP
  - ACWP also referred as **AC** (Actual Cost)

Progress Indicators in Earned Value Analysis

- SPI = BCWP/BCWS
  - Schedule performance index (SPI) is an indication of the efficiency with which the project is utilizing scheduled resources
  - SPI close to 1.0 indicates efficient execution of the project schedule
- SV = BCWP – BCWS
  - Schedule variance (SV) is an absolute indication of variance from the planned schedule
- PSFC = BCWS/BAC
  - Percent scheduled for completion (PSFC) provides an indication of the percentage of work that <u>should have been completed</u> by time t
- PC = BCWP/BAC
  - Percent complete (PC) provides a quantitative indication of the percent of work that <u>has been completed</u> at a given point in time t
- ACWP = sum of actual costs for completed tasks as of time t
  - Actual cost of work performed (ACWP) includes all tasks that have been completed by a point in time t on the project schedule
- CPI = BCWP/ACWP
  - A cost performance index (CPI) close to 1.0 provides a strong indication that the project is within its defined budget
- CV = BCWP – ACWP
  - The cost variance is an absolute indication of cost savings (against planned costs) or shortfall at a particular stage of a project

EVA Example

A $10,000 software project is scheduled for 4 weeks.

At the end of the third week, the project is 50% complete and the actual costs to date is $9,000

Planned Value (PV) = $7,500

Earned Value (EV) = $5,000

Actual Cost (AC) = $9,000

What is the project health?

Schedule Variance

= EV – PV = $5,000 – $7,500 = - $2,500

Schedule Performance Index (SPI)

= EV/PV = $5,000 / $7,500 = .66

Cost Variance

= EV – AC = $5,000 - $9,000 = - $4,000

Cost Performance Index (CPI)

= EV/AC = $5,000 / $9,000 = .55

Objective metrics indicate the project is behind schedule and over budget.

On-target projects have an SPI and CPI of 1 or greater

Forecasting Costs

- If the project continues at the current performance, what is the true cost of the project?
- Estimate At Complete

  = Budget At Complete (BAC) / CPI

  = $10,000 / .55 = $18,181

  At the end of the project, the total project costs will be $18,181

Earned value for Software Project

- Tasks

– Specify module	5 days
– Code module	8 days
– Test module	6 days

- At the beginning of day 20, PV = 19 days
- If everything but testing completed EV = 13 days
- Schedule variance = EV-PV i.e. 13-19 = -6
- Schedule performance indicator (SPI) = 13/19 = 0.68
- SV negative or SPI <1.00, project behind schedule

Actual Cost

- Actual cost (AC) is also known as Actual cost of work performed (ACWP)
- In previous example, if
  - ‘Specify module’ actually took 3 days
  - ‘Code module’ actually took 4 days
- Actual cost = 7 days
- Cost variance (CV) = EV-AC i.e. 13-7 = 6 days
- Cost performance indicator = 13/7 = 1.86
- Positive CV or CPI > 1.00 means project within budget

Projected costs

- CPI can be used to produce new cost estimate
- Budget at completion (BAC) – current budget allocated to total costs of project
- Estimate at completion (EAC) – updated estimate = BAC/CPI
  - e.g. say budget at completion is £19,000 and CPI is 1.86
  - EAC = BAC/CPI = £10,215 (projected costs reduced because work being completed in less time)

- What about projecting schedule?
- How to report schedule underperformance at end of project?
  - PMI is considering incorporating Earned Schedule on lines of Earned Value

Establish Ranges to Guide Traffic Light Status

- Traffic Light status is useful in conveying overall project with one color
- Establish objective SPI and CPI ranges to determine the true project color.

Green	[1.0 - .95]
Yellow	[.94-.85]
Red	[.84, 0]

Earned Value Summary

- Earned Value is an objective method of determining project performance instead of subjective approaches
- Apply Earned Value enforces the project discipline of tracking project actual performance against baseline costs and dates
- Estimate at Complete calculation can forecast true project costs based on project performance

Review

Session 1

- Software – its characteristics
- Software Engineering Definitions
- Layered view of Software Engineering
- Framework & Umbrella activities
- Software Myths
- Process Patterns

Session 2

- Software processes are adapted to meet the needs of software engineers and managers as they undertake the development of a software product.
- A software process provides a framework for managing activities that can very easily get out of control.
- Different types of projects require different software processes.
- Plan-driven development vs. Agile development
- The software engineer's work products (programs, documentation, data) are produced as consequences of the activities defined by the software process.
- The best indicators of how well a software process has worked are the quality, timeliness, and long-term viability of the resulting software product.

Session 3

- Project Definition
- Project Characteristics
- Software Project Characteristics
- Project Activities vs. Management Activities
- PMI View & Nomenclature
- Project Objectives(SMART) vs. Business Case
- Success & Failure of Software Projects
- Declaration of Interdependence
- GROW model of Leadership
- Goals that are Realistic arrived by considering Options and backed by Will(commitment)

Session 4

- Program definition
- Project vs. Program vs. Portfolio Management
- COBIT5 for Risk Reduction with IT investments
- Cost-Benefit Analysis helps in choice of projects in a program
  - Net Profit
  - Payback period
  - Return on Investment
  - Net Present Value
  - Internal Rate of Return
- Decision trees can help in handling uncertainties that can be quantified

Session 5

- Measures, Metrics, Indicators definitions
- Metrics attributes, measurement etiquette
- Product metric taxonomy
  - Analysis
  - Design
  - Coding
  - Testing
  - Maintenance
- Metrics for Process, Project
- Metrics for Agile development
- Helpful vs. Harmful metrics

Session 6

- Estimation concepts, techniques
- Decomposition Techniques
  - Product-based
  - Process-based
- Empirical Models
  - COCOMO2
    - Application composition, Early design, Reuse, Post-architectural
    - Duration estimation
  - Software Equation
  - Lorenz-Kidd approach for OO software
- Agile Techniques

Session 7

- Project Scheduling Practices, Principles
- Software Project effort distribution
- Software project activity networks
- Activity Planning
- Precedence Diagram
  - Project duration
  - Activity floats, Free float, Interfering float
- Agile burndown charts
- Risk Management
  - Risk Projection
  - Risk Mitigation, Monitoring, and Management Plan (RMMM)

Session 8

- Reducing project/activity durations
- Impact of crashing on direct/indirect costs
- Software project reporting lines
- Project monitoring and control
- Exception Planning
- Earned Value Analysis

Thank You

Any Questions?