Web Development: Estimating Quick-to-Market Software

Donald J. Reifer

15th International Forum on COCOMO and Software Estimation
Setting the Stage

- Business and government are rushing to web
- Many organizations are embracing quick-to-market paradigms (death march projects, etc.)
- These changes result in new way for applications to be engineered
# Web Versus Traditional Projects

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Traditional</th>
<th>Web</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary aim</td>
<td>Build products at minimum cost</td>
<td>Bring products to market quickly</td>
</tr>
<tr>
<td>Typical size</td>
<td>Medium to large (10 – 100+ engineers)</td>
<td>Small (3-10 engineers)</td>
</tr>
<tr>
<td>Timeline</td>
<td>12-18 months</td>
<td>3-6 months</td>
</tr>
<tr>
<td>Technology used</td>
<td>OOT, CASE tools, generators, C++, etc.</td>
<td>CBSE, frameworks, Java, multi-media, etc.</td>
</tr>
<tr>
<td>Process</td>
<td>CMM-based</td>
<td>Ad hoc, death marches</td>
</tr>
</tbody>
</table>
Reasons for Death Marches

• Startup mentality of fledging, entrepreneurial companies
• Naïve promises made by marketing, senior executives, inexperienced project leaders
• The “Marine Corp” mindset – real programmers don’t need sleep (feed them pizza and keep them coding)
• Intense pressure put on for quick return by investors
• Heat placed on firm due to globalization of marketplace
• Many unplanned and unexpected crises (turnover, etc.)

## More on Web Versus Traditional

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Traditional</th>
<th>Web</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Products</strong></td>
<td>Code-based systems, done in-house, mostly new, many external interfaces, often complex</td>
<td>Object-based systems, multi-media, done out-house, many reusable parts, few external interfaces, often simple</td>
</tr>
<tr>
<td><strong>Development staff</strong></td>
<td>Professional software engineers</td>
<td>Graphics designers, software engineers, etc.</td>
</tr>
<tr>
<td><strong>Estimating technology</strong></td>
<td><strong>Size</strong>: SLOC or fp</td>
<td><strong>Size</strong>: ??? (<strong>web objects</strong>)</td>
</tr>
<tr>
<td></td>
<td><strong>Resources</strong>: models or WBS estimate</td>
<td><strong>Resources</strong>: ad hoc or via WBS estimate</td>
</tr>
</tbody>
</table>
# Web-Based Estimating Challenges

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Traditional</th>
<th>Web</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimating process</td>
<td>Most use analogy and past experience</td>
<td>Most use WBS (inputs from developers)</td>
</tr>
<tr>
<td>Size estimation</td>
<td>Requirements drive estimates</td>
<td>Applications are built using frameworks or templates (90% solution)</td>
</tr>
<tr>
<td></td>
<td>• SLOC or function points used</td>
<td>• Web objects proposed (applets, video, etc.)</td>
</tr>
<tr>
<td></td>
<td>• Legacy, reuse, etc. handled as % new</td>
<td>• No agreed to metric</td>
</tr>
<tr>
<td>Schedule estimation</td>
<td>Most estimated using cube root relationship</td>
<td>Analysis show estimates too long (square root?)</td>
</tr>
</tbody>
</table>
## Web-Based Estimating Challenges

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Traditional</th>
<th>Web</th>
</tr>
</thead>
<tbody>
<tr>
<td>Effort estimation</td>
<td>Estimated using regression or via knowledge base</td>
<td>Estimated by breaking the job down into tasks and having responsible engineer estimate what is needed to do job</td>
</tr>
<tr>
<td></td>
<td>• Modified by cost drivers</td>
<td>• Little history</td>
</tr>
<tr>
<td></td>
<td>• Allocated via rules or Rayleigh curves</td>
<td>• No knowledge base</td>
</tr>
<tr>
<td>Calibration</td>
<td>Use past experience</td>
<td>Based on folklore</td>
</tr>
<tr>
<td>“What-if” analysis</td>
<td>Quantitative</td>
<td>Qualitative</td>
</tr>
</tbody>
</table>
New Size Metrics

Traditional

• Source lines of code
  – Non-visual, no motion, etc.
• Function points
  – Conventions don’t exist
• Function point extensions
  (3D, predictive object points, etc.)
  – Requirements driven, often
    OO-based, non-visual
• UML metrics
  – Same as extensions

Non-Traditional

• Multi-media files
  – Graphics - Text
  – Audio - Motion
  – Pictures (JPEG, etc.)
• Web components
  – Applets, Agents, etc.
• Fine-grained building blocks
  – DCOM, CORBA, etc.
• Coarse-grained components
  – Shopping carts, etc.
Time for Some Heresy

Halstead’s Equation for Volume

\[ V = N \log_2 (n) = (N_1 + N_2) \log_2 (n_1 + n_2) \]

Where:
- \( N \) = number of occurrences of Operands/Operators
- \( n \) = number of distinct Operands/Operators
- \( N_1 \) = total occurrences of Operand estimator
- \( N_2 \) = total occurrences of Operator estimator
- \( n_1 \) = number of unique Operands estimator
- \( n_2 \) = number of unique Operator estimator
- \( V \) = volume of work involved represented as we objects
## Web-Based Size Predictors

<table>
<thead>
<tr>
<th>Web Object Predictors</th>
<th>Example Operands</th>
<th>Example Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td># of building blocks</td>
<td>Widgets, fined-grained components (ActiveX, DCOM, OLE, etc.), etc.</td>
<td>Create, apply, call, dispatch, interface, terminate, etc.</td>
</tr>
<tr>
<td># of COTS components</td>
<td>COTS programs, library routines, web objects (cart), etc.</td>
<td>Transform, access, bind, generate, interface, etc.</td>
</tr>
<tr>
<td># of multi-media files</td>
<td>Text, video, audio, etc. (not graphics files)</td>
<td>Create, cut, paste, clear, edit, etc.</td>
</tr>
<tr>
<td># of application or object points (or others proposed)</td>
<td># server tables, # client tables, # states, # entities, attributes, etc.</td>
<td>Transform, access, modify, instantiate, generate, etc.</td>
</tr>
</tbody>
</table>
## More Web-Based Predictors

<table>
<thead>
<tr>
<th>Web Object Predictors</th>
<th>Example Operands</th>
<th>Example Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td># of web components</td>
<td>Applets, agents, guards, etc.</td>
<td>Create, schedule, dispatch, etc.</td>
</tr>
<tr>
<td># of xml, sgml, html and query lines</td>
<td>Lines including links to data attributes</td>
<td>Create, call, browse, link, traversal, etc.</td>
</tr>
<tr>
<td># of graphics files</td>
<td>Templates, pictures, images, etc.</td>
<td>Apply, align, import, export, insert, etc.</td>
</tr>
<tr>
<td># of scripts (visual language, audio, motion, etc.)</td>
<td>Macros, containers, etc.</td>
<td>Create, store, edit, distribute, serialize, generate, etc.</td>
</tr>
</tbody>
</table>
## Web Object Calculation Worksheet

<table>
<thead>
<tr>
<th>Web Object Predictors</th>
<th>Low</th>
<th>Medium</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td># of building blocks</td>
<td>1</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td># of components</td>
<td>2</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td># of multi-media files</td>
<td>1</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td># of application or object points</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td># of web components</td>
<td>2</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td># of xml, sgml, html &amp; query lines</td>
<td>3</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td># of graphics files</td>
<td>2</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td># of scripts</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Other</td>
<td>2</td>
<td>4</td>
<td>6</td>
</tr>
</tbody>
</table>

* Assume weights have already been applied
Interesting Findings

• Size of application is extremely sensitive to counting conventions
  – Must use one set of conventions consistently
  – We use IFPUG and weighted methods per class
• Fit data using web objects, not SLOC or other metric
• Weighted goodness of fit significantly better than for unweighted predictors
  – Based upon 46 completed projects whose size ranged from 20 to 100 web objects
• Currently defining weightings more precisely
Estimation via COCOMO II

• Effort estimation via Early Design model
  – Had co-linearity in the data which was noisy

• Simplified the effort estimation model further to eliminate co-linearity/fit the data
  – Resulting model has eight drivers and fixed power laws based upon four types of web applications researched

• Schedule estimation via Early Design model
  – Estimate was way off so used square instead of cube root relationship (based on scaling rules)
WEBMO – Web Estimation Model

Effort = \( A \sum_{i=1}^{8} C_i \cdot (\text{Size})^{P1} \)  
Duration = \( B(\text{Effort})^{P2} \)

Where:  
- \( A \) = effort coefficient  
- \( B \) = duration coefficient  
- \( C_i \) = cost drivers  
- \( P1 \) = effort power law  
- \( P2 \) = duration power law
Cost Drivers

• Kept Early Design model drivers:
  – **RCPX** – Product Reliability & Complexity
    • RELY, DATA, CPLX, DOCU
  – **PDIF** – Platform Difficulty
    • TIME, STOR, PVOL
  – **PERS** – Personnel capability
    • ACAP, PCAP, PCON
  – **PREX** – Personnel experience
    • APEX, PLEX, LTEX
  – **FCIL** – Facilities
    • TOOL, SITE
  – **SCED** – Required development schedule

• Eliminated one driver
  – **RUSE** – not significant

• Could not calibrate with scale factors; used two new cost drivers to handle
  – **TEAM** - Teamwork
  – **PEFF** – Process Efficiency
## WEBMO Parameters/Power Laws

<table>
<thead>
<tr>
<th>Category</th>
<th>A</th>
<th>B</th>
<th>P1</th>
<th>P2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web based electronic commerce</td>
<td>2.3</td>
<td>2.0</td>
<td>1.05</td>
<td>*</td>
</tr>
<tr>
<td>Financial/trading application</td>
<td>2.7</td>
<td>2.2</td>
<td>1.05</td>
<td>*</td>
</tr>
<tr>
<td>B-to-B applications</td>
<td>2.0</td>
<td>1.5</td>
<td>1.00</td>
<td>*</td>
</tr>
<tr>
<td>Web based information utilities</td>
<td>2.1</td>
<td>2.0</td>
<td>1.00</td>
<td>*</td>
</tr>
</tbody>
</table>

* Either 0.5 or 0.33 based upon scaling ( > 40 web objects)
## Driver Rating Scales

<table>
<thead>
<tr>
<th>Driver</th>
<th>VL</th>
<th>L</th>
<th>N</th>
<th>H</th>
<th>VH</th>
</tr>
</thead>
</table>
| **RCPX** | - Client side  
- No distribution  
- Invocation  
- Simple math  
- Simple I/O  
- Limited data  
- Reliability not a factor | - Client/server  
- Ltd distribution  
- Adaptation  
- Standard math  
- File management  
- Some files  
- Easy to recover from losses | - Client/server  
- Fully distributed  
- Integration  
- Statistics  
- DBMS  
- Databases  
- Moderate recovery goals | - Configurable web  
- Wide distribution  
- Synchronization  
- Math intensive  
- Distributed DB  
- Virtual database  
- High financial loss due to error | - Seamless web  
- Full distribution  
- Collaboration  
- Soft real-time  
- Persistent DB  
- Virtual database  
- Errors cause risk to human life |
| **PDIF** | - Rare changes to platform  
- Speedy net  
- Best possible connectivity  
- No computer resource limitations | - Few changes to platform  
- Fast net service  
- Rare lose of connectivity  
- Few computer resource limitations | - Platform stable  
- Acceptable net performance  
- Acceptable connectivity  
- Must watch use of computer resources | - Frequent changes to platform  
- Slow network performance  
- Poor connectivity  
- Lack of computer resources causing problems | - Platform not stable  
- Unacceptable net performance  
- Unacceptable connectivity  
- Timing and storage impacts |
## Driver Rating Scales

<table>
<thead>
<tr>
<th>Driver</th>
<th>VL</th>
<th>L</th>
<th>N</th>
<th>H</th>
<th>VH</th>
</tr>
</thead>
<tbody>
<tr>
<td>PERS</td>
<td>- 15&lt;sup&gt;th&lt;/sup&gt; percentile - Major delays due to turnover and attrition</td>
<td>- 35&lt;sup&gt;th&lt;/sup&gt; percentile - Minor delays due to turnover and attrition</td>
<td>- 55&lt;sup&gt;th&lt;/sup&gt; percentile - Few delays due to turnover and attrition</td>
<td>- 75&lt;sup&gt;th&lt;/sup&gt; percentile - Infrequent delays due to turnover and attrition</td>
<td>- 90&lt;sup&gt;th&lt;/sup&gt; percentile - No delays due to turnover and attrition</td>
</tr>
<tr>
<td>PREX</td>
<td>- ≤ 2 months average applications, platform, language and tool experience</td>
<td>-≤ 6 months average applications, platform, language and tool experience</td>
<td>-≤ 1 year average applications, platform, language and tool experience</td>
<td>-≤ 3 years average applications, platform, language and tool experience</td>
<td>-≤ 6 years average applications, platform, language and tool experience</td>
</tr>
</tbody>
</table>
## Driver Rating Scales

<table>
<thead>
<tr>
<th>Driver</th>
<th>VL</th>
<th>L</th>
<th>N</th>
<th>H</th>
<th>VH</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SCED</strong></td>
<td>- Must shorten -75 percent of nominal</td>
<td>- Must shorten -85 percent of nominal</td>
<td>- Keep as is - Nominal</td>
<td>- Can relax -130 percent of nominal</td>
<td>- Can extend -160 percent of nominal</td>
</tr>
<tr>
<td><strong>REUSE</strong></td>
<td>Not used</td>
<td>Not used</td>
<td>Note used</td>
<td>Not used</td>
<td>Not used</td>
</tr>
</tbody>
</table>
## Driver Rating Scales

<table>
<thead>
<tr>
<th>Driver</th>
<th>VL</th>
<th>L</th>
<th>N</th>
<th>H</th>
<th>VH</th>
</tr>
</thead>
</table>
| TEAM   | - No shared vision  
- Stakeholders do not work to meet each others objectives  
- Teamwork limited | - Little shared vision  
- Stakeholders talk and build some respect each other’s goals  
- Some teamwork | - Some shared vision  
- Stakeholders pull together and work towards joint goals  
- Basic teamwork | - Considerable shared vision  
- Stakeholders respect each others goals and collaborate on the effort  
- Integrated teams | - Extensive shared vision  
- Stakeholders pull together from day 1 and mount a goal-directed effort  
- Seamless teams |
| PEFF   | - Totally ad hoc, confused process  
- Reliance on hero’s to get job done | - Project-based processes (little sharing with others)  
- Reliance on management leadership to meet goals | - Streamlined process tailored for fast-paced jobs and teams  
- Reliance on process for guidance | - Efficient process matched to getting the job done  
- Process is the way engineers do their work | - Effective process used to generate quality products on-time & budget  
- Everyone uses and believes in the process |
## Cost Driver Values

<table>
<thead>
<tr>
<th>Driver</th>
<th>VL</th>
<th>L</th>
<th>N</th>
<th>H</th>
<th>VH</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCPX</td>
<td>0.63</td>
<td>0.85</td>
<td>1.00</td>
<td>1.30</td>
<td>1.67</td>
</tr>
<tr>
<td>PDIF*</td>
<td>0.75</td>
<td>0.87</td>
<td>1.00</td>
<td>1.21</td>
<td>1.41</td>
</tr>
<tr>
<td>PERS</td>
<td>1.55</td>
<td>1.35</td>
<td>1.00</td>
<td>0.75</td>
<td>0.58</td>
</tr>
<tr>
<td>PREX</td>
<td>1.35</td>
<td>1.19</td>
<td>1.00</td>
<td>0.87</td>
<td>0.71</td>
</tr>
<tr>
<td>FCIL</td>
<td>1.35</td>
<td>1.13</td>
<td>1.00</td>
<td>0.85</td>
<td>0.68</td>
</tr>
<tr>
<td>SCED*</td>
<td>1.35</td>
<td>1.15</td>
<td>1.00</td>
<td>1.05</td>
<td>1.10</td>
</tr>
<tr>
<td>TEAM</td>
<td>1.45</td>
<td>1.31</td>
<td>1.00</td>
<td>0.75</td>
<td>0.62</td>
</tr>
<tr>
<td>PEFF</td>
<td>1.35</td>
<td>1.20</td>
<td>1.00</td>
<td>0.85</td>
<td>0.65</td>
</tr>
</tbody>
</table>

* Significant differences from original model
Current Status

• Goal is to help clients better estimate resources needed for web development
• In the process of learning, validating and extending the model
  – Current database has 46 completed projects (need more)
  – Using linear regression to fit the actuals to estimates
  – Data is noisy (no timecards, 80 hour weeks, etc.)
• This is not a USC sponsored project
  – Our intent is to develop a model for use in our consulting business
  – We are funded by several clients and have a proposal in to NSF for a grant
Next Steps

- Analyze more data
- Determine if we can use Early Design model or CORADMO out of the box
- Improve predictive accuracy
  - Current accuracy is estimates are within 30 percent of actuals 60 percent of the time
  - Better than winging it, but not good enough
- Have fun, help clients and make money
  - Reifer Consultants is a for profit firm
Final Thoughts

• The rush to the Internet represents a modern day gold rush
• Those who provision the prospectors will make the fortunes
• We want to provide them with tools they need to figure out what it will take for them to be successful
Acknowledgements

• USC Center for Software Engineering
  – Barry Boehm and the COCOMO team
• Galorath Inc.
  – Lee Fischman and Dan Galorath
• Marotz
  – Bill Roetzheim
• The Multimedia House of Quality
  – Adrian Cowderoy
• Those nameless individuals who worked hard to supply the data