Improving Software Project Management Through System Dynamic Modeling

Doug Sycamore
Motorola GSTG
8220 E. Roosevelt, MS R2208
Scottsdale, Arizona 85252
doug.sycamore@email.mot.com
http://www.eas.asu.edu/~sdm
(602) 441-1331
Creating schedules and tracking progress are two activities software managers perform.

These activities become more complex with larger projects.

Wrong decisions implemented into a project could have disastrous results, reducing the probability of success.

Understanding the many system dynamics and feedback loops associated with a project, especially a large project, is a formidable task.
There are plenty of Commercial Off The Shelf (COTS) project management tools available.

Many perform basic features such as scheduling, budget tracking, man-hour reporting, resource allocation, and capital expenditure tracking.

None of the tools available take into consideration how system dynamic variables affect a project.
The following thesis is proposed:

Using the "iThink" tool available from High Performance Systems, a new tool can be developed to help a software manager plan, track, and predict the outcome of a software project better than the more traditional approaches by incorporating system dynamic feedback loops into the tool.
"Staying the Course With the Project Control Panel"
by Susan Mann Beagley

4 Status Indicators:
- Schedule
- Budget
- Percent Complete
- Quality

4 Controllable Parameters:
- Resources
- Work Sequence
- Project Scope
- Productivity
INCREMENTAL DESIGN APPROACH
### Rate of productivity of each level of Engineer.

<table>
<thead>
<tr>
<th>Level</th>
<th>Productivity</th>
<th>0 - Don't Include; 1 - Include</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eng 1</td>
<td>0.8</td>
<td>Include Schedule Press = 1.0</td>
</tr>
<tr>
<td>Eng 2</td>
<td>1.0</td>
<td>Include Comm Overhead = 1.0</td>
</tr>
<tr>
<td>Eng 3</td>
<td>1.2</td>
<td>Include Quality = 1.0</td>
</tr>
</tbody>
</table>

### Schedule Pressure:

<table>
<thead>
<tr>
<th>0 - Don't Include; 1 - Include</th>
</tr>
</thead>
<tbody>
<tr>
<td>Include Schedule Press = 1.0</td>
</tr>
</tbody>
</table>

### Communication Overhead:

<table>
<thead>
<tr>
<th>0 - Don't Include; 1 - Include</th>
</tr>
</thead>
<tbody>
<tr>
<td>Include Comm Overhead = 1.0</td>
</tr>
</tbody>
</table>

### Quality:

Quality will automatically generate defects that are generated by the engineers. These defects are converted to hours to fix and filtered back into the model. Quality must be included in order to include the reviews and/or rework %

#### 0 - Don't Include; 1 - Include

<table>
<thead>
<tr>
<th>Include Quality = 1.0</th>
</tr>
</thead>
</table>

### Rework %:

Quality must also be included for this to work. Percent of time allocated for rework.

#### 0 - Don't Include; 1 - Include

<table>
<thead>
<tr>
<th>Entered Rework = 0.2</th>
</tr>
</thead>
</table>

### Include Reviews:

Quality must also be included for this to work. When included the model will use 10% of the design and code phase for doing reviews.

#### 0 - Don't Include; 1 - Include

| Include Review = 1.0 |

### Errors per hr:

<table>
<thead>
<tr>
<th>0.025</th>
</tr>
</thead>
</table>

| 0.10  |
Increment #3

The scope of the work effort for each phase.

<table>
<thead>
<tr>
<th>Phase</th>
<th>Effort</th>
</tr>
</thead>
<tbody>
<tr>
<td>Init Design 3</td>
<td>2500</td>
</tr>
<tr>
<td>Init Code 3</td>
<td>2500</td>
</tr>
<tr>
<td>Init Int 3</td>
<td>2500</td>
</tr>
</tbody>
</table>

This is the level of messaging that will appear.

- 0 - No messages
- 1 - Message Upon Completion of Task
- 2 - 1 + Msg after Code Phase Completes
- 3 - 2 + Msg after Design Phase Completes

- Incr 3 Msg Level = 1

Number of Engineers Separated by the level expertise. Level 1 (less) -> Level 3 (Highest)

<table>
<thead>
<tr>
<th>Engineer</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eng 1 Incr 3</td>
<td>0</td>
</tr>
<tr>
<td>Eng 2 Incr 3</td>
<td>0</td>
</tr>
<tr>
<td>Eng 3 Incr 3</td>
<td>0</td>
</tr>
</tbody>
</table>

Task #3 Dependencies

- 0 - None; 1 - Design 2 - Code; 3 - Increment

- Incr 3 Dependent on 1 = 0
- Incr 3 Dependent on 2 = 0
Percent Complete Danger Zone

1: Danger Zone  2: Schedule % Comp

2% offset

20% offset
Scenario #1 - Does *Not* Include Communication Overhead
Scenario #2 - Does Include Communication Overhead.
The validate approach.

- Identify a group of well qualified, reputable experts with good credentials that are familiar with software engineering principles.

- Demo the tool with canned data to these experts, explaining the implementation and characteristics of the tool.

- Give each evaluator the opportunity to make modifications to the canned data.

- Give each evaluator the opportunity to run their own project supplied data.

- Ask each evaluator to fill out and return an evaluation summary form.
The first is to enhance the depth of the tool. (For example, increase the number of increments or better model the current feedback loops.)

The second area is to improve the tool by adding breadth. (For example, model the effects of various testing techniques on a project or adding requirements analysis to the model.)

The third area is identifying other useful applications for the tool. (For example, using it for training software managers.)