Project Management – Estimation

Week 11
Announcement

• Midterm 2
  – Wednesday, May. 4
  – Scope
    • Week 11 – Week 13
  – Short answer questions
Agenda (Lecture)

• Estimation
Agenda (Lab)

- Implement a software product based on your design documents
- Submit a weekly project progress report at the end of the Wednesday lab session
How's your project coming along?

It's a steaming pile of failure.

It's like fifteen drunken monkeys with a jigsaw puzzle.

How's your project coming along?

Fine.
Guide to the Software Engineering Body of Knowledge
2004 Version

(a) Software Requirements
   - Software Requirements Fundamentals
   - Requirements Process
   - Requirements Elicitation
   - Requirements Analysis
   - Requirements Specification
   - Requirements Validation
   - Practical Considerations

(b) Software Design
   - Software Design Fundamentals
   - Key Issues in Software Design
   - Software Structure and Architecture
   - Software Design Quality Analysis and Evaluation
   - Software Design Notations
   - Software Design Strategies and Methods

(c) Software Construction
   - Software Construction Fundamentals
   - Managing Construction
   - Practical Considerations

(d) Software Testing
   - Software Testing Fundamentals
   - Test Levels
   - Test Techniques
   - Test Related Measures
   - Test Process

(e) Software Maintenance
   - Software Maintenance Fundamentals
   - Key Issues in Software Maintenance
   - Maintenance Process
   - Techniques for Maintenance

Figure 2 First five KAs
Figure 3 Last six KAs
Software Project Success Rate

Data on 280,000 projects completed in 2000 - Standish Group Data

Statements about Management

• “Software project management is an essential part of software engineering.”
• “Without proper planning, a software development project is doomed.”
• “Good management cannot guarantee project success. However, bad management usually result in project failure: The software is delivered late, costs more than originally estimated, and fails to its requirement.”
Project

• Organizations perform works: operations and projects
• Commonalities between operations and projects
  – Performed by people
  – Constrained by the limited resources
  – Planned, executed, and controlled
• Differences between operations and projects
  – Operations are on-going and repetitive
  – Projects are temporary and unique
Project Management

• Project Management Body Of Knowledge (PMBOK)
  – Project Management Institute
• www.csun.edu/~twang/380/Slides/pmbok.pdf
Software Project Management

• Software project management is especially difficult because ....


• Software project management : The Manager’s View
Metrics

• Numerical measures that quantify the degree to which software, a process or a project possesses a given attribute

• Metrics help the followings
  – Determining software quality level
  – Estimating project schedules
  – Tracking schedule process
  – Determining software size and complexity
  – Determining project cost
  – Process improvement
Software Metrics

• Without measure it is impossible to make a plan, detect problems, and improve a process and product
• A software engineer collects measure and develops metrics so that indicators will be obtained
• An indicator provides insight that enables the project manager or software engineers to adjust the process, the project, or the product to make things better
Software Metrics (cont’d)

• The five essential, fundamental metrics:
  – Size (LOC, etc.)
  – Cost (in dollars)
  – Duration (in months)
  – Effort (in person-month)
  – Quality (number of faults detected)
Product Size Metrics

• Conventional metrics
  – Size-oriented metrics
  – Function-oriented metrics
  – Empirical estimation models
• Object-Oriented metrics
  – Number of scenario scripts
  – Number of key classes
  – Number of support classes
  – Average number of support classes per key classes
• User-Case oriented metrics
Product Size Metrics (cont’d)

• Web engineering product metrics
  – Number of static web pages
  – Number of dynamic web pages
  – Number of internal page links
  – Number of persistent page links
• The accuracy of estimation increases as the process proceeds
Size Estimation

• The methods to achieve reliable size and cost estimates:
  – LOC-based estimation
  – FP-based estimation
  – Empirical estimation models
    • COCOMO
LOC-based Estimation

• The problems of lines of code (LOC)
  – Different languages lead to different lengths of code
  – It is not clear how to count lines of code
  – A report, screen, or GUI generator can generate thousands of lines of code in minutes
  – Depending on the application, the complexity of code is different
LOC-based Estimation - Example

- **Function**
  - User interface
  - 2-D geometric analysis
  - 3-D geometric analysis
  - Database management
  - Graphic display facilities
  - I/O control function
  - Analysis function

- **Estimated LOC**
  - 2,300
  - 5,300
  - 6,800
  - 3,500
  - 4,950
  - 2,100
  - 8,400

- **Total estimated LOC**
  - **33,350**
LOC-based Estimation - Exercise

• Average productivity based on historical data
  – 620 LOC/pm
  – $8,000 per month
    -> $12.91/LOC

• If the estimated project is 33,200 LOC,
  – then the total estimated project cost is $_______ and
  – the estimated effort is __ person-months
FP-based Estimation

• Based on FP metric for the size of a product
  – Based on the number of inputs (Inp), outputs (Out), inquiries (Inq), master files (Maf), interfaces (Inf)
  – Step 1: Classify each component of the product (Inp, Out, Inq, Maf, Inf) as simple, average, or complex (Figure 1)
    • Assign the appropriate number of function points
    • The sum of function pointers for each component gives UFP (unadjusted function points)
FP-based Estimation (cont’d)

– Step 2: Compute the technical complexity factor (TCF)
  • Assign a value from 0 (“not present”) to 5 (“strong influence throughout”) to each of 14 factors such as transaction rates, portability (Figure 2)
  • Add the 14 numbers: This gives the total degree of influence (DI)
    – TCF = 0.65 + 0.01 × DI
    – The technical complexity factor (TCF) lies between 0.65 and 1.35

– Step 3. The number of function points (FP) is then given by
  • FP = UFP × TCF
FP-based Estimation (cont’d)

![Table]

<table>
<thead>
<tr>
<th>Component</th>
<th>Simple</th>
<th>Average</th>
<th>Complex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input item</td>
<td>3</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>Output item</td>
<td>4</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>Inquiry</td>
<td>3</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>Master file</td>
<td>7</td>
<td>10</td>
<td>15</td>
</tr>
<tr>
<td>Interface</td>
<td>5</td>
<td>7</td>
<td>10</td>
</tr>
</tbody>
</table>

1. Data communication
2. Distributed data processing
3. Performance criteria
4. Heavily utilized hardware
5. High transaction rates
6. Online data entry
7. End-user efficiency
8. Online updating
9. Complex computations
10. Reusability
11. Ease of installation
12. Ease of operation
13. Portability
14. Maintainability

Figure 1

Figure 2
FP-based Estimation (cont’d)

• The same product was coded both in assembler and in ADA and the results compared

<table>
<thead>
<tr>
<th></th>
<th>Assembler Version</th>
<th>Ada Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source code size</td>
<td>70 KDSI</td>
<td>25 KDSI</td>
</tr>
<tr>
<td>Development costs</td>
<td>$1,043,000</td>
<td>$590,000</td>
</tr>
<tr>
<td>KDSI per person-month</td>
<td>0.335</td>
<td>0.211</td>
</tr>
<tr>
<td>Cost per source statement</td>
<td>$14.90</td>
<td>$23.60</td>
</tr>
<tr>
<td>Function points per person-month</td>
<td>1.65</td>
<td>2.92</td>
</tr>
<tr>
<td>Cost per function point</td>
<td>$3,023</td>
<td>$1,170</td>
</tr>
</tbody>
</table>
Exercise Problems

• A target product has 7 simple inputs, 2 average input, and 10 complex inputs. There are 56 average output, 8 simple inquires, 12 average master files, and 17 complex interfaces. Determine the unadjusted function points (UFP).

• If the total degree of influence for the product of the question above is 49, determine the number of function points.
## Average LOC Per One Function Point

<table>
<thead>
<tr>
<th>Programming Languages</th>
<th>LOC/FP (average)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assembly Language</td>
<td>320</td>
</tr>
<tr>
<td>C</td>
<td>128</td>
</tr>
<tr>
<td>COBOL</td>
<td>105</td>
</tr>
<tr>
<td>FORTRAN</td>
<td>106</td>
</tr>
<tr>
<td>Pascal</td>
<td>90</td>
</tr>
<tr>
<td>C++</td>
<td>64</td>
</tr>
<tr>
<td>Ada95</td>
<td>53</td>
</tr>
<tr>
<td>Visual Basic</td>
<td>32</td>
</tr>
<tr>
<td>Smalltalk</td>
<td>22</td>
</tr>
<tr>
<td>Powerbuilder</td>
<td>16</td>
</tr>
<tr>
<td>SQL</td>
<td>12</td>
</tr>
</tbody>
</table>
COCOMO

• COnstructive COst MOdel
• Empirical model
  – Metrics such as LOC and FP are used as input to a model for determining product cost and duration
• Well documented, and supported by public domain and commercial tools; Widely used and evaluated
• Has a long pedigree from its first instantiation in 1981
  – COCOMO I (81)
  – COCOMO II
COCOMO (cont’d)

• Based on water fall process model
• The vast majority of software would be developed from the scratch
• There are three forms of the COCOMO
  – Basic COCOMO (macro estimation) which gives an initial rough estimate of man months and development time
  – Intermediate COCOMO which gives a more detailed estimate for small to medium sized projects
  – Detailed COCOMO (micro estimation) which gives a more detailed estimate for large projects.
COCOMO (cont’d)

• Effort = A * Size^B * M
  – Where A is coefficient
  – The exponent B reflects the increased effort required as the size of the product increases
  – The multiplier M is based on the project characteristics
# Intermediate COCOMO

<table>
<thead>
<tr>
<th>Organic mode (Simple)</th>
<th>Semi-detached mode (Moderate)</th>
<th>Embedded mode (Complex)</th>
</tr>
</thead>
</table>
| $\text{MM}_d = 3.2(\text{KLOC})^{1.05} \text{M}$  
(NE = 3.2(\text{KLOC})^{1.05}) | $\text{MM}_d = 3.0(\text{KLOC})^{1.12} \text{M}$  
(NE = 3.0(\text{KLOC})^{1.12}) | $\text{MM}_d = 2.8(\text{KLOC})^{1.20} \text{M}$  
(NE = 2.8(\text{KLOC})^{1.20}) |

- NE: Nominal effort (a rough estimate of the development effort using two parameters)
- $\text{MM}_d$: Man-month for estimated development effort
- M: 15 software development effort multipliers
- KLOC: number of thousands of line of code
Intermediate COCOMO (cont’d)

• Step 1. Estimate the length of the product in KLOC
• Step 2. Estimate the product development mode
  – Simple (organic, straightforward)
  – Moderate (medium sized, semidetached)
  – Complex (embedded)
• Step 3. Compute the **nominal effort**
• Step 4. Multiply the nominal value by **15 software development cost multipliers**
• Step 5. Estimate the calendar time (TDEV) in months required to complete a project
<table>
<thead>
<tr>
<th>Cost Drivers</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Attributes</td>
<td>Very Low</td>
</tr>
<tr>
<td>Required software reliability</td>
<td>0.75</td>
</tr>
<tr>
<td>Database size</td>
<td>0.94</td>
</tr>
<tr>
<td>Product complexity</td>
<td>0.70</td>
</tr>
<tr>
<td>Low</td>
<td>0.88</td>
</tr>
<tr>
<td>Nominal</td>
<td>1.00</td>
</tr>
<tr>
<td>High</td>
<td>1.15</td>
</tr>
<tr>
<td>Very High</td>
<td>1.40</td>
</tr>
<tr>
<td>Extra High</td>
<td>1.65</td>
</tr>
<tr>
<td>Computer Attributes</td>
<td>Very Low</td>
</tr>
<tr>
<td>Execution time constraint</td>
<td>1.00</td>
</tr>
<tr>
<td>Main storage constraint</td>
<td>1.00</td>
</tr>
<tr>
<td>Virtual machine volatility*</td>
<td>0.87</td>
</tr>
<tr>
<td>Computer turnaround time</td>
<td>0.87</td>
</tr>
<tr>
<td>Low</td>
<td>1.00</td>
</tr>
<tr>
<td>Nominal</td>
<td>1.00</td>
</tr>
<tr>
<td>High</td>
<td>1.15</td>
</tr>
<tr>
<td>Very High</td>
<td>1.30</td>
</tr>
<tr>
<td>Extra High</td>
<td>1.66</td>
</tr>
<tr>
<td>Personnel Attributes</td>
<td>Very Low</td>
</tr>
<tr>
<td>Analyst capabilities</td>
<td>1.46</td>
</tr>
<tr>
<td>Applications experience</td>
<td>1.29</td>
</tr>
<tr>
<td>Programmer capability</td>
<td>1.42</td>
</tr>
<tr>
<td>Virtual machine experience*</td>
<td>1.21</td>
</tr>
<tr>
<td>Programming language experience</td>
<td>1.14</td>
</tr>
<tr>
<td>Low</td>
<td>1.19</td>
</tr>
<tr>
<td>Nominal</td>
<td>1.00</td>
</tr>
<tr>
<td>High</td>
<td>0.86</td>
</tr>
<tr>
<td>Very High</td>
<td>0.71</td>
</tr>
<tr>
<td>Project Attributes</td>
<td>Very Low</td>
</tr>
<tr>
<td>Use of modern programming practices</td>
<td>1.24</td>
</tr>
<tr>
<td>Use of software tools</td>
<td>1.24</td>
</tr>
<tr>
<td>Required development schedule</td>
<td>1.23</td>
</tr>
<tr>
<td>Low</td>
<td>1.10</td>
</tr>
<tr>
<td>Nominal</td>
<td>1.00</td>
</tr>
<tr>
<td>High</td>
<td>0.91</td>
</tr>
<tr>
<td>Very High</td>
<td>0.82</td>
</tr>
<tr>
<td>Extra High</td>
<td>0.83</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*For a given software product, the underlying virtual machine is the complex of hardware and software (operating system, database management system) it calls on to accomplish its task.*

Figure 5. Intermediate COCOMO software development effort multipliers
Intermediate COCOMO – Example

- Example: Microprocessor-based communications processing software for electronic funds transfer network
- Step 1. Estimate the length of the product
  - 10,000 LOC (10 KLOC)
- Step 2. Estimate the product development mode
  - Complex (“embedded”) mode
- Step 3. Compute the nominal effort
  - Nominal effort = $2.8 \times (10)^{1.20} = 44$ man-months
Intermediate COCOMO
- Example (cont’d)

• Step 4. Multiply the nominal value by 15 software development cost multipliers (see table on the next slide)
  – Product of effort multipliers = 1.35
  – Estimated effort for project is therefore 1.35 * 44 = 59 person (man)-months
### Intermediate COCOMO - Example (cont’d)

<table>
<thead>
<tr>
<th>Cost Drivers</th>
<th>Situation</th>
<th>Rating</th>
<th>Effort Multiplier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required software reliability</td>
<td>Serious financial consequences of software fault</td>
<td>High</td>
<td>1.15</td>
</tr>
<tr>
<td>Data base size</td>
<td>20,000 bytes</td>
<td>Low</td>
<td>0.94</td>
</tr>
<tr>
<td>Product complexity</td>
<td>Communications processing</td>
<td>Very high</td>
<td>1.30</td>
</tr>
<tr>
<td>Execution time constraint</td>
<td>Will use 70% of available time</td>
<td>High</td>
<td>1.11</td>
</tr>
<tr>
<td>Main storage constraint</td>
<td>45K of 64K store (70%)</td>
<td>High</td>
<td>1.06</td>
</tr>
<tr>
<td>Virtual machine volatility</td>
<td>Based on commercial microprocessor hardware</td>
<td>Nominal</td>
<td>1.00</td>
</tr>
<tr>
<td>Computer turnaround time</td>
<td>2 hour average turnaround time</td>
<td>Nominal</td>
<td>1.00</td>
</tr>
<tr>
<td>Analyst capabilities</td>
<td>Good senior analysts</td>
<td>High</td>
<td>0.86</td>
</tr>
<tr>
<td>Applications experience</td>
<td>3 years</td>
<td>Nominal</td>
<td>1.00</td>
</tr>
<tr>
<td>Programmer capability</td>
<td>Good senior programmers</td>
<td>High</td>
<td>0.86</td>
</tr>
<tr>
<td>Virtual machine experience</td>
<td>6 months</td>
<td>Low</td>
<td>1.10</td>
</tr>
<tr>
<td>Programming language experience</td>
<td>12 months</td>
<td>Nominal</td>
<td>1.00</td>
</tr>
<tr>
<td>Use of modern programming practices</td>
<td>Most techniques in use over 1 year</td>
<td>High</td>
<td>0.91</td>
</tr>
<tr>
<td>Use of software tools</td>
<td>At basic minicomputer tool level</td>
<td>Low</td>
<td>1.10</td>
</tr>
<tr>
<td>Required development schedule</td>
<td>9 months</td>
<td>Nominal</td>
<td>1.00</td>
</tr>
</tbody>
</table>
Results of the Intermediate COCOMO

- COCOMO has been validated with respect to broad samples (63)
- COCOMO was the most accurate estimation method of its time
- Major problem
  - If the estimate of the number of lines of codes of the target product is incorrect, then everything is incorrect
COCOMO II

• 1995 extension to 1981 COCOMO that incorporates
  – Object orientation, Modern life-cycle models, Rapid prototyping,
    Fourth-generation languages, COTS software

• COCOMO II is far more complex than the first version
Exercise Problem

• You are in charge of developing a 76-KLOC embedded product that is nominal except that the database size is rated very high and the use of software tools is low. Using Intermediate COCOMO, what is the estimated effort in person (man)-months?