Unit III

Software Development

Object Points

- Method developed for object-oriented technology
- Counting ”object points” to determine software size
- Conducted at a more macro level than function points
- Assigns one object point to each unique class or object
- Otherwise similar to function and feature points

Model Blitz

- Estimating gets better with each passing phase
- Concept of blitz modelling is counting number of process (object classes) * number of programs per class * avg program size = Estimated size (LOC)
- A historical database is essential
- Function-strong systems and data-strong systems are calculated separately

Contd.

- 20 object classes implemented to 5 procedural programs & on an avg 75 LOC per procedural prg
- No. of Process X no. of programs per class X Avg Prg Size = Estimated Size
- 20 X 5 X 75 = ?
- 7500 LOC

Advantages of Model Blitz

- Easy to use with structured methods and with object-oriented classes
- Accuracy increases with historical data
- Continuous improvement activities used for estimation techniques

Disadvantages of Model Blitz

- Requires use of design methodology
- Estimation can not begin until design is complete
- Requires historical data
- Does not evaluate environmental factors

Wideband Delphi

- Popular and simple technique to estimate size and effort
- Group consensus approach
- Uses experience of several people to reach an estimate

Wideband Delphi steps

1. Present experts with the problem and a response form
2. Group discussion
3. Collect opinions anonymously
4. Feed back a summary of results
5. Another group discussion
6. Iterate until consensus

Advantages of Wideband Delphi

- Easy and inexpensive
- Expertise of several people
- Participants become better educated about the software and project
- Does not require historical data
- Used for high-level and detailed estimation
- Results more accurate than in LOC

Disadvantages of Wideband Delphi

- Difficult to repeat with different group of experts
- Possible to reach consensus on an incorrect estimate, people may not be skeptical enough
- Can develop a false sense of confidence
- May fail to reach a consensus
- Experts may be biased in the same subjective direction

Effects of Reuse on Software Size

- Many softwares are derived from previous programs
- Result in savings of cost and time, increased quality
- Can also cost more, take longer time and yield lower quality
- First step in code reuse is to separate new code from modified and reused code

Effects of Reuse continues

- If the unit has changed, it is modified
- If more than 50% of the unit is changed, it is considered to be ”new”
- Reused code will be converted to equivalent new code
- Conversion factor reflects the amount of effort saved by reuse
- Reuse factors come from experience (e.g. 30% for reused, 60% for modified)
- Can also be done on more accurate level