Software Development Management

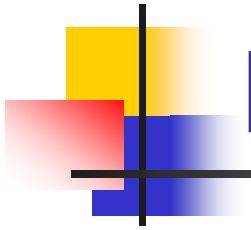
Lecture 6

Size and Cost Estimation

By: Hossein Momeni

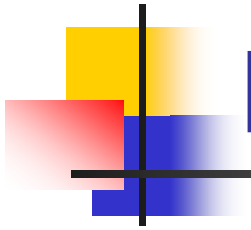
momeni@iust.ac.ir

Mazandaran University of Science and Technology



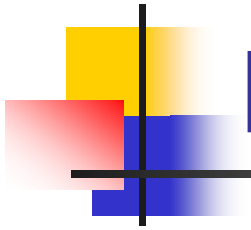
Different level of estimation

- Before a project is decided to pursue
  - The estimation is coarse
  - The estimation is in high level terms
    - Profit? Good to the organization? etc.
- After the project is decided to go ahead
  - More detailed size and cost estimations are required



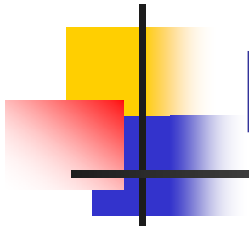
Project Evaluation

- It is a high level assessment of the project
  - to see whether the project is worthwhile to proceed
  - to see whether the project will fit in the strategic planning of the whole organization



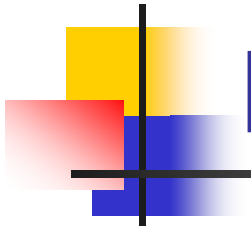
Project Evaluation - Why

- Want to decide whether a project can proceed before it is too late
- Want to decide which of the several alternative projects has a better success rate, a higher turnover, a higher ...
- Is it desirable to carry out the development and operation of the software system



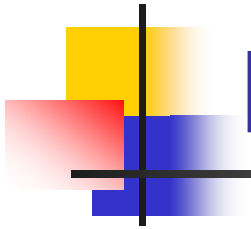
Project Evaluation - Who

- Senior management
- Project manager/coordinator
- Team leader



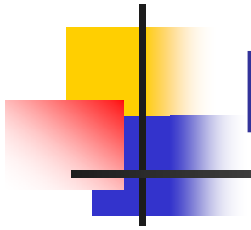
Project Evaluation - When

- Usually at the beginning of the project
  - e.g. Step 0 of Step Wise Framework



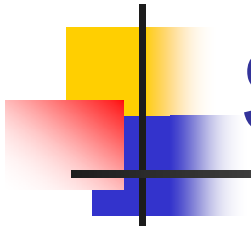
Project Evaluation - What

- Strategic assessment
- Technical assessment
- Economical assessment



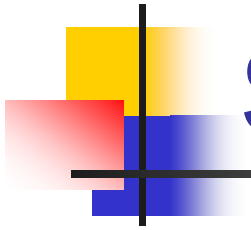
Project Evaluation - How

- Cost-benefit analysis
- Cash flow forecasting
- Cost-benefit evaluation techniques
- Risk Analysis



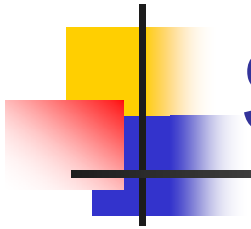
Strategic Assessment

- Use to assess whether a project fits in the *long-term goal* of the organization
- Usually carry out by senior management
- Need a strategic plan that clearly defines the objectives of the organization
- Evaluate individual projects against the strategic plan or the overall business objectives



Strategic Assessment (cont'd)

- Programme management
  - suitable for projects developed for use in the organizations
- Portfolio management
  - suitable for project developed for other companies by software houses



SA – Programme Management

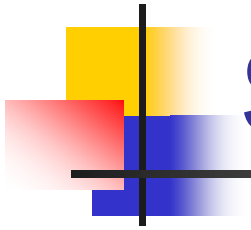
- Individual projects as components of a programme within the organization

  Programme as "a group of projects that are managed in a coordinated way to gain benefits that would not be possible were the projects to be managed independently"

  by D.C. Ferns

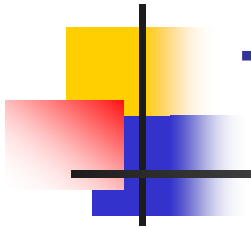
  Journal of Project Management

  Aug. 1991



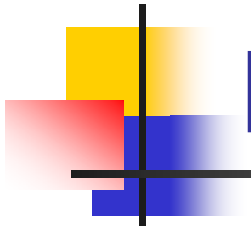
SA – Portfolio Management

- suitable for product developed by a software company for an organization
  - outsourcing
- need to assess the product for the client organization
  - Programme management issues apply
- need to carry out strategic assessment for the servicing software company



Technical Assessment

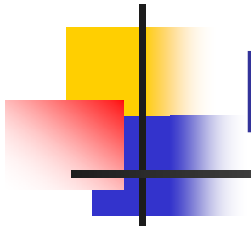
- Functionality against hardware and software
- The strategic IS plan of the organization
- any constraints imposed by the IS plan



Economic Assessment

Why?

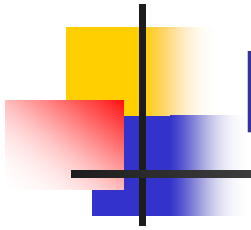
- Consider whether the project is the best among other options
- Prioritise the projects so that the resources can be allocated effectively if several projects are underway



Economic Assessment (cont'd)

How?

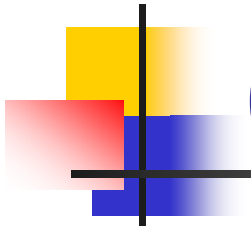
- Cost-benefit analysis
- Cash flow forecasting
- Various cost-benefit evaluation techniques
  - NPV and IRR



EA – Cost-benefit Analysis

- A standard way to assess the economic benefits
- Two steps
  - Identify and estimate all the costs and benefits of carrying out the project
  - Express the costs and benefits in a common unit for easy comparison (e.g. $)

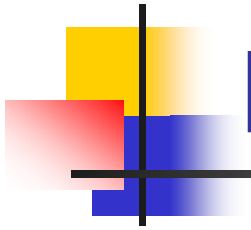
EA – Cost-benefit Analysis (cont'd)



- Costs
  - Development costs
  - Setup costs
  - Operational costs

EA – Cost-benefit Analysis (cont'd)

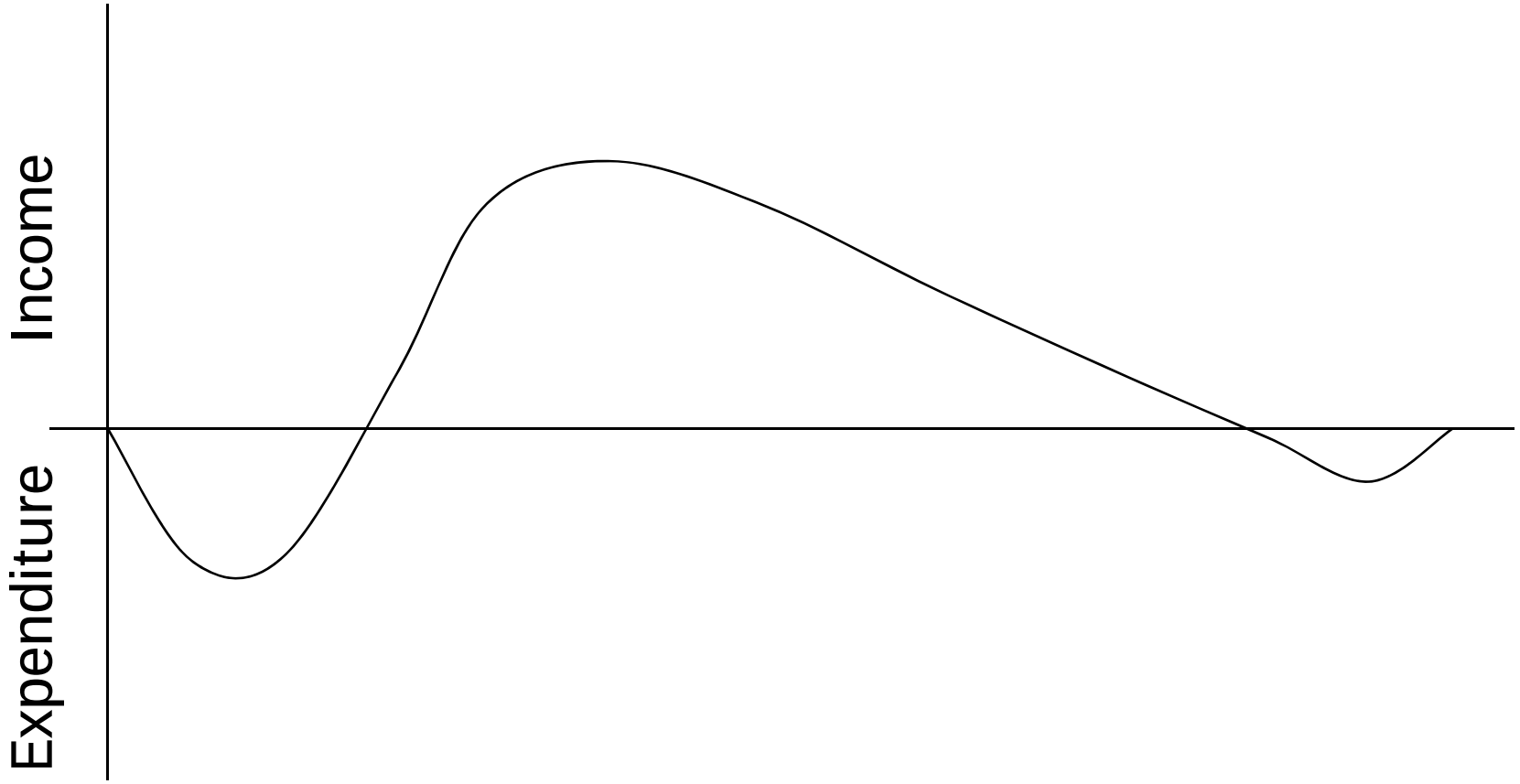
- Benefit
  - Direct benefits
    - Reduction of staff employment
  - Assessable indirect benefits
    - Increase of accuracy through a more user-friendly screen
  - Intangible benefits
    - Enhanced job interest



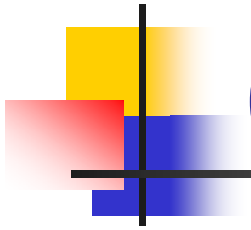
EA – Cash Flow Forecasting

- What?
  - Estimation of the cash flow **over time**
- Why?
  - An excess of estimated benefits over the estimated costs is not sufficient
  - Need detailed estimation of benefits and costs versus time

EA – Cash Flow Forecasting (Cont'd)



EA – Cash Flow Forecasting (Cont'd)



- Need to forecast the expenditure and the income
- Accurate forecast is not easy
- Need to revise the forecast from time to time

Cost-benefit Evaluation Techniques Example

Year	*Project 1*	*Project 2*	*Project 3*	*Project 4*
0	-100,000	-1,000,000	-100,000	-120,000
1	10,000	200,000	30,000	30,000
2	10,000	200,000	30,000	30,000
3	20,000	200,000	30,000	30,000
4	20,000	200,000	20,000	25,000
5	100,000	350,000	20,000	50,000
Net Profit	60,000	150,000	30,000	45,000
Payback	5	5	4	4
ROI	12%	3%	6%	7.5%

Cost-benefit Evaluation Techniques

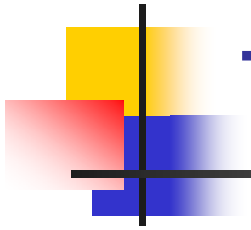
- Net profit
  = Total income – Total costs
- Payback period
  = Time taken to break even
- Return on Investment (ROI)

$$= \frac{\text{average annual profit}}{\text{total investment}} \times 100\%$$

- Example project 1:

$$= \frac{12000}{100000} \times 100\% = 12\%$$

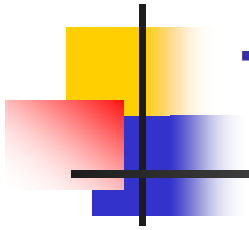
Cost-benefit Evaluation Techniques – NPV



Net present value (NPV)

- It is the sum of the present values of all future amounts.
- *Present value* is the value of which a future amount worth at present
- It takes into account the profitability of a project and the timing of the cash flows

Cost-benefit Evaluation Techniques – NPV (cont'd)



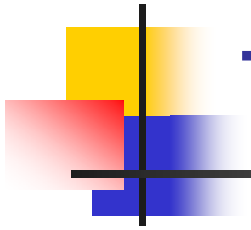
- *Discount rate* is the annual rate by which we discount future earning
  - e.g. If discount rate is 10% and the return of an investment in a year is $110, the present value of the investment is $100.

Cost-benefit Evaluation Techniques – NPV (cont'd)

- Let *n* be the number of year and *r* be the discount rate, the present value (PV) is given by

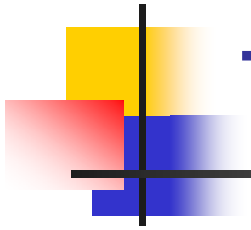
$$\mathrm{PV} = \frac{\text{value in year } n}{(1+r)^n}$$

Cost-benefit Evaluation Techniques – NPV (cont'd)

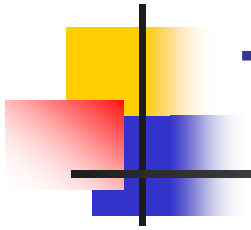


- Issues in NPV
  - Choosing an appropriate discount rate is difficult
  - Ensuring that the rankings of projects are not sensitive to small changes in discount rate

Cost-benefit Evaluation Techniques – NPV (cont'd)



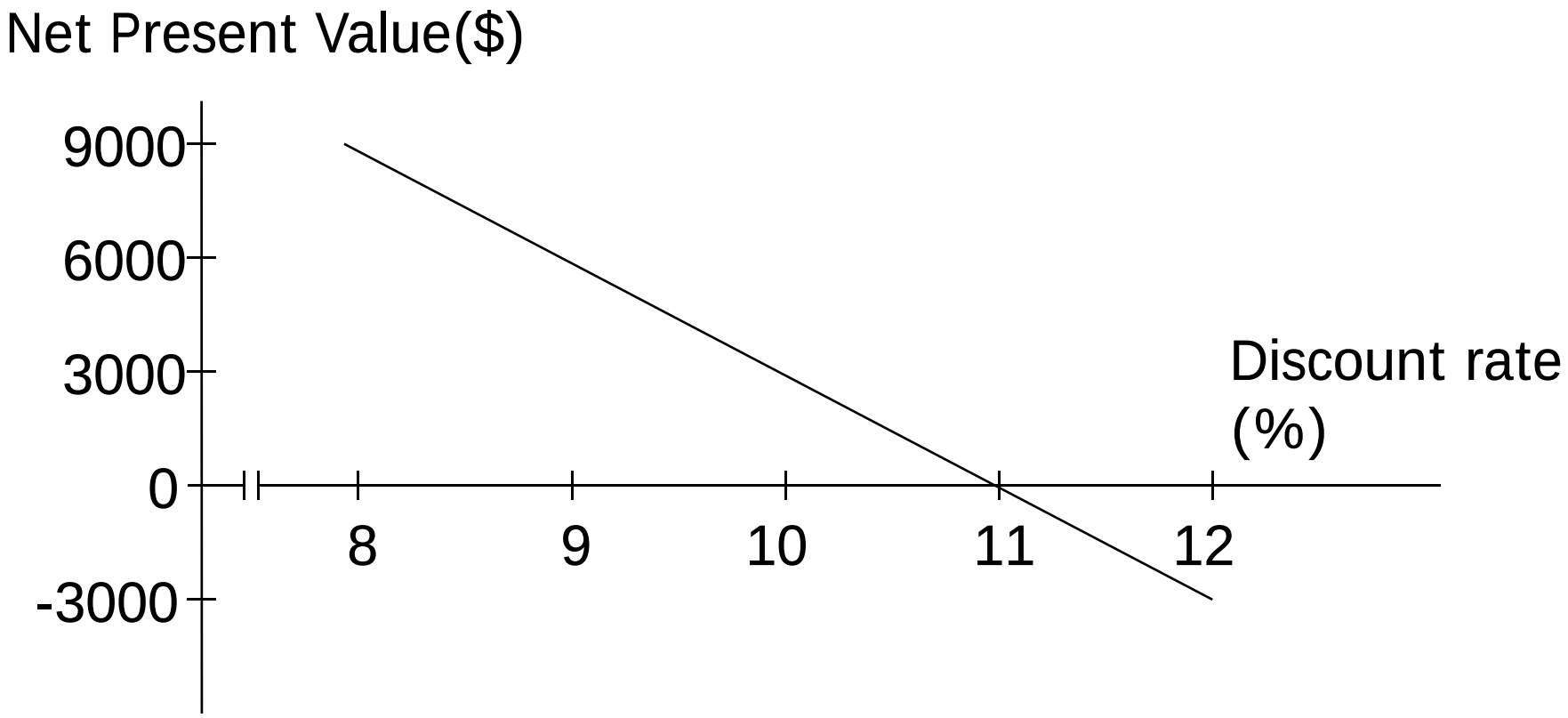
- Disadvantage
  - May not be directly comparable with earnings from other investments or the costs of borrowing capital



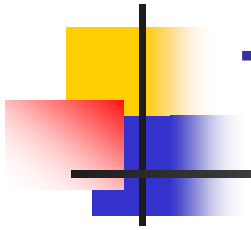
Cost-benefit Evaluation Techniques – IRR

- Internal Rate of Return (IRR)
  - The percentage discount rate that would produce a NPV of zero
  - A relative measure

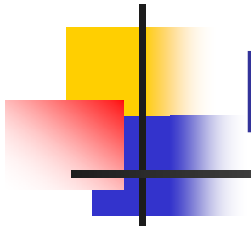
Cost-benefit Evaluation Techniques – IRR (cont'd)



Cost-benefit Evaluation Techniques – IRR (cont'd)

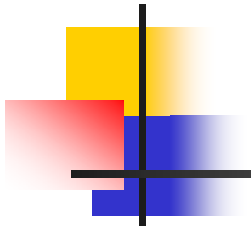


- Advantages
  - Convenient
    - Directly comparable with rate of return on other projects and with interest rates
  - Useful
    - Dismiss a project due to its small IRR value
    - Indicate further precise evaluation of a project



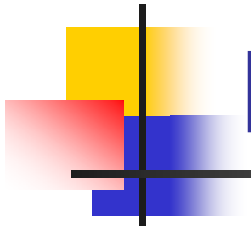
Estimation

- Why? – to define the project budget and to ‘refine’ the product to realize the budget
- Who? – the manager
- What? – size and cost
- When? – always
- How? – techniques and models



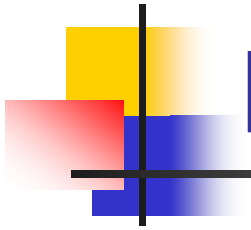
Issues related to Estimation

- Difficult to make accurate estimation
- Better to have previous data and analyze the actual values against their estimates so that you know how accurate you are
- Even better to have previous data of the whole organization so that you know how accurate the estimation method, if any, used within the organization is



Positive Attitude Towards Estimation

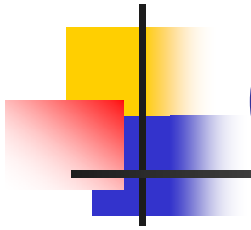
- Use your estimation as a guide to manage your project
- From time to time, you need to revise your estimation based on the current status of the project



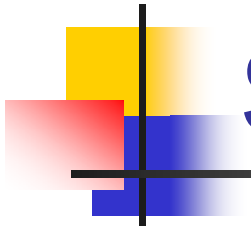
Estimation Approaches

- Expert judgement
  - Ask the knowledgeable experts
- Estimation by analogy
  - Use the data of a similar and completed project
- Pricing to win
  - Use the price that is low enough to win the contract

Estimation Approaches (cont'd)



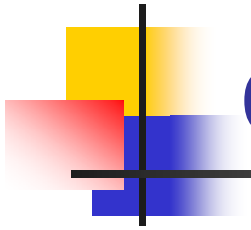
- Top-down
  - An overall estimate is determined and then broken down into each component task
- Bottom-up
  - The estimates of each component task are aggregate to form the overall estimate
- Algorithmic model
  - Estimation is based on the characteristics of the product and the development environment.



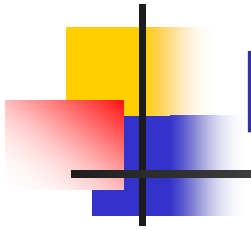
Size Estimation

- Problems related to size estimation
- Size Estimation Model
  - Function Point Analysis (FPA)
    - FPA is a top-down approach.

Problems related to size estimation



- Nature of software
- Novel application of software
- Fast changing technology
- Lack of homogeneity of project experience
- Subjective nature of estimation
- Political implication with the organization

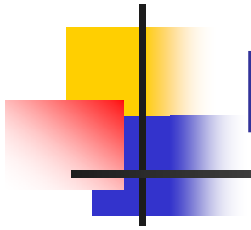


Function Point Analysis (FPA)

- Developed by A. Albrecht in IBM
- Aim: To estimate the LOC of a system

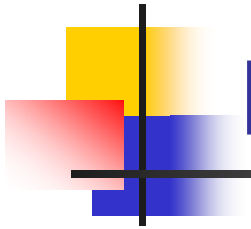
LOC of system

= FP of system × LOC-per-FP of the language



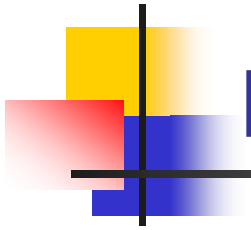
Line of Code

- LOC means Line of Code (programming statement)
  - For COBOL, the LOC per FP is 91.
  - For C, the LOC per FP is 128.
  - For Fortran, the LOC per FP is 106.
  - For VB, the LOC per FP is 32.
  - For Pascal, the LOC per FP is 90.
  - For Assembly, the LOC per FP is 320.



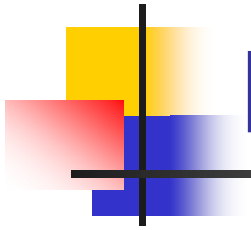
Function Point Analysis (cont'd)

- Idea: Software system comprises of five major components (or, *external user type*)
  - External input types
    - Input transactions that update internal computer files
  - External output types
    - Transactions that output data to user such as report printing



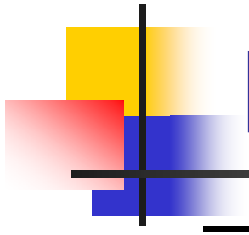
Function Point Analysis (cont'd)

- Logical internal file types
  - The standing file used by the system
- External interface file types
  - Input and output that may pass from and to other computer applications
- External inquiry types
  - Transactions initiated by the user that provide information but do not update the internal files



Function Point Analysis - Steps

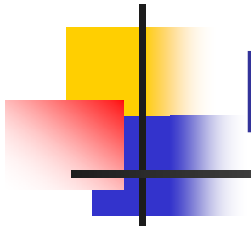
- Identify each instance of each external user type in the proposed system
- Classify each instance as having high, medium or low complexity
- Assign the FP of each instance
- FP of the system = sum of FP of individual components



Function Point Analysis

Number of FPs	Complexity		
External user type	Low	Average	High
External input type	3	4	6
External output type	4	5	7
Logical internal file type	7	10	15
External interface file type	5	7	10
External inquiry type	3	4	6

Function Point Analysis - Example

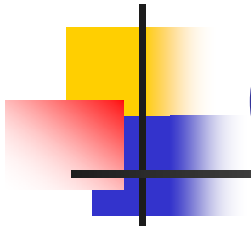


- A component of an inventory system consisting of ‘Add a record’, ‘Delete a record’, ‘Display a record’, ‘Edit a record’, and ‘Print a record’ will have
  - 3 external input types (all of low complexity)
  - 1 external output type (average complexity)
  - 1 external inquiry type (high complexity)

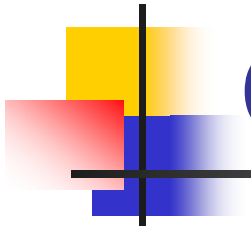
  Then, assign FPs based on the complexity of each external types

  Result: 3*3 + 1*5 + 1*6 = 20.

Function Point Analysis (cont'd)

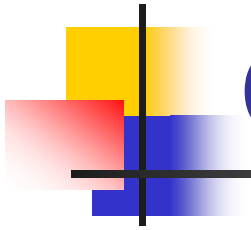


- Other issues
  - The assignment of level of complexity is rather subjective
  - International FP User Group (IFPUG) imposes rules on assigning the level of complexity to individual external user types



Object Point Analysis

- Similar to function point analysis
- Used on 4GL development projects
- Take account of features that may be more readily identifiable if the system is built on a high-level application building tools

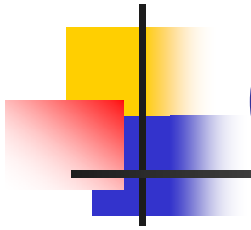


Object Point Analysis – Steps

- Identify the number of screens, reports and 3GL components
- Classify each object as Simple, Medium and Difficult
- Assign the weight accordingly
- Calculate the total object points

  Total OP = sum of individual OP × weighting

Object Point Analysis – Steps (cont'd)



- Deduct the reused objects (r% reused)

$$NOP = OP \times (1 - r\%)$$

- Identify the productivity rate of both developer and CASE
- Productivity rate = average of the two PRs
- Calculate the effort

$$Effort = NOP / Productivity\ Rate$$

Object Point Analysis – Screens

Number of views contained	Number and source of data tables		
	Total < 4 (<2 server, <2 client)	Total < 8 (2-3 server, 3-5 client)	Total 8+ (>3 server, >5 client)
< 3	Simple	Simple	Medium
3 – 7	Simple	Medium	Difficult
8+	Medium	Difficult	Difficult

Object Point Analysis – Reports

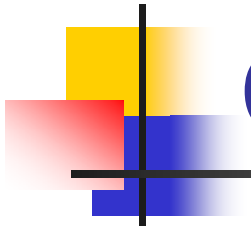
Number of sections contained	Number and source of data tables		
	Total < 4 (<2 server, <2 client)	Total < 8 (2-3 server, 3-5 client)	Total 8+ (>3 server, >5 client)
< 2	Simple	Simple	Medium
2 or 3	Simple	Medium	Difficult
> 3	Medium	Difficult	Difficult

Object Point Analysis – Complexity Weightings

	Complexity		
Type of object	Simple	Medium	Difficult
Screen	1	2	3
Report	2	5	8
3GL component	N/A	N/A	10

Object Point Analysis – Productivity Rate

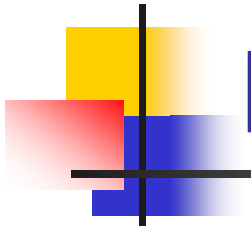
	Very low	Low	Nominal	High	Very High
Developer's experience and capability	4	7	13	25	50
CASE maturity and capability	4	7	13	25	50



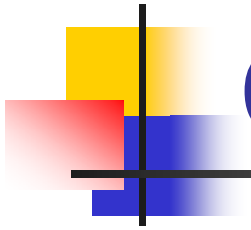
Object Point Analysis – Issues

- Adopted in Boehm's COCOMO II in the application composition stage

Object Point Analysis – Example

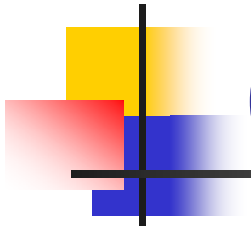


- See separate handout



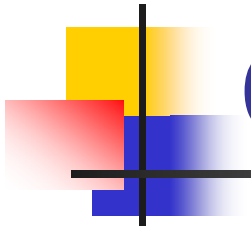
Cost Estimation

- Cost Estimation Model
  - COCOMO II



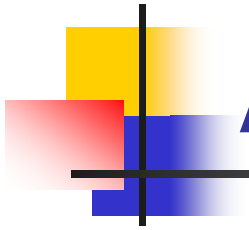
Constructive Cost Model II (COCOMO II)

- A parametric cost model
  - Important aspects of software project are characterized by variables (or, parameters)
  - Once the value of the parameters are determined, the cost can be computed from the equation



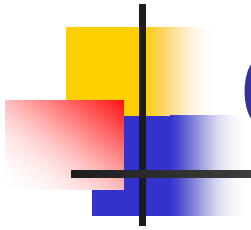
COCOMO II (cont'd)

- Recognize different approaches to software development
  - Prototyping, Incremental development etc.



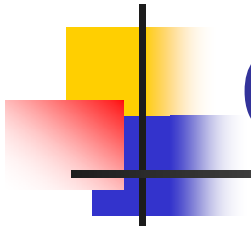
A history of COCOMOs

- *COCOMO* originally proposed by Boehm in 1981, now called *COCOMO 81*
- Later evolved to *Ada COCOMO* in 1989
- In 1995, Boehm proposes *COCOMO II*



COCOMO II

- A family of models
  - Use different models in 3 different stages of the project
- 3 stages: application composition, early design and post architecture
  - Support estimation early in the process
  - Allow further detailed estimation after the system architecture has been defined



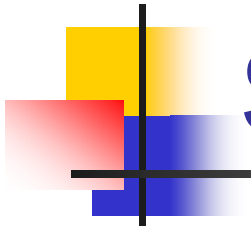
COCOMO II (cont'd)

- The basic model equation

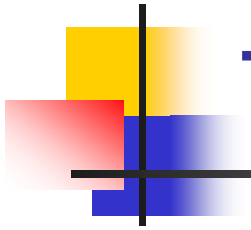
$$\text{Effort} = \text{Constant} \times (\text{Size})^{\text{scale factor}} \times \text{Effort Multiplier}$$

  - Effort in terms of person-month
  - Size: Estimated Size in KSLOC
  - Scale Factor: a combined effects of factors related to the process
  - Effort Multiplier (EM): a combined effect of factors related to the effort

The Application Composition Stage



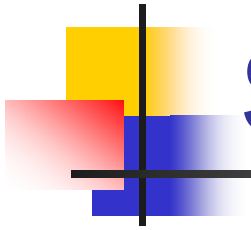
- Estimation at the early stage
- Corresponding to exploratory work such as prototyping
- Use object points to estimate the size of the product



The Early Design Stage

- Estimate after the requirements specification is completed and possibly with some design
- Use the basic model equation
- Estimate the size by FPs (preferred) or KSLOC
- Assign process exponent estimation accordingly

The Early Design Stage – Scale Factor

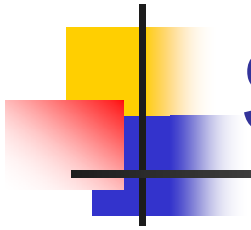


- Estimation on the scale factor
  - A combined effect of 5 parameters
- Application precedentedness
- Process flexibility
- Architecture risk resolution
- Team cohesion
- Process maturity

The Early Design Stage – Scale Factor (cont'd)

Parameter	Very Low (0.05)	Low (0.04)	Nominal (0.03)	High (0.02)	Very High (0.01)	Extra High (0.00)
Precedentedness	Thoroughly unprecedented	Largely unprecedented	Somewhat unprecedented	Generally familiar	Largely familiar	Thoroughly familiar
Development flexibility	Rigorous	Occasional relaxation	Some relaxation	General conformity	Some conformity	General goals
Architecture risk resolution	Little 20%	Some 40%	Often 60%	Generally 75%	Mostly 90%	Full 100%
Team cohesion	Very difficult interactions	Some difficult interactions	Basically cooperative	Largely cooperative	Highly Cooperative	Seamless interactions
Process maturity	Level 1	Level 2	Level 2+	Level 3	Level 4	Level 5

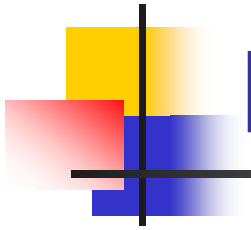
The Early Design Stage – Scale Factor (Cont'd)



- Calculate the scale factor based on the equation

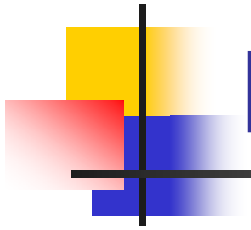
  Scale factor = 1.01 + sum of the values

The Early Design Stage – Effort Multiplier



- 7 factors on Effort Multiplier
  - product Reliability and ComPleXity (RCPX)
  - required reusability (RUSE)
  - Platform DIFficulty (PDIF)
  - PERSonnel capability (PERS)
  - PeRsonnel EXperience (PREX)
  - FaCILities available (FCIL)
  - SChEDule pressure (SCED)

The Early Design Stage – Effort Multiplier (cont'd)

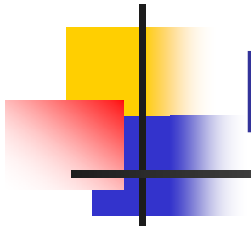


- Assess each factor by
  - Very low, low, nominal, high, very high, and extra high
- Assign each factor using a value between 0.5 and 1.5 (inclusive)
- EM is the product of all these values

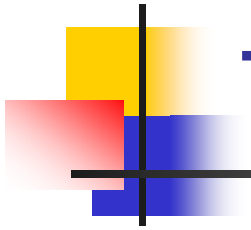
The Early Design Stage – Effort Multiplier (cont'd)

Early Design	Very Low – Extra High
RCPX	0.5 – 1.5
RUSE	0.5 – 1.5
PDIF	0.5 – 1.5
PERS	1.5 – 0.5
PREX	1.5 – 0.5
FCIL	1.5 – 0.5
SCED	1.5 – 0.5

The Early Design Stage – Example



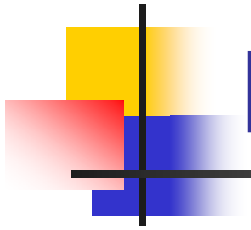
- See separate handout



The Post-architecture Stage

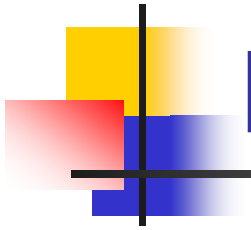
- Estimation after the software architecture has been defined
- The same basic model equation
- Size estimation by KSLOC (preferred) or FPs
- Same process exponent estimation
- 17 factors in EM (more than 7 in early design stage)

The Post-architecture Stage – Effort Multiplier



- 17 factors in 4 different categories
  - Product attributes
  - Platform attributes
  - Personnel attributes
  - Project attributes

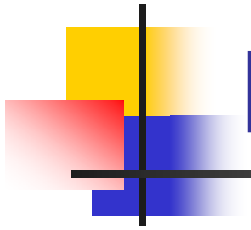
The Post-architecture Stage – Effort Multiplier



- Product attributes
  - Required reliability (RELY)*
  - Database size (DATA)
  - Product complexity (CPLX)*
  - Required reuse (RUSE)**
  - Documentation (DOCU)

*Relate to RCPX in early design stage

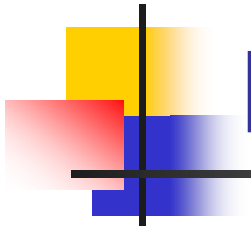
The Post-architecture Stage – EAF (Cont'd)



- Platform attributes
  - execution TIME constraint (TIME)*
  - main STORage constraint (STOR)*
  - Platform VOLatility (PVOL)*

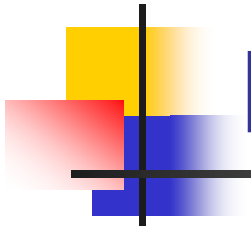
  *Related to Platform DIFficulty (PDIF) in early design stage

The Post-architecture Stage – EAF (Cont'd)



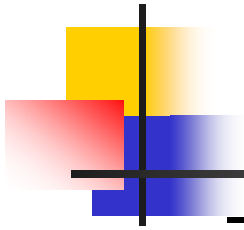
- Personnel attributes
  - Analyst CAPabilities (ACAP)^
  - Application EXPerience (AEXP)*
  - Programmer CAPabilities (PCAP)^
  - Personnel EXPerience (PEXP)*
  - programming Language/Tool EXperience (LTEX)*
  - Personnel CONtinuity (PCON)^

The Post-architecture Stage – EAF (Cont'd)



- Project attributes
  - use of software TOOLs (TOOL)*
  - multiSITE development team communications (SITE)*

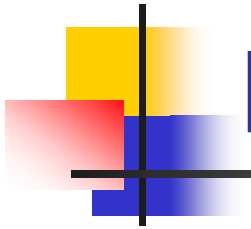
*Relate to FCIL in early design model



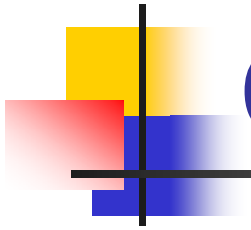
EAF Relations

Early Design	Post-Architecture
RCPX	RELY, DATA, CPLX, DOCU
RUSE	RUSE
PDIF	TIME, STOR, PVOL
PERS	ACAP, PCAP, PCON
PREX	AEXP, PEXP, LTEX
FCIL	TOOL, SITE
SCED	SCED

The Post-architecture Stage – Example



- See separate handout



COCOMO II (cont'd)

- Advantages
  - Good improvement over COCOMO
  - Good match for iterative development, modern technology, and management process
- Disadvantages
  - Still immature, diverse projects in database
  - Hard to believe that it will be any more reliable than the original COCOMO model