Algorithmic estimation of software

- Basic cost model

\[ \text{Effort} = A \times \text{Size}^B \times m(X) \]

Size: Some measurement of the software size
A: Constant factor that depends on organizational practices
   Type of software
B: Usually lies between 1 and 1.5
X: Vector of cost factors
m: Adjustment multiplier

Cost models

\[ \text{Effort} = A \times \text{Size}^B \times m(X) \]

- Cost models
  - Define a way to determine the size
  - Define cost factors X
  - Provide defaults for parameters A, B, m (based on hundreds of projects)

Measuring size: Lines of code

- Software size can be measured in lines of source code
  - Most commonly used metric
- Difficult in early phases of the project (before design is known)
  - Reuse, make-or-buy decisions
- Influenced heavily by choice of programming language
- Should only be used indirectly

Function point analysis

- Size is estimated based on requirements

Inputs

- Inquiries
- Outputs

Function

- Internal files
- External files

Outputs

Inquiries

Functions

- Inputs
  - Forms, dialogs, messages, XML documents
- Outputs
  - Web pages, reports, graphs, messages, XML documents
- Inquiries (input/output combinations)
  - Simple web inputs, generally producing a single output
- Logical internal files (controlled by the program)
  - Tables, views or files in database
- External files (controlled by other programs)
  - Tables or files used from other systems or databases
Complexity of functions

<table>
<thead>
<tr>
<th>Factor</th>
<th>Simple</th>
<th>Average</th>
<th>Complex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inputs</td>
<td>3</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>Outputs</td>
<td>4</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>Inquiries</td>
<td>3</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>Ext. files</td>
<td>7</td>
<td>10</td>
<td>15</td>
</tr>
<tr>
<td>Int. files</td>
<td>5</td>
<td>7</td>
<td>10</td>
</tr>
</tbody>
</table>

- **Determine complexity of each function**
- **Weight each function according to complexity**

<table>
<thead>
<tr>
<th>Input</th>
<th>Simple</th>
<th>Average</th>
<th>Complex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data elements</td>
<td>1-5</td>
<td>6-10</td>
<td>&gt;10</td>
</tr>
<tr>
<td>Checking</td>
<td>Formal</td>
<td>Formal, logical</td>
<td>Formal, logical, requires DB access</td>
</tr>
</tbody>
</table>

Cost factors

<table>
<thead>
<tr>
<th>Data communications</th>
<th>Rate each element from 0 - 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed processing</td>
<td>0: no influence</td>
</tr>
<tr>
<td>Performance</td>
<td>1: insignificant influence</td>
</tr>
<tr>
<td>Heavy use</td>
<td>2: moderate influence</td>
</tr>
<tr>
<td>Transaction rate</td>
<td>3: average influence</td>
</tr>
<tr>
<td>Online data entry</td>
<td>4: significant influence</td>
</tr>
<tr>
<td>Complex interface</td>
<td>5: strong influence</td>
</tr>
</tbody>
</table>

Technical complexity factor

\[ TCF = 0.65 + 0.01 \times \text{sum} \]

Determining effort and size

- **Empirical value for effort**
  - Or use a table
- **Empirical value for size**
- **Huge differences in productivity**
  - Factor 10-20 between individual programmers
  - Factor 4 between companies

\[ \text{Effort} = \frac{\text{FP}^{1.4}}{150} \]

Function point computation

<table>
<thead>
<tr>
<th></th>
<th>Simple</th>
<th>Average</th>
<th>Complex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inputs</td>
<td>6 x 3  = 18</td>
<td>2 x 4 = 8</td>
<td>3 x 6 = 18</td>
</tr>
<tr>
<td>Outputs</td>
<td>7 x 4  = 28</td>
<td>7 x 5 = 35</td>
<td>0 x 7 = 0</td>
</tr>
<tr>
<td>Inquiries</td>
<td>0 x 3  = 0</td>
<td>2 x 4 = 8</td>
<td>4 x 6 = 24</td>
</tr>
<tr>
<td>Ext. files</td>
<td>9 x 5  = 45</td>
<td>0 x 7 = 0</td>
<td>2 x 10 = 20</td>
</tr>
<tr>
<td>Int. files</td>
<td>5 x 7  = 35</td>
<td>2 x 10 = 20</td>
<td>3 x 15 = 45</td>
</tr>
<tr>
<td>Unadjusted function points (UFP)</td>
<td>304</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Technical complexity factor (TCF)</td>
<td>1.15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adjusted function points</td>
<td>350</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Function point analysis: Discussion

**Pros**
- Based on requirements (instead of code size)
- Can be applied in early project phases
- Can be calibrated (for company, project type)
- Counting standards by "International Function Points User Group"
- Technology-independent

**Cons**
- Estimation of overall effort (not per phase)
- Tailored towards functional decomposition (rather than OO)
- Tailored towards information systems
- Needs calibration to produce reliable results

Observation about software size

- Consider a project that requires 10 Web pages, 15 reports, and 20 database tables
  - 315 function points, if each item is medium complexity
- How many lines of C code would it have?
  - About 32,000 lines
- What if you used Excel?
  - About 2,000 lines
- Why do you think there are so many spreadsheets out there?