Software Project Planning and Management

• Usually guided by a Development Plan which specifies:
  – Project deliverables
  – Process and standards
  – Schedule and milestones
  – documentation and artifacts
  – Validation/Verification methods and criteria
  – Configuration management plan

Project Planning and Scheduling

• Key factors
  – Ability to estimate effort, cost, and schedule
  – Ability to track progress
  – Risk management
• Estimating Effort
  – Past Experience
  – Intuition
  – Cost Models
Planning and Scheduling

- Some initial observations
  - Requirements analysis and specification will typically consume 10 to 30% of a project’s total time (Boehm, *Software Eng. Economics*, 1981)
  - Rework due to defective requirements, design errors, and coding bugs typically consumes 40 to 50% of total project cost
  - Typical project experiences 25% change in requirements during development
  - Accuracy of time and effort estimation is poor at early stages
Effort and Cost Estimation

• Some Rough Rules-of-thumb:
  – Requirements analysis and design takes twice as long as coding
  – Validation also requires twice as long as coding
  – Maintenance costs may constitute 60% or more of total life-cycle costs

Approximate Life Cycle Costs
(from Schach, 1996)

- Maintenance: 67%
- Integration: 8%
- Module testing: 7%
- Module Coding: 5%
- Design: 6%
- Analysis & Specification: 6%
- Planning: 1%
Approximate Effort Breakdown

<table>
<thead>
<tr>
<th>Activity</th>
<th>Small Project (2,500 LOC)</th>
<th>Large Project (500 KLOC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture</td>
<td>10%</td>
<td>30%</td>
</tr>
<tr>
<td>Detailed design</td>
<td>20%</td>
<td>20%</td>
</tr>
<tr>
<td>Code/debug</td>
<td>25%</td>
<td>10%</td>
</tr>
<tr>
<td>Unit test</td>
<td>20%</td>
<td>5%</td>
</tr>
<tr>
<td>Integration</td>
<td>15%</td>
<td>20%</td>
</tr>
<tr>
<td>System Test</td>
<td>10%</td>
<td>15%</td>
</tr>
</tbody>
</table>

Software Cost Estimation Models

- Function Point Analysis
  - Provides an numerical estimate of functional size and complexity of a system.
    - Functional Size estimate derived from assessment of transactions and data functions of the system.
    - Complexity estimate derived from assessment of a standard set of system characteristics (complexity factors).
  - Cost Estimate based upon historical data
    - e.g. function points per person-month of effort
Function Point Estimation

- The number of function points in a program is derived from the number and complexity of:
  - **Inputs**--screens, forms, dialog boxes, etc.
  - **Outputs**--screens, reports, graphs, messages, etc.
  - **Inquiries**--e.g. database queries
  - **Logical internal files**
  - **External interface files**


---

Function Point Multipliers

<table>
<thead>
<tr>
<th>Program Characteristic</th>
<th>Low complexity</th>
<th>Medium complexity</th>
<th>High complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of inputs</td>
<td>x3</td>
<td>x4</td>
<td>x6</td>
</tr>
<tr>
<td>Number of outputs</td>
<td>x4</td>
<td>x5</td>
<td>x7</td>
</tr>
<tr>
<td>Inquiries</td>
<td>x3</td>
<td>x4</td>
<td>x6</td>
</tr>
<tr>
<td>Logical internal files</td>
<td>x7</td>
<td>x10</td>
<td>x15</td>
</tr>
<tr>
<td>External interface files</td>
<td>x5</td>
<td>x7</td>
<td>x10</td>
</tr>
</tbody>
</table>
### Function Point Estimation Example


<table>
<thead>
<tr>
<th>Function Points</th>
<th>Low complexity</th>
<th>Medium complexity</th>
<th>High complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of inputs</td>
<td>6x3=18</td>
<td>2x4=8</td>
<td>3x6=18</td>
</tr>
<tr>
<td>Number of outputs</td>
<td>7x4=28</td>
<td>7x5=35</td>
<td>0x7=0</td>
</tr>
<tr>
<td>Inquiries</td>
<td>0x3=0</td>
<td>2x4=8</td>
<td>4x6=24</td>
</tr>
<tr>
<td>Logical internal files</td>
<td>5x7=35</td>
<td>2x10=20</td>
<td>3x15=45</td>
</tr>
<tr>
<td>External interface files</td>
<td>9x5=45</td>
<td>0x7=0</td>
<td>2x10=20</td>
</tr>
<tr>
<td>Unadjusted function-point total</td>
<td>304</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Influence multiplier</td>
<td>1.15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adjusted function-point total</td>
<td>350</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Function Point Analysis--
The 14 Influence factors

- Data communications
- Distributed processing
- Performance
- Heavily used
- Transaction rate
- On-line data entry
- End-user efficiency
- On-line update
- Complex processing
- Reusability
- Installation Ease
- Operational ease
- Multiple Sites
- facilitate change

Function-point analysis--Influence Factors

- Each Influence factor is assigned a degree of influence on a scale of zero to five:
  - 0--Not present
  - 1--Incidental influence
  - 2--Moderate influence
  - 3--Significant influence
  - 5--Strong influence throughout

Function Point Analysis--Influence Factors

- An Example: Performance
  - 0--No special customer requirements
  - 1--Stated requirements, but no special actions required.
  - 2--Response time or throughput is critical during peak periods. Response time is next business day.
  - 3--Response time is critical during all business hours. Deadline requirements with interfacing systems are constraining
  - 4. Performance requirements are stringent enough to require performance analysis tasks during design
  - 5--Performance analysis tools required during design/implementation
Function Point Analysis--
Computation of Influence Multiplier

• The Influence Multiplier, IM, is computed as:
  \[ IM = (TDI * 0.01) + 0.65 \]
  
  Where TDI is the sum of the scores for the 14 influence factors.

  Note that \( 0.65 \leq IM \leq 1.35 \)

---

First-Order Schedule Estimation
Based upon Function Points


The development time, T, in months of a project with adjusted function point total, F, can be estimated as:

\[ T = F^e \]

Where \( e \) is determined from the following table:

<table>
<thead>
<tr>
<th>Kind of software</th>
<th>Best Practices</th>
<th>Average</th>
<th>Poor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Systems</td>
<td>0.43</td>
<td>0.45</td>
<td>0.48</td>
</tr>
<tr>
<td>Business</td>
<td>0.41</td>
<td>0.43</td>
<td>0.46</td>
</tr>
<tr>
<td>Shrink-wrap</td>
<td>0.39</td>
<td>0.42</td>
<td>0.45</td>
</tr>
</tbody>
</table>
First Order Schedule Estimation Example

Consider a shrink-wrap project with an adjusted function point total of 350:

<table>
<thead>
<tr>
<th>Type of Organization</th>
<th>Estimated development time (in months)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best in class</td>
<td>9.8 months</td>
</tr>
<tr>
<td>Average</td>
<td>11.7 months</td>
</tr>
<tr>
<td>Poor</td>
<td>14 months</td>
</tr>
</tbody>
</table>

Software Cost Estimation Models

- COCOMO (Constructive Cost Model)
  - Based upon data collected from past software projects.
  - Three COCOMO models:
    - Basic--Just relates estimated cost and effort to estimated program size (lines of code)
    - Intermediate--Includes assessment of subjective attributes (cost-drivers).
    - Advanced--Assesses the impact of cost-drivers on each step of the process (analysis, design, etc).
The COCOMO Model

• Basic Model:
  \[ E = aKLOC^b \]
  \[ D = cE^d \]
  
  Where:
  
  \( E \) is the estimated effort in person-months
  \( D \) is the development time in months
  \( KLOC \) is the estimated lines of code (in thousands)
  
  \( a, b, c, d \) are determined as follows:

  \[ \begin{array}{cccc}
  \text{Project Type} & a & b & c & d \\
  \text{organic} & 2.4 & 1.05 & 2.5 & 0.38 \\
  \text{semi-attached} & 3.0 & 1.12 & 2.5 & 0.35 \\
  \text{embedded} & 3.6 & 1.20 & 2.5 & 0.32 \\
  \end{array} \]

• Intermediate Model:
  \[ E = (eKLOC^f)EAF \]
  
  Where
  
  \( E \) is the project effort in person-months.
  \( KLOC \) is the project size in thousands of lines of code.
  
  \( e \) and \( f \) are determined as follows:

  \[ \begin{array}{cc}
  \text{Project Type} & e & f \\
  \text{organic} & 3.2 & 1.05 \\
  \text{semi-detached} & 3.0 & 1.12 \\
  \text{embedded} & 2.8 & 1.20 \\
  \end{array} \]

  EAF is an effort adjustment factor.
The COCOMO Model

- EAF Based Upon 15 Attributes:
  - Product attributes: required reliability, database size, product complexity.
  - System attributes: execution time constraints, storage constraints, virtual machine volatility, turn-around time
  - Personnel attributes: analyst capability, application experience, virtual machine experience, programmer capability, programming language experience
  - Project attributes: Modern programming practices, software tools, required development schedule

### COCOMO EAF Multipliers

<table>
<thead>
<tr>
<th>Attribute</th>
<th>VL</th>
<th>L</th>
<th>N</th>
<th>H</th>
<th>VH</th>
<th>EH</th>
</tr>
</thead>
<tbody>
<tr>
<td>RELY</td>
<td>0.75</td>
<td>0.88</td>
<td>1</td>
<td>1.15</td>
<td>1.4</td>
<td>-</td>
</tr>
<tr>
<td>DATA</td>
<td>-</td>
<td>0.94</td>
<td>1</td>
<td>1.08</td>
<td>1.16</td>
<td>-</td>
</tr>
<tr>
<td>CPLX</td>
<td>0.7</td>
<td>0.85</td>
<td>1</td>
<td>1.15</td>
<td>1.3</td>
<td>1.65</td>
</tr>
<tr>
<td>TIME</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1.11</td>
<td>1.3</td>
<td>1.66</td>
</tr>
<tr>
<td>STOR</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1.06</td>
<td>1.21</td>
<td>1.56</td>
</tr>
<tr>
<td>VIRT</td>
<td>-</td>
<td>0.87</td>
<td>1</td>
<td>1.15</td>
<td>1.3</td>
<td>-</td>
</tr>
<tr>
<td>TURN</td>
<td>-</td>
<td>0.87</td>
<td>1</td>
<td>1.07</td>
<td>1.15</td>
<td>-</td>
</tr>
<tr>
<td>ACAP</td>
<td>1.46</td>
<td>1.19</td>
<td>1</td>
<td>0.86</td>
<td>0.71</td>
<td>-</td>
</tr>
<tr>
<td>AEXP</td>
<td>1.29</td>
<td>1.13</td>
<td>1</td>
<td>0.91</td>
<td>0.82</td>
<td>-</td>
</tr>
<tr>
<td>PCAP</td>
<td>1.42</td>
<td>1.17</td>
<td>1</td>
<td>0.86</td>
<td>0.7</td>
<td>-</td>
</tr>
<tr>
<td>VEXP</td>
<td>1.21</td>
<td>1.1</td>
<td>1</td>
<td>0.9</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>LEXP</td>
<td>1.14</td>
<td>1.07</td>
<td>1</td>
<td>0.95</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MODP</td>
<td>1.24</td>
<td>1.1</td>
<td>1</td>
<td>0.91</td>
<td>0.82</td>
<td>-</td>
</tr>
<tr>
<td>TOOL</td>
<td>1.24</td>
<td>1.1</td>
<td>1</td>
<td>0.91</td>
<td>0.83</td>
<td>-</td>
</tr>
<tr>
<td>SCED</td>
<td>1.23</td>
<td>1.08</td>
<td>1</td>
<td>1.04</td>
<td>1.1</td>
<td>-</td>
</tr>
</tbody>
</table>
Software Cost Estimation Models--Some Caveats

• Models tend to be very sensitive to parameter adjustments.
• Models are not always accurate:
  – The Intermediate COCOMO model is accurate to within 25% in 63% to 75% of projects.
  – The mean error of COCOMO is approximately 20%.
  – The mean error for basic COCOMO is approximately 60%.

Software Project Risk Management

• Risk--unwanted Event with negative consequences.
• Risk characteristics:
  – Uncertainty--may or may not happen
  – loss--if it happens, there will be a cost
• Classes of of risks:
  – Project risks--threaten the development plan
  – Technical risks
  – Business risks
Risk Management Activities

• Risk Assessment
  – Risk identification
  – Risk analysis
  – risk prioritization

• Risk Control
  – Risk avoidance
  – Risk management planning
  – Risk resolution

Risk Assessment

• Risk Identification--identify risks to project schedule for each alternative course of action.

• Risk Analysis--Estimate the impact (cost) and likelihood of occurrence of each risk, for each alternative

• Risk Prioritization--Prioritize risks in order of exposure:
  – Exposure = (prob. of occurrence) x (impact)
Risk Control

- **Risk-management planning**--development plan specifically addresses risk management

- **Risk resolution**--develop strategies to mitigate risks--e.g.
  - find a way to avoid it all together
  - bring in a “hired gun”
  - develop contingency plans
  - decide to live with it--cost of doing business
  - identify a scapegoat (just kidding)

- **Risk Monitoring**--track risks and resolution efforts

---

Risk Management--An Example (from Pfleeger, 1998)

- Should regression testing be done on new releases of an incrementally developed product?

```
Do regression testing?
  Yes
    Find critical fault P(U0)=0.75, L(U0)=$.375M Exposure $0.375M Total Exposure $1.975M
    Miss critical fault P(U0)=0.05, L(U0)=$30M $1.5M $30M $31.5M
    No critical fault P(U0)=0.2, L(U0)=$.5M $0.10M

  No
    Find critical fault P(U0)=0.25, L(U0)=$.5M $0.125M
    Miss critical fault P(U0)=0.55, L(U0)=$30M $16.5M $16.725M
    No critical fault P(U0)=0.2, L(U0)=$.5M $0.10M
```
Some Primary Risk Areas

- Personnel shortfalls
- Unrealistic schedule and budget
- Incorrect specification
- Wrong interfaces
- “Gold-plating”
- Changing requirements
- Problems with externally-supplied tasks or components.
- Performance shortfalls
- Technical feasibility