Empirical Assessment of Cost Factors and Productivity during Software Evolution through the Analysis of Software Change Effort

Hans Christian Benestad
June 30, 2009

Presentation of thesis for the degree of Ph.D.

Supervisors
Bente Anda
Erik Arisholm
Changes to operational software are inevitable*

Corrections: Developers will commit errors

Adaptations: Technological environments will change

Perfective/Enhancive: Users will require more functionality

Dimensions of maintenance**

*Essence of Lehman’s “first law of software evolution” (1976)
** As proposed by Swanson (1976)

50 billion USD worth of evolution costs, annually
The overall aim was to better understand development costs involved in making changes to software.

Systematic analysis of change effort

Goal 1: Identify factors that affect evolution costs
Goal 2: Improve methods to assess trends in productivity during software evolution

Improved practices

Improved evaluation of practices

1. Identify problems
2. Evaluate effects
3. Propose changes
Costs of software evolution can be assessed by analyzing drivers of change effort

People
- Experience

Product
- Structural attributes

Performed changes
- Size, type

Practices
- Collaboration

Costs of software evolution can be assessed by analyzing drivers of change effort.
<table>
<thead>
<tr>
<th>Systematic analysis of change effort</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal 1:</strong> Identify factors that affect evolution costs</td>
</tr>
<tr>
<td><strong>Goal 2:</strong> Improve methods to assess trends in productivity during software evolution</td>
</tr>
</tbody>
</table>
A framework for change-based studies was established using systematic literature procedures.

Systematic literature review
“A rigorous methodology to ensure a fair evaluation and interpretation of all research relevant to a phenomenon”

<table>
<thead>
<tr>
<th>Change attribute</th>
<th>Data provided by</th>
<th>Question asked</th>
<th>Typical values</th>
<th>Goal 1 studies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>Delta</td>
<td>How many logical units will be changed, added or deleted by the change?</td>
<td>Count of changed, units, weighted by complexity</td>
<td>-</td>
</tr>
<tr>
<td>Points</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Location</td>
<td>Development</td>
<td>Where are human resources located physically?</td>
<td>Distributed</td>
<td>-</td>
</tr>
<tr>
<td>Development org.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maintenance</td>
<td>Human</td>
<td>For how long had the developers performed software maintenance work?</td>
<td>Number of years</td>
<td>-</td>
</tr>
<tr>
<td>Experience</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Resource</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maintenance level</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change type</td>
<td></td>
<td>What was the purpose of the change?</td>
<td>Fix/Enhance/Adapt</td>
<td>[21, 37, 41-47, 49-51]</td>
</tr>
</tbody>
</table>
A case study investigated costs factors in two commercial software organizations.

Research model and hypotheses based on the established framework.

Collected 31 measures for 336 changes.

Evidence-driven statistical analysis.

Data-driven statistical analysis.

Interview analysis to explain large residuals.
Developers’ effort to comprehend and change dispersed code was an important cost driver.

Strong correlation

Number of components changed $\rightarrow$ Change effort

Comprehension of dispersed "relevant" code

May explain the observed lack of effect of static, class-based metrics

Comprehension occurred along object interactions within user scenarios, rather than architectural units

An additional effect occurred when comprehension and change spanned several technologies

Design practices and tools should recognize developers’ need to comprehend functional crosscuts of the software, in particular when several technologies are involved.
Volatile change request was an important cost driver

*Number of updates to a change request* Correlates with *Change effort*

*Correcting errors by omission* mediates

*Unforeseen side effects of change request*

*Insufficient knowledge in the interface between software and the business*

Software organizations should cultivate knowledge in the interface between the software and the business domain.
The qualitative analyses proposed a number of cost drivers not captured by statistical models.

Understand a complex underlying state-model

Develop reusable mechanisms

Circumvent technology flaws
## Systematic analysis of change effort

<table>
<thead>
<tr>
<th>Goal 1: Identify factors that affect evolution costs</th>
<th>Goal 2: Improve methods to assess trends in productivity during software evolution</th>
</tr>
</thead>
</table>

---

*Image: The Journal of Systems and Software* 

*Image: ENASE 2009*
Practical and trustworthy measures of productivity are needed to support process improvement

\[
\text{productivity} = \frac{\text{output production}}{\text{input effort}}
\]

\[
\text{productivity} = \frac{\text{developed size}}{\text{developer effort}}
\]

Measures of developed size:
- **Lines of code** [1]
- **Developed components**
- **Function points** [2]
- **Specification weight metrics** [3]

Such measures can be adopted to software evolution [4, 5, 6]

1. Fenton&Pfleeger-1997
2. Albrecht-1983
3. DeMarco m-1984
4. Ramil&Lehman – 2000
5. Maya&Abran -1996
6. Abran&Maya - 1995

Claim: Current measures are either difficult to collect or have questionable validity
Compare the time needed to complete change tasks between two time periods: Four variants

Compare mean effort

Control for change characteristics

Compare actual with predicted effort

Estimate the same tasks in P0 and P1

Inspired by:

[Trad.]

[Eick et al., 2001]

[Kitchenham&Mendes, 2000]

[Arisholm&Sjøberg, 2001]
First application: Assessment showed opposite productivity trends, consistent with major project events.

- Performed a major restructuring effort of the software.
- Switched from fixed price to time and material contracts.

<table>
<thead>
<tr>
<th></th>
<th>ICPR₁</th>
<th>ICPR₂</th>
<th>ICPR₃</th>
<th>ICPR₄</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICPR₁</td>
<td>0.81</td>
<td>1.50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICPR₂</td>
<td>0.90</td>
<td>1.50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICPR₃</td>
<td>0.78***</td>
<td>1.18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICPR₄</td>
<td>1.00</td>
<td>1.33*</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

First cut evaluation: Statistical tests to determine statistical significance.

- Indicates higher productivity.
- Indicates lower productivity.
Second application: Assessing the effect of a new estimation practice

Surprisingly, the estimation method seemed to affect change effort, rather than estimation accuracy.

Indication of lower productivity using P1

No difference in productivity when controlling for differences in complexity of changes

Structured interviews: P1 helped in identifying subtasks/side effects

Indicators help in discovering, and understanding causes for productivity trends
In summary, the systematic change-based analysis proved effective to understand development costs during evolution.

A framework for measuring and analyzing changes that combines quantitative and qualitative methods.

A promising method for assessing productivity trends during software evolution.

Empirical evidence and understanding of important cost drivers in software evolution.

Thank you for listening.
The overall aim was to better understand development costs during software evolution.
Comparison of change task properties helps in validating the indicators’ assumptions

Table 3. Properties of change tasks in RCN

<table>
<thead>
<tr>
<th>Variable</th>
<th>$P_0$</th>
<th>$P_1$</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>chLoc (mean)</td>
<td>26</td>
<td>104</td>
<td>0.0004</td>
</tr>
<tr>
<td>crWords (mean)</td>
<td>107</td>
<td>88</td>
<td>0.89</td>
</tr>
<tr>
<td>filetypes (mean)</td>
<td>2.7</td>
<td>2.9</td>
<td>0.50</td>
</tr>
<tr>
<td>isCorrective (%)</td>
<td>38</td>
<td>39</td>
<td>0.90</td>
</tr>
</tbody>
</table>

Change tasks were indeed different between the periods

Table 4. Properties of change tasks in MT

<table>
<thead>
<tr>
<th>Variable</th>
<th>$P_0$</th>
<th>$P_1$</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>addCC (mean)</td>
<td>8.7</td>
<td>44</td>
<td>0.06</td>
</tr>
<tr>
<td>components (mean)</td>
<td>3.6</td>
<td>7</td>
<td>0.09</td>
</tr>
<tr>
<td>crTracks (mean)</td>
<td>4.8</td>
<td>2.5</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>systExp (mean)</td>
<td>1870</td>
<td>2140</td>
<td>0.43</td>
</tr>
</tbody>
</table>

Indicator controlling for the differences was justified
This presentation describes the motivation, research approach and key results.

- Change effort

- Software evolution cost

**Analysis of individual changes to understand software evolution costs**

**Systematic review and multiple case study as key research methods**

**Evidence on cost factors**

**Method for measuring productivity**
Frameworks for improving software processes and products presume that productivity can be measured.

Deming

Productivity measures are essential

GQM

QIP

21