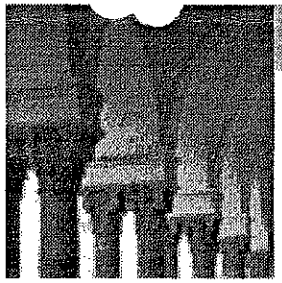
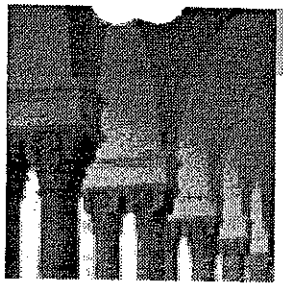


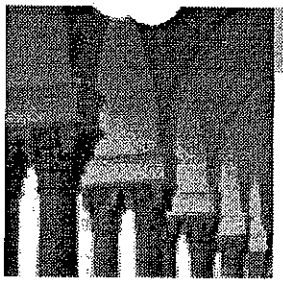
Objectives

- Understand the purpose of the project kick-off meeting
- Identify the key planning documents recommended for a software project
- Understand the different methods used to estimate software projects



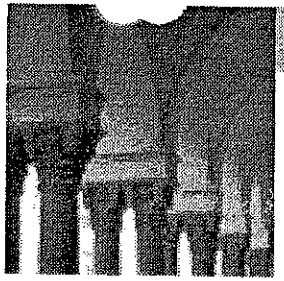
Software Project Planning

- The Software Engineering Institute states
  - The purpose of software project planning is to establish reasonable plans for performing the software engineering and managing the software project
- Software project planning involves
  - Defining a plan to perform the work
  - Establishing commitments to do the work
  - Developing estimates for the work to be performed



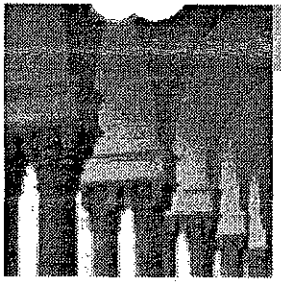
Project Kick-off Meeting

- The project kick-off meeting provides the forum to obtain buy-in to the project plan / SDP
- This meeting communicates
  - Project objectives
  - Roles and responsibilities
  - Project schedule and individual task assignments
  - Project technical administration
  - Project timekeeping and reporting



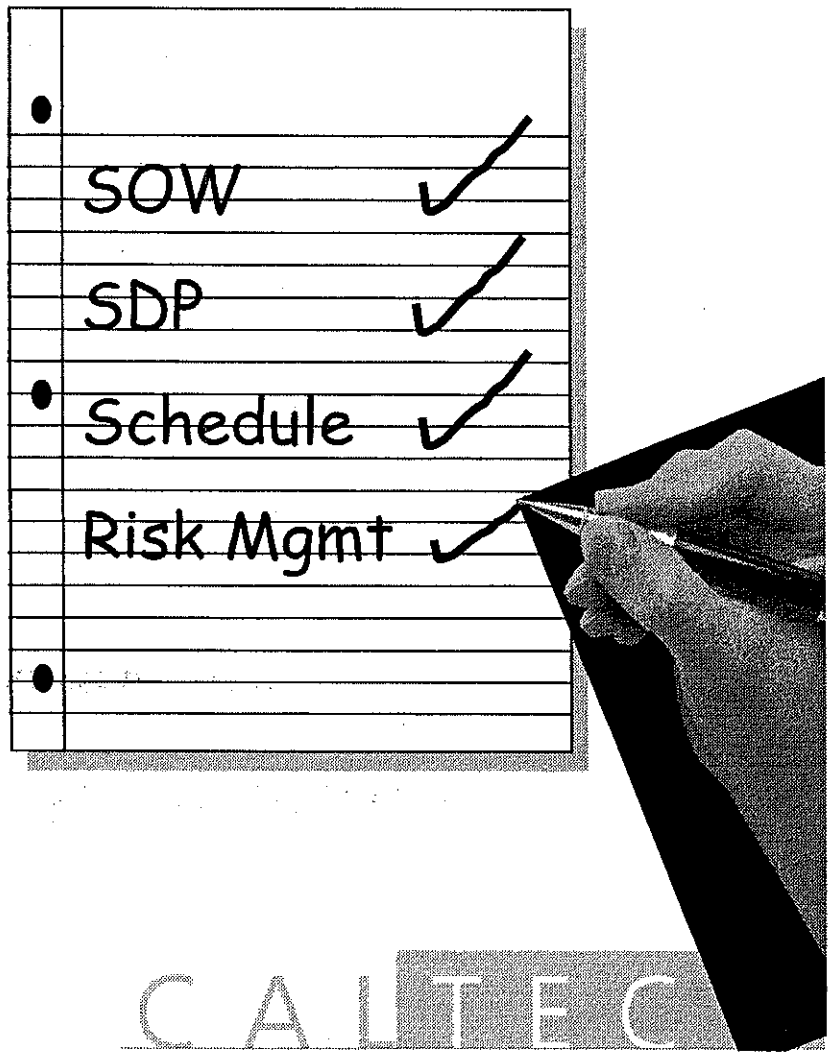
Project Kick-off Meeting

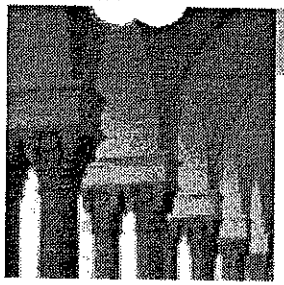
- The project kickoff meeting introduces project team members to the project organization and management
- Audience
  - Project Manager
  - Project Technical staff
  - Client Project Manager
  - Managers of any affected groups/support groups



Deliverables - Revisited

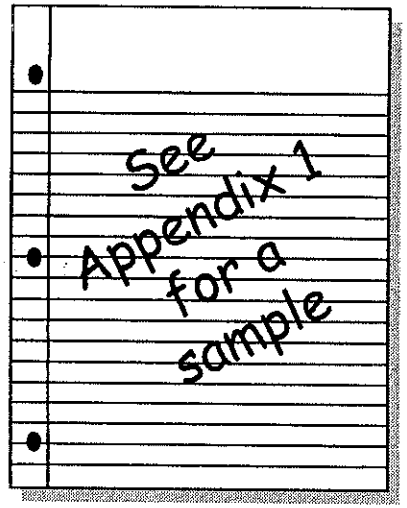
- The Deliverables List is a checklist for planning
  - Statement of Work (SOW)
  - Software Development Plan
  - Project Schedule and Estimates
  - Risk Management Plan
  - Software Configuration Management Plan
  - Verification and Validation Plan
  - Maintenance Plan

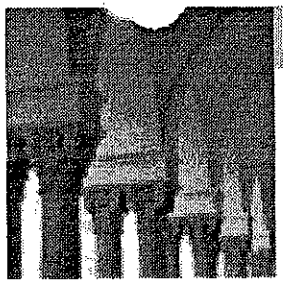




Statement of Work (SOW)

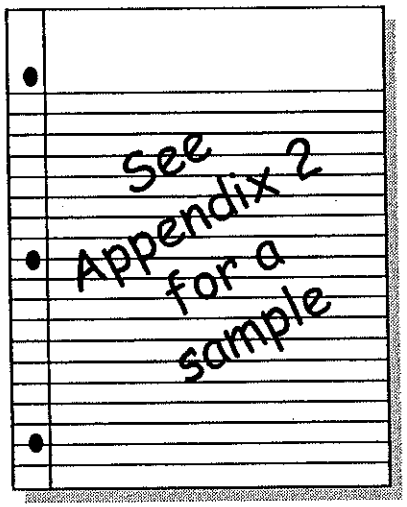
- Documents the scope of a proposed project
- Establishes the initial commitment to do the project
- Provides a conceptual estimate (± 50%)
- Prepared using business language
- Is **NOT** a **low-level** definition of the project

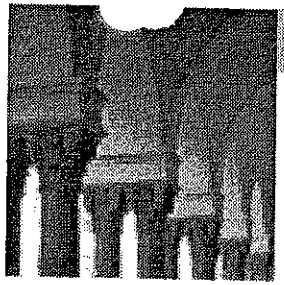




Software Development Plan (SDP)

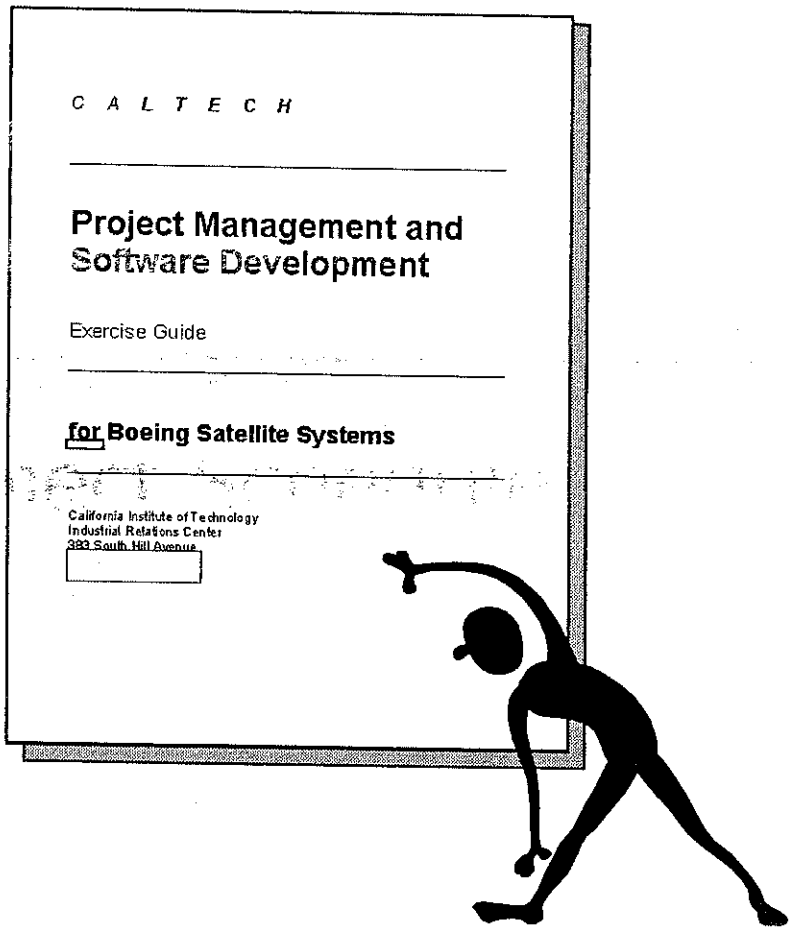
- Key management document for a software project which details
  - How the project will be run
  - What will be delivered
  - What kind of resources are needed
  - Estimates for size, duration, effort and cost
  - How risk will be managed
- Timing depends on project size and complexity
  - At a minimum, upon completion of requirements
  - By project phase
  - When major scope change occurs (outside of ± 25% of estimates)

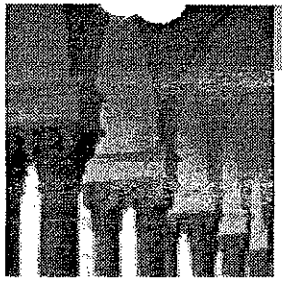




Exercise

- 4.1 – Developing a Plan

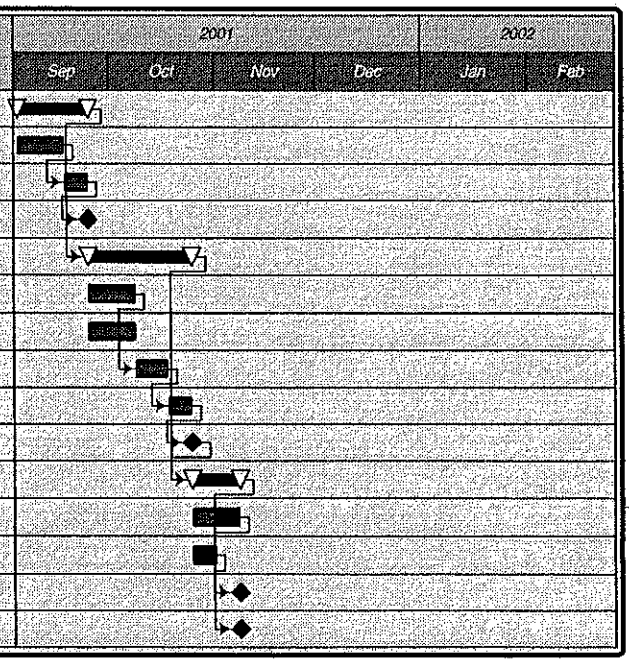


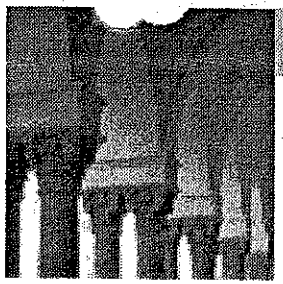


Software Project Schedule

- Delivered with SOW
  - High-level milestones with limited precision
- Delivered with SDP
  - Baseline, tracking and working version

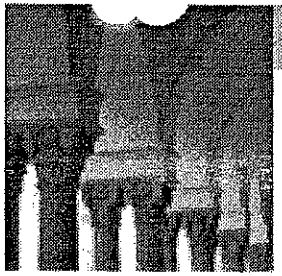
ID	Task Name	Start Date	End Date	Duration	Actual Start	Actual End	Percent Complete	Actual Hours
1	**Task 1**	**9/4/2001**	**9/25/2001**	**15d**	**9/4/2001**	**9/24/2001**	**0.00%**	
2	Subtask 1-1	9/4/2001	9/17/2001	80h	9/4/2001	9/17/2001	100.00%	75 hr
3	Subtask 1-2	9/18/2001	9/24/2001	40h	9/18/2001	9/24/2001	100.00%	40h
4	Task 1 complete	9/25/2001	9/25/2001	0d	9/25/2001	9/25/2001	100.00%	
5	**Task 2**	**9/25/2001**	**10/25/2001**	**22d 4h**	**9/25/2001**	**10/25/2001**	**0.00%**	
6	Subtask 2-1	9/25/2001	10/8/2001	80h	9/26/2001	10/9/2001	100.00%	80h
7	Subtask 2-2	9/25/2001	10/8/2001	80h	9/25/2001	10/10/2001	100.00%	88h
8	Subtask 2-3	10/9/2001	10/18/2001	60h	10/9/2001	10/18/2001	100.00%	45h
9	Subtask 2-4	10/18/2001	10/25/2001	40h	10/18/2001	10/25/2001	100.00%	40h
10	Task 2 complete	10/25/2001	10/25/2001	0d	10/25/2001	10/25/2001	100.00%	
11	**Task 3**	**10/25/2001**	**11/8/2001**	**10d**	**10/25/2001**	**11/8/2001**	**0.00%**	
12	Subtask 3-1	10/25/2001	11/8/2001	80h	10/25/2001	11/8/2001	50.00%	60h
13	Subtask 3-2	10/25/2001	11/1/2001	40h	10/25/2001	11/1/2001	25.00%	24h
14	Task 3 Complete	11/8/2001	11/8/2001	0d	11/8/2001	11/8/2001	0.00%	
15	Project Complete	11/8/2001	11/8/2001	0d	11/8/2001	11/8/2001	0.00%	





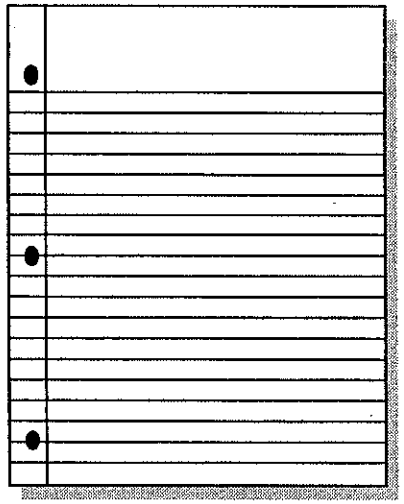
Schedule Basics

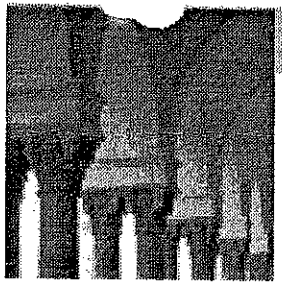
- Use a task-based WBS
- Establish milestones for deliverables and critical tasks
- Break tasks into manageable chunks
  - Load tasks with estimated hours
  - 80 man-hours or less
  - Resource loaded, not-to-exceed 80%
- Get commitments



What to Estimate

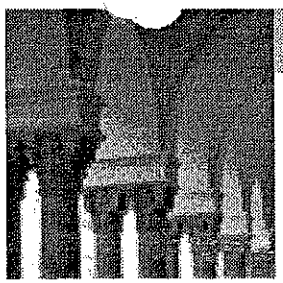
- Software size
  - Software lines of code (SLOC) or thousand lines of code (KLOC)
  - Feature or function points
- Duration for completion of all project tasks
- Resources
  - Staff and critical computer resources
  - Facilities
- Effort (costs and person-hours for each task)
  - Costs for each task
  - Man-hours for each task





When to Estimate

- Target accuracy varies according to the timing of the estimate
  - SOW based (± 50%)
  - Requirements based (± 25%)
  - Design based (± 10%)
- Revise estimates when
  - A major scope change occurs
  - Project risk event tripwire is hit

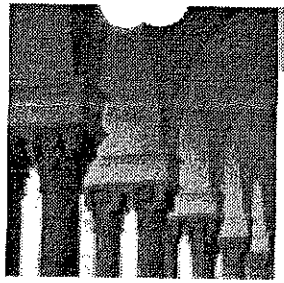


Estimating Challenges for Software

- History of estimates and actuals for similar projects may not exist
- Software development is a process of gradual refinement
  - Accuracy of estimate increases as the project progresses and the software product becomes more clear
- Resistance of management, customers and developers

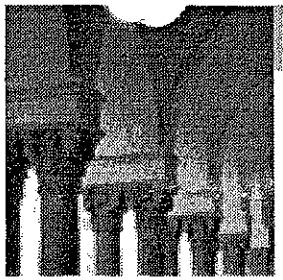
  "Real men (and women) just start coding"

  "We've always done it this way"



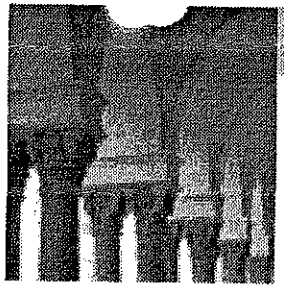
Software Estimating Techniques

- Analogy
- Expert Judgment
- Delphi
- Bottom Up
- Parametric Estimating Models



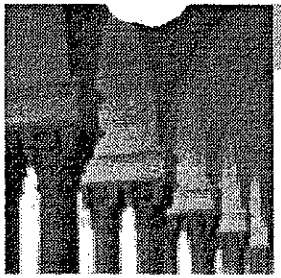
Analogy

- Comparison with similar projects
- Estimates are based on actual experience
- Similar project must exist
- Good historic documentation must exist for estimates and actuals



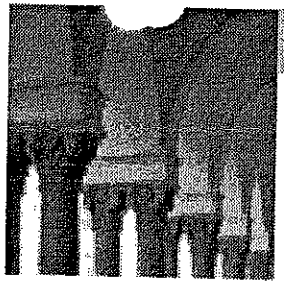
Expert Judgment

- Consult with one or more subject matter experts
- Little or no historical data is needed
- Good for new or unique projects
- Challenges
  - Subject matter experts tend to be biased
  - Many aren't really subject matter experts



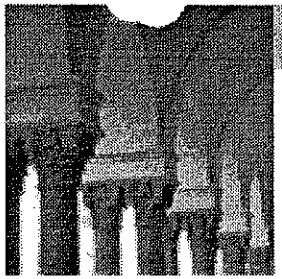
Delphi

- Expert judgment technique where experts are provided system information and a ballot form
  - Each ballot indicates an "optimistic," "pessimistic" and "most likely" estimate for a task
  - Experts score or vote on each task via blind ballots
  - Results are tallied by a coordinator
- Challenges
  - Time consuming
  - Integration costs are sometimes disregarded
  - Very sensitive to the level of information available
  - Experts might not be so expert on your type of project



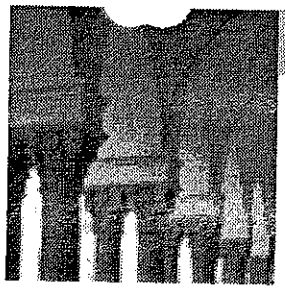
Bottom Up

- Development team estimates each component
- Component estimates are summed to calculate the total estimate
- Detailed basis of estimate promotes individual responsibility
- Challenges
  - Time consuming
  - Detailed data may not be available, especially early in a project
  - Integration costs are sometimes disregarded



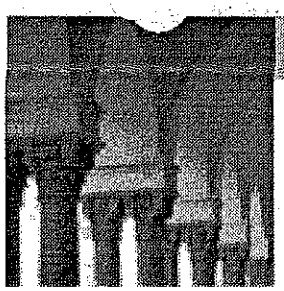
Parametric Estimating Models

- Perform overall estimate using design parameters and mathematical algorithms
- Models are usually fast and "easy" to use
- Useful early in a project
- Challenges
  - Models are inaccurate if not calibrated and validated
  - Historical data used for calibration may not be relevant to new projects
  - User must be familiar with the details of the operation of the model



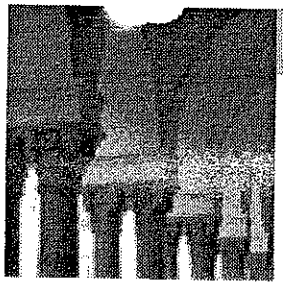
Parametric Estimating Tools

- Knowledge Plan™ - Software Productivity Research (SPR), Inc (www.spr.com)
  - Parametric estimating model
  - Large repository of historical project information
  - Can tune the historical database with own projects
- Construx Estimate™ - Construx Software (www.construx.com)
  - Parametric estimating model
  - Can calibrate with historical data



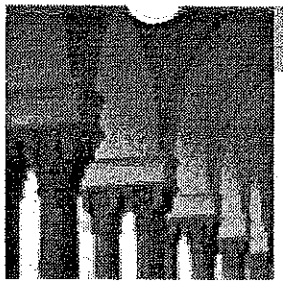
COCOMO II

- Objective cost model for planning and executing software projects
- Developed by Barry Boehm as an update to COCOMO (Constructive Cost Model)



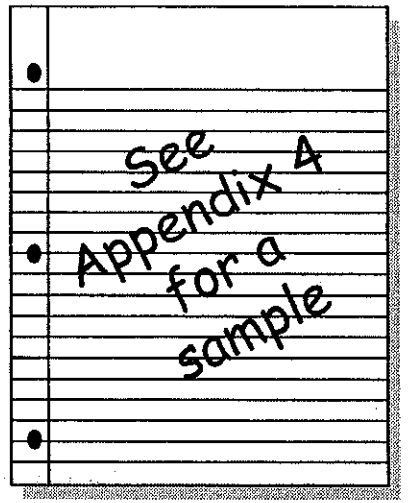
COCOMO II Objectives

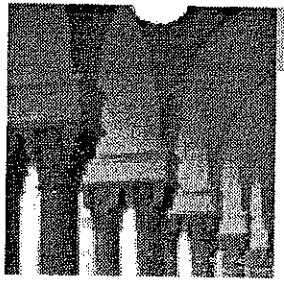
- Provide accurate cost and schedule estimates for current and future software projects
- Enable organizations tailor or extend COCOMO II to their unique situations
- Provide a constructive model
- Provide an evolving model
- Provide careful, easy to understand definitions of the model's inputs, outputs and assumptions



Risk Management Plan

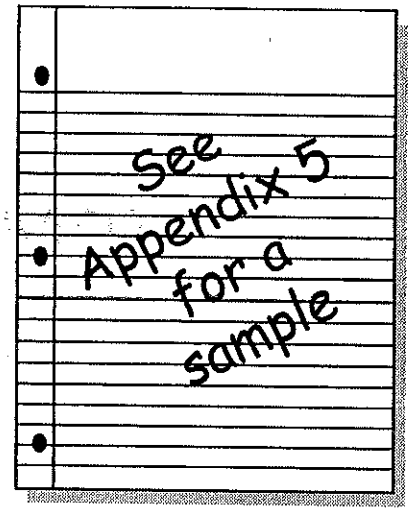
- Documents the risk analysis methodology
- Identifies risk events
- Details risk event mitigation
- May be included as part of the SDP or as a separate document
- Approved by Project Manager

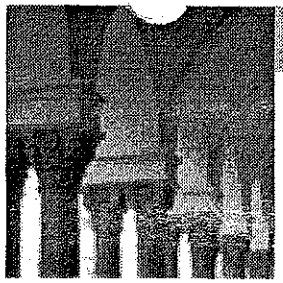




Software Quality Assurance Plan

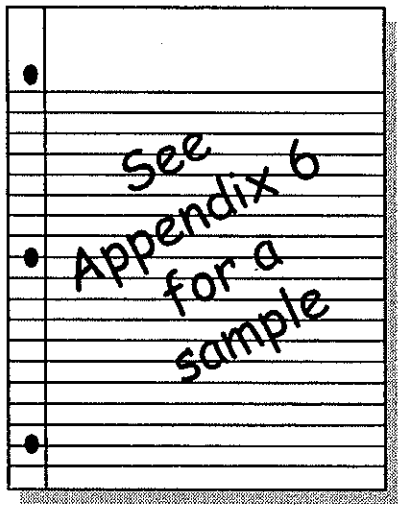
- Prescribes the scope, approach, resources and schedule of quality assurance activities
- Provides assurance that the software will perform to its technical and operational requirements, as defined in the requirements and design specifications
- Delivered with the SDP
  - Some organizations use a standing SQAP due to their large number of small projects

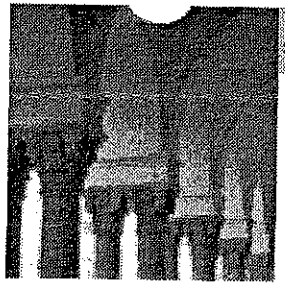




Software Configuration Management Plan (SCMP)

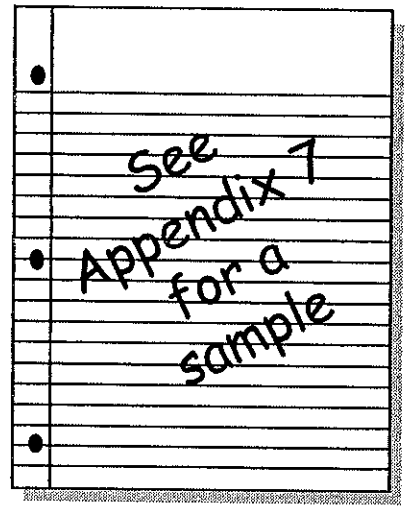
- Addresses the activities to be performed on a project, including schedule and responsibilities for
  - Creation and management of the project's software baseline library
  - Identification of the work products to placed under configuration control
  - Controlling access to the baseline library
  - Creation of products from the baseline library
  - Configuration management reports
- Delivered with the SDP
  - May be an appendix or separate document
  - May be a standard document

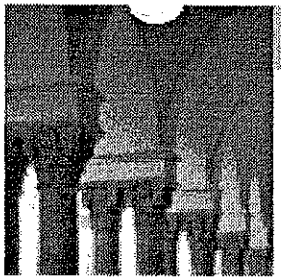




Verification and Validation Plan (V&V)

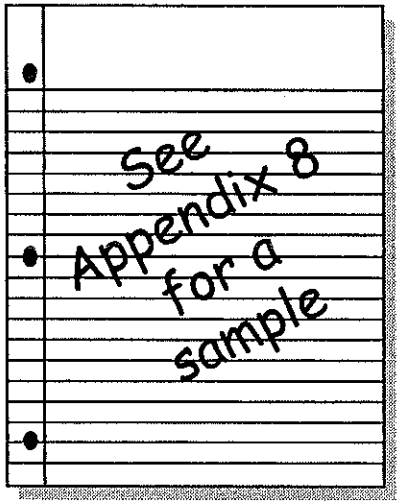
- Provides a comprehensive guide for
  - Activities that ensure the software correctly implements the required functions (verification)
  - Activities that ensure that the software that has been built is traceable to customer requirements (validation)
  - Activities include analysis, documentation, peer reviews and testing
- Delivered after the SDP
  - Developed in parallel with project planning activities
- V & V activities take place throughout the software life cycle

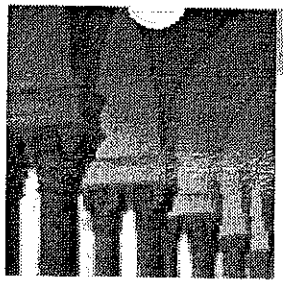




Maintenance Plan

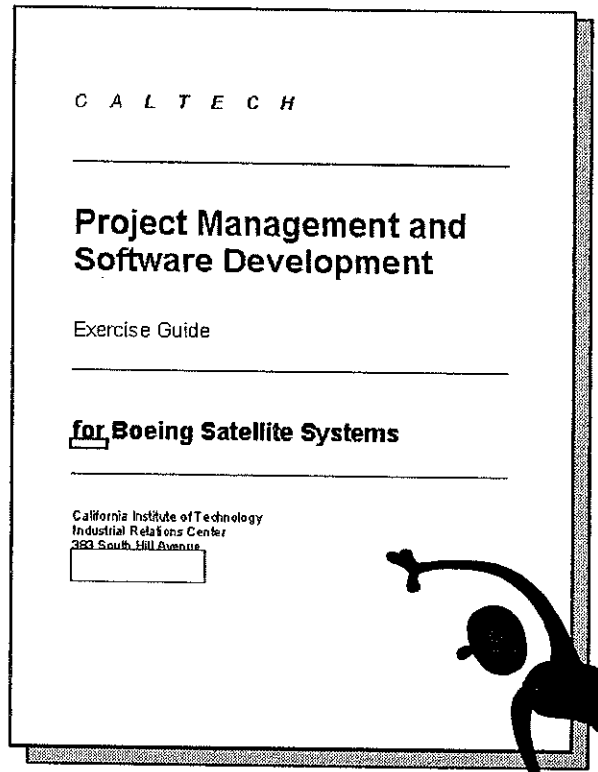
- Provides a comprehensive guide to maintenance and support activities for the software, associated subsystems, objects and supporting applications
- Provides the methodology for
  - Implementation of maintenance and enhancement releases
  - Details on how to implement changes
  - Maintenance Configuration Control
  - Application backup and recovery responsibilities
- Typically delivered with the software

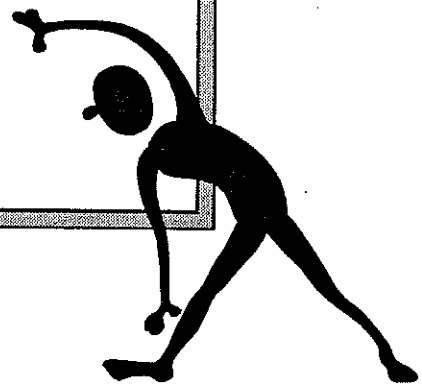


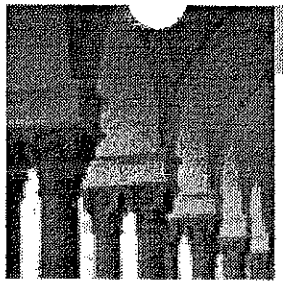


Exercise

- 4.2 – Estimating: Buy or Build

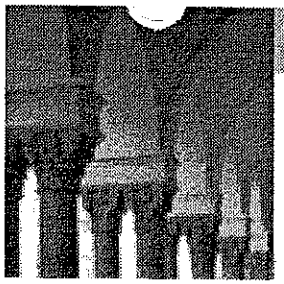






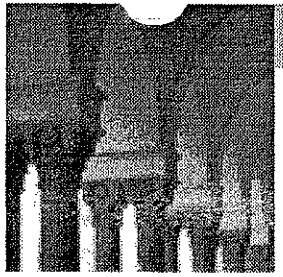
Feature Point

- Bottom up parametric estimating technique based on the application development language and the number of features (screens and reports) to be delivered
- Best used when requirements have been defined
- Provides size and effort estimate
- Duration estimate can be derived from project schedule using effort estimate and planned resources



Feature Point Estimating Steps

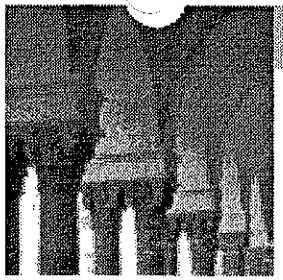
- Estimate the following features
  - Application window construction effort
  - Report construction effort
  - Application interface construction effort
- Use the construction effort estimates to develop estimates for:
  - Application analysis effort
  - Design effort
  - Formal review effort
  - System testing effort
  - Configuration management effort
  - Project management effort



Estimate Application Window
Construction Effort

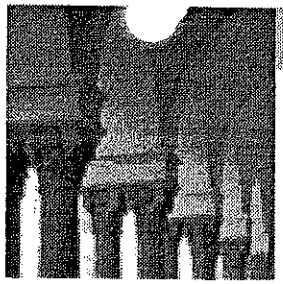
Powerbuilder Window Construction (Code and Unit Test)
Feature Point Estimation Table

Level of Complexity	Description	Effort per Window	Qty	Total Mhrs *Low*	Total Mhrs *High*	Total Mhrs *Most Likely*
Low	• Database retrieval only • No data input • Display only • Functional menu	8 mhrs				
Moderate	• Retrieval/Update • Updateable, freeform data window • Predefined searches with less than 5 criteria • Minimal processing logic • Validations for input formatting (numeric/alpha, drop-down data windows) • 2 or less data windows	24-56 mhrs				
High	• Updateable, tabular data window • Complex SQL • Non-defined search combinations • External device interface (bar code, image scanner, etc.) • Inter-window dependency • Interfaces to external applications • Complicated processing logic/algorithms • Validations (tables, other columns/fields)	80-160 mhrs				
	TOTAL Mhrs, Low, High, Most Likely					



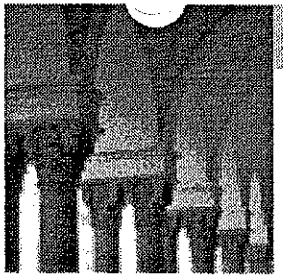
Function Point Estimating

- Parametric estimating technique that can provide a more accurate measure of program size
  - Easier to calculate from a requirements specification than lines of code
  - Provide more realistic measure of productivity
    - Lines of code tend to penalize 3rd generation languages
  - Remains constant regardless of programming language
- Supported by many software estimating tools
- International Function Point user group
  - www.ifpug.org



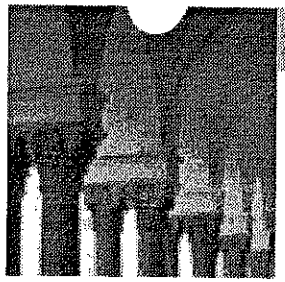
Function Point Estimating

- Can use organization or industry historical data to calculate project effort and duration
- Organizational historical data gives a more accurate estimate
  - $141/function point developed
  - $497/function point enhanced
  - 2.40 Man-hours/function point developed
  - 4.68 Man-hours/function point enhanced



What is a Function Point?

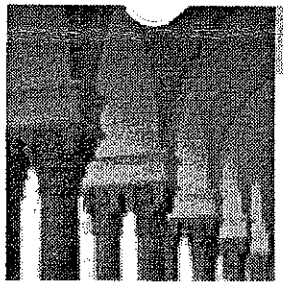
- Functions points are weighted sums of five different factors that relate to user requirements
  - External inputs (EI): Input screens and tables
  - External Outputs (EO): Output screens and reports
  - External Inquiries (EQ): Prompts and interrupts
  - Internal Files (ILF): Databases and directories
  - External Interfaces (EIF): Shared databases and shared mathematical routines



Function Point Calculations

- "Basic" Function Points (BFP)
  - 4(EI) + 5(EQ) + 10(ILF) + 7 (EIF) *with ± 25% Complexity Adjustment*
- Unadjusted Function Points (UFP)
  - Weight five attributes as simple, average or complex

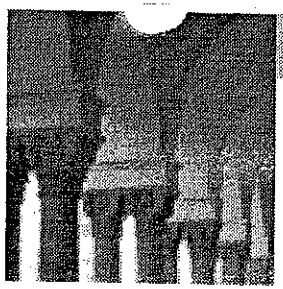
Attribute	Complexity			Total
	Simple	Average	Complex	
EI	3	4	6	
EO	4	5	7	
EQ	3	4	6 (or 7)	
ILF	7	10	15	
EIF	5	7	10	



Function Point Calculations

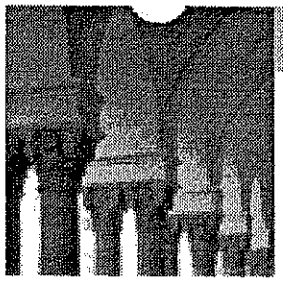
- Adjusted Function Points (AFP)
  - UFP X (0.65 + [0.01(CA)])
    (CA is Complexity Adjustment: Sum of 14 Factors, Rated 1 to 5 for Influence [0 - None, 1 - Little, 2 - Moderate, 3 - Average, 4 - Significant, 5 - Strong]; Ratings Defined for Each Factor)
- 14 Factors

1. Data Communications
2. Distributed Data Processing
3. Performance Objectives
4. Heavily-Used Configuration
5. Transaction Rate
6. On-Line Data Entry
7. End-User Efficiency
8. On-Line Update
9. Complex Processing
10. Reusability
11. Conversion and Installation Ease
12. Operational Ease
13. Multiple Site Usage
14. Facilitate Change



Function Point Estimating Challenges

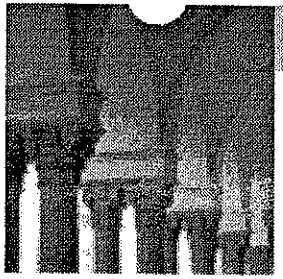
- Difficult to visualize
- Have only been studied extensively for business and data processing applications
  - Some attempts to adapt function point concepts to real-time and scientific environments have occurred
- Accurate counting requires certified specialists
- Can be time-consuming and expensive
- Automated tools are of unknown accuracy



Estimate Selection Matrix

- Example
  - Used by the Application Services Group at San Onofre Nuclear Generation Station
  - Shows recommended methods to apply for indicated project phases (C = Conceptual; R = Requirements; D = Design)

Estimate Type → / Project Type ↓	Analogy	Delphi	Feature / Function Point	Knowledge Plan (KP)
New Development	C	C, R, D	R, D	C, R, D
Enhancement	C, R, D	C, R, D	R, D	C, R, D
Maintenance	C, R, D	C, R, D	R, D	C, R, D
Conversion	C, R, D	C, R, D	R, D	C, R, D



In Summary

- Can you list the objectives of the project kick-off meeting?
- Can you identify the key planning documents recommended for a software project?
- List four different methods used to estimate software projects?