DOING THE MATH

How to Estimate and Manage Results

Presented to the IEEE-CNSV Consultants’ Network of Silicon Valley

John Carter

December 15, 2009
Presentation Outline

• The limits of planning
• Identification of key initiatives
  – Analysis of a process
  – Selection of an Improvement Initiative
• Inch Wide, Mile Deep: Minimizing Technical & Organizational Complexity
  – Predictive Metrics
• Estimation and management for change initiatives
  – Field analysis of improvement programs
  – Creation of the “Half Life” Concept
  – Example for an MRD (Marketing Requirements Document Initiative)
• Case Study: Tracking Progress over Time
• Bonus: Estimation based on no information
  – Copernican Principle
The Limits of Planning

- Even the slightest change in one attribute can cause uncertainly large changes in any other attribute.
- You get more control over estimation by learning from evolutionary early and frequent result deliveries, than you will if you try to estimate in advance for a whole large project.
- Any method which gives you early feedback and correction of reality is more likely to give you control over the final result than big bang methods.
- Data from past projects might be useful, but it can never be as useful to you as current data from your present project.

Estimation methods alone will not change a result which is off the track. Active correction must be a part of your methodology.

Overview: How to Determine Root Causes

WHAT ARE THE ISSUES IMPACTING RELEASE TIME AND QUALITY FOR THE PROJECT?

- Definition
- Design
- Integration
- Validation
- Testing
- Release

Timeline Analysis

Event 1

Event 2

Event 3

Root Cause Analysis

Root Cause Synthesis

Identify root causes with short half life and high impact to avoid boiling ocean
Example Project History Synthesis

A project without a road map and a destination gets you to an unknown place at an unknown time.

There is no commitment to follow a product development process.

There was no definition of customer or product requirements documented for the engineering team.

Product requirements were not clearly defined.

50% of the customer requirements were not known by the design team during implementation.

Product requirements were never frozen (not even frozen, thawed, frozen...).

There was no product definition.

There was no customer or product acceptance criteria documented for the engineering team.

No common software development process was used.

No formal testing process was allowed.

No defect detection was done on the software design.

The application interface was not formally reviewed prior to implementation.

There was no software design review for the Venus 2 module.

There was no method for communicating product knowledge.

This was a customer-driven generic product.

Only 1 customer was used to gather requirements for the product.

Man-hour estimate was not changed after customer requirements changed.

No time was scheduled to provide documentation to application writers.

We did not provide team members with baseline products S/W knowledge.

As new people were added to the team, project leadership did not pass on design or functional methods formally.

There was insufficient H/W available to S/W.

During integration we had one test setup in lieu of three needed.

Sharing resources with hardware engineers limited software engineer's test time by 20% of what was needed.

The delivery date for hardware left only 1.5 months for H/W - S/W integration and testing.

The application interface was not formally reviewed prior to implementation.

Customer acceptance testing was not scheduled.

Addition of customer demo to compressed schedule directly reduced testing time.

1 hour of testing was done when 2 weeks were required.

The Venus 2 software architecture is not documented outside of the code.

We did not test the system prior to shipment.

There was a lack of commitment to testing.

What are the issues impacting release time and quality for the project?
What are “Predictive Metrics”?

Definition: The measurement of a key driver or initiative, which if executed correctly, will lead to the achievement of overall goals

- Track Progress to plan
- Key Drivers and Milestones – Not just $
- Indicators of the outcome
- Simple, lightweight and easy to deploy

- Benefits are...
  - Prevents bad outcomes
  - Focuses management on key drivers
  - Saves time in preparation for project reviews
  - Higher quality meetings as a result of capturing history
Half Life Principles

- How fast does one expect to change?
- It depends on many factors including urgency, simplicity, number of dependencies (people, process, or technology) and the organizational scope
- Art Schneiderman, VP of Quality at Analog Devices performed a survey of nearly 100 improvement initiatives

Reference: “Setting Quality Goals” Quality Progress, Arthur Schneiderman
### Example Workflow Improvement Initiative Half Lives

A table showing the half lives in months for various areas of workflow improvement initiatives, along with the number of cycles.

<table>
<thead>
<tr>
<th>Area</th>
<th>Half Life</th>
<th>Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Days late in Delivery</td>
<td>0.8</td>
<td>7.6</td>
</tr>
<tr>
<td>Errors in Purchase Orders</td>
<td>2.3</td>
<td>1.5</td>
</tr>
<tr>
<td>Late Deliveries to Customers</td>
<td>3</td>
<td>2.7</td>
</tr>
<tr>
<td>Downtime</td>
<td>4.5</td>
<td>1.3</td>
</tr>
<tr>
<td>In-Process Defect Rate</td>
<td>5.3</td>
<td>1.1</td>
</tr>
<tr>
<td>Defects per Unit</td>
<td>7.6</td>
<td>4.6</td>
</tr>
<tr>
<td>Customer Defect Reporting</td>
<td>10.1</td>
<td>7.1</td>
</tr>
<tr>
<td>Defects</td>
<td>10.4</td>
<td>5.2</td>
</tr>
<tr>
<td>Software Documentation Errors</td>
<td>10.5</td>
<td>1.2</td>
</tr>
<tr>
<td>Defects on Arrival</td>
<td>16.9</td>
<td>2</td>
</tr>
<tr>
<td>Field Failure Rate</td>
<td>20.3</td>
<td>1.3</td>
</tr>
<tr>
<td>Software Error Rates in Development</td>
<td>24</td>
<td>2</td>
</tr>
<tr>
<td>Software Execution Errors</td>
<td>29.9</td>
<td>0.4</td>
</tr>
<tr>
<td>Reuse</td>
<td>30</td>
<td>1.6</td>
</tr>
<tr>
<td>Development Staff Efficiency Improve</td>
<td>42</td>
<td>1.1</td>
</tr>
<tr>
<td>Product Development Cycle Time</td>
<td>55.3</td>
<td>1.1</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>17.1</strong></td>
<td><strong>2.6</strong></td>
</tr>
</tbody>
</table>

Reference: Analog Devices: The Half Life System, HBS Case 9-190-061

- Half Life is shown in months, and these examples are from reports from industry and from Analog Devices study (subset from prior chart)
- Chart is a graphical representation from table
Rule Of Thumb For Improvement Goals

- Use the chart below to estimate how many months it will take to increase the frequency of use by a factor of two from the current level – this is “half life”
- Use guidelines when no other means exists to determine rate of improvement

<table>
<thead>
<tr>
<th>Project Type</th>
<th>Examples</th>
<th>Typical Half Life</th>
<th>Minimum Half Life</th>
<th>Maximum Half Life</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uni-Functional</td>
<td>Marketing Requirements Document</td>
<td>3</td>
<td>0-1</td>
<td>6</td>
</tr>
<tr>
<td>Cross-Functional</td>
<td>New Product Cycle Time</td>
<td>9</td>
<td>6-18</td>
<td>12-48</td>
</tr>
<tr>
<td>Multi-Entity</td>
<td>Vendor Quality</td>
<td>18</td>
<td>12-18</td>
<td>24-48</td>
</tr>
</tbody>
</table>

Examples of Predictive Metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>Half Life Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase Review Nomenclature</td>
<td>1</td>
</tr>
<tr>
<td>MRD</td>
<td>2</td>
</tr>
<tr>
<td>Programs with Program Plans</td>
<td>2</td>
</tr>
<tr>
<td>Teams with Charters</td>
<td>2</td>
</tr>
<tr>
<td>Software Adoption</td>
<td>3</td>
</tr>
<tr>
<td>Unit Testing Adoption</td>
<td>3</td>
</tr>
<tr>
<td>Formal Inspections</td>
<td>6</td>
</tr>
<tr>
<td>Process rollout to divisions</td>
<td>6</td>
</tr>
</tbody>
</table>

\[ y = 1 - e^{-\ln(2)t/T} \]
### Half-life Plot For Marketing Requirement Documents (MRDs)

- **Precise Definition**
  - Out of the total projects listed as being in investigation phase in the weekly/monthly updates, how many (%) have an MRD. A project is considered to have an MRD if the team identifies a specific document as fulfilling that function, regardless of its title.

- **Sample Baseline**
  - Unmeasured, and difficult to measure without identifying project phases. Based on the sampling from “slotting exercises,” roughly 10% of the projects are likely to have MRDs at the start. The goal is 90%

- **Half life**
  - 3 months (Example: Time to go from initial value of 10% to 1/2 the gap of 90% that value, or 50%)

- **Baseline value**
  - 10% of projects have MRDs currently

\[
y = 1 - \left( (y_o - y_{min}) e^{\frac{\ln(2)t}{T_1/2} + y_{min}} \right)
\]

\[
y_o = 0.9, \quad y_{min} = 0.1
\]

\[=1-((Yo-Ymin)\times\text{EXP}(-1\times a\times E37/T2)+Ymin) \quad \text{[From Excel where } a=\ln(2) \text{ and E37 is a cell reference for time]}\]
Any method which gives you early feedback and correction of reality is more likely to give you control over the final result than big bang methods.

For Background see: “New Product Development: PM Network, March 1994”
Estimation based on little (or no) information

- In Cosmology the Copernican Principle, named after Nicolas Copernicus, states the Earth is not located at the center of the universe.
- The time analog to the center of the solar system, is we are not observing a phenomenon at a special time.
- There is a 50% chance you are observing sometime during the middle two quarters of its existence.
- There is a 95% chance you are not making your observation during the short end (2.5%) or the long end (2.5%).
- To get the 95% confidence range of existence, divide and multiply current life by 39.

Where ‘point estimates’ have been useful:
- Lifetime of partner relationship
- Lifetime of a startup
- Lifetime of a vendor
- Lifetime of a business

For Background see: “A Survival Imperative for Space Colonization” John Tierney, NYT July 17, 2007