Economic Governance of Software Delivery

Walker Royce
Chief Software Economist
Software Delivery is an Economic Discipline

Transforming for Improved Predictability and Speed

Engineering Governance

- Distinct development phase
- Distinct handoff to maintenance
- Requirements-design-code-test Sequence of activities
- Measure activities and artifacts
- Requirements/Technology led

Deterministic Engineering Discipline

Economic Governance

- Continuously evolving systems
- No distinct boundary between development and maintenance
- Sequence of released capabilities with ever increasing value
- Measure progress/quality trends in change managed code and test base
- Business value and outcome led

Probabilistic Economic Discipline
Deliver business critical software product release within 12 months

ESTIMATE: 11 Months

BUSINESS NEED

CHALLENGE

SCHEDULE
Late Scrap and Rework
INSIGHT

Mean estimate = 11 MONTHS

Area under curve = Probability of delivering in 11 to 12 MONTHS

Probability of delivering in 11 to 12 MONTHS
There is roughly a 50% chance of making the date.
Move out the date to improve likelihood of shipping?

OPTION 1

Absolutely NOT!
Decrease time estimate by Sacrificing quality or content?

**OPTION 2**

**PROBABILITY**

**SCHEDULE**

95%
Reduce the variance to improve likelihood of shipping.
Uncertainty in Plans, Scope and Design

Measure validated learning
Reduce the variance

INITIAL TARGET
Uncertainty in stakeholder satisfaction
Best/Worst practices in agile measurement

Integrate
- Plans/management
  - Plan for integration to precede unit testing

Collaborate
- Progress measures
  - Quantify progress trends from the integrated code and test base

Optimize
- Quality measures
  - Quantify cost-of-change trends to demonstrate true agility

Avoid false precision in plans and requirements
Don’t rely on subjective and speculative measures
Don’t attack the easy things first
Measured Improvement: Progress Measures

Conventional Engineering Governance

Modern Economic Governance

Planning Progress

Requirements
Design
Coding
Test and Release

Early Releases
Test Releases

Technical Progress

Late scrap and rework

Progressions and digressions

Economic Progress

Requirements
Design
Coding
Test and Release
Measured Improvement: Quality Measures

Conventional Engineering Governance

<table>
<thead>
<tr>
<th>Maturity</th>
<th>Defect Trend</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit Test</td>
<td>Integration</td>
</tr>
<tr>
<td></td>
<td>Operation</td>
</tr>
</tbody>
</table>

Modern Economic Governance

<table>
<thead>
<tr>
<th>Modularity</th>
<th>Change Volume Trend</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integration</td>
<td>Unit Test</td>
</tr>
<tr>
<td>Operations</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Adaptability</th>
<th>Cost of Change Trend</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit Test</td>
<td>Operations</td>
</tr>
</tbody>
</table>

Bayesian Reasoning For Better Predictions in the Face of Uncertainty

- Mathematical foundations
  - Probability and statistics
  - Common distributions
  - Computer simulation of large number of possible outcomes

- Understanding sources of uncertainty in extremely complex systems
- Making better decisions

\[
P(B|A) = \frac{P(A|B)P(B)}{P(A)}
\]
Bayes Theorem

\[
P(A) = \int_A d(e)de
\]
Probability Density

Monte Carlo Simulations

- Priors (Well reasoned Predictions)
- Evidence (New facts)
- Posteriors (More Reliable Predictions)
Monte Carlo simulation of project scope items

<table>
<thead>
<tr>
<th>Plan Item(s)</th>
<th>User stories</th>
<th>Work Items</th>
<th>Code</th>
<th>Release</th>
</tr>
</thead>
</table>

Developer Duration Estimates

Actual distribution of work item completion times

Monte Carlo Simulation of remaining work

91% 9%
IBM Research Prototype

- Likelihood measure
- Risk Trend
- Burndown status and forecast
- History Slider
Accelerated delivery demands a quid pro quo

**Engineering Practitioners**

*Embrace Measurement*

- Design, create, test
- Reuse knowledge, best practices
- Address uncertain things first
- Be adaptive to change

**Governance Stakeholders**

*Enable Agility*

- Achieve predictable outcomes
- Manage risk
- Ensure compliance
- Improve software economics
- Visibility and transparency

---

**The Speed Of Trust**
The Moral of This Story

Better software economics is a result of:

1. Measured improvement for improved predictability
   • The foundation of economic governance
   • Measurement helps you manage uncertainty

2. Agility for improved operational efficiency
   • Best measured by cost of change trends
   • Best achieved by accelerating integration testing

3. Automated Bayesian analytics for improved predictability
   • Painless governance and less overhead perceived by practitioners
   • Quantum leaps in trust among stakeholders

If you play better defense you can play more offense!
References