Application of economic models on software maintenance

Anh Nguyen Duc
Trial lecture
Trondheim, Norway
April 10th 2015
Agenda

- Software maintenance
- Cost models of the maintenance phase
- Economic measures of maintenance activities
- Value driven prioritization of maintenance task
Significance of software maintenance

2 to 100 times:
maintenance cost vs. development cost

Figure from Floris and Harald, 2010
Software maintenance types

The totality of activities required to provide cost-effective support to software

Corrective
Adaptive
Perfective
Preventive

Fault repair (17%)
Software adaptation (18%)
Functionality addition or modification (65%)
Software maintenance process
Economic concerns in software maintenance

**Business level**
- Should we continue maintaining the system in the next X year or buying a new system?
- How much to invest on maintenance for the greatest earned value (saved cost, increased income)?

**Product level**
- How much people need to be assigned for maintaining the module Y?
- How much does maintenance cost in comparison to development cost?

**Project level**
- How long does it take to fix a post-release software defect?
- Which features require the least cost but provide the greatest customer value?
Agenda

- Software maintenance
- Cost models of maintenance phase
- Economic measures of maintenance activities
- Value driven prioritization of maintenance task
History of parametric cost models

- 65-85: the search for right parametric forms
- 85-95: advances in size and complexity metrics
- 95-05: proliferation of software development styles
- 05-15: advances in prediction and classification models
Belady and Lehman (1972)

\[
\text{Effort} = p + K^{c-d}
\]

- **Effort**: total maintenance effort
- \(p\): productive effort, including analysis, design, code testing
- \(d\): degree of maintenance team familiarity with the software
- \(c\): complexity caused by lack of structured design and document
- \(K\): empirical constant, depends on the environment
COCOMO (1981)

- Effort (person-month) as a function of size (LOC)

\[ \text{Effort} = b \times \text{Size}^c \]

- **Organic**: small teams develops software in known environment \((b=2.4, c=1.05)\)
- **Embedded**: inflexible and constrained environment \((b=3.6, c=1.20)\)
- **Semidetached**: varying levels of team experience working on larger projects \((b=3.0, c=1.12)\)

\[ \text{Effort} = b \times \text{Size}^c \times \text{EAF} \]

- Intermediate model
- \(b, c\): calibrated factors
- EAF: effort adjustment factor
COCOMO (1981) – Maintenance cost model

Maintenance effort = ACT x Development effort

Annual change traffic (ACT) = (NNL + NML)/ NOL

- NNL: number of new LOCs
- NML: number of modified LOCs
- NOL: number of original LOCs

Example:
- A ERP module with original size of 10,000 KLOC
- Development effort is 600 PMs
- Estimated 1,500 new and modified KLOC per year
- ACT = 1500/ 10000 = 15% per year

- Maintenance cost of a 10 year project:
  \[ = 0.15 \times 600 \times 10 = 900 \text{ PMs} \]
COCOMO (1997) – Post architecture level

- **Application composition**
- **Early design**
- **Post-architecture**

\[
\text{Effort} = A \times \left[ \text{size} \right]^{1.01} \times \prod_{i=1}^{17} \text{EM}_i \]

- **SF**: cost drivers, i.e. development flexibility, architecture/risk resolution, team cohesion and process maturity
- **EM**: 17 effort multipliers, i.e. the required software reliability, database size, product complexity, required reusability, documentation, etc.
Function Point based estimation

• Function point (FP) as a consistent and early size metric

EFP = (ADD + CHG + CFP) × VAF_a + (DEL × VAF_b)

• EFP: enhanced project function point
• ADD, CHG, DEL: added, changed and deleted FPs
• CFP: unadjusted function points added by conversion
• VAF_a and VAF_b: value adjustment factors of the application after and before the project
COCOMO suite of models (1981-2007)

**Advantages**
- Repeatable estimations
- Easy to modify input
- Easy to customize and refine formula

**Disadvantages**
- Subjective inputs
- Unable to deal with exceptional conditions
- Mainly designed for waterfall
- Needs historical data for calibration

Figure from Boehm et al. 2008
Model calibration

• A systematic adjustment of model parameters to get an expected output with known inputs

• Proper calibration is important

  ▪ COCOMO II 1997 calibration of 83 data points: accuracy within 20% of actual values in 46% of the times

  ▪ COCOMO II 1998 recalibration of 161 data points: accuracy within 30% of actual values in 75% of the times
Model evaluation

• Modest predicting results
  – “… (estimating in general varies) from as much as 85 - 610 % between predicated and actual values …” (Kemerer 1993)
  – “…best models had MMRE (Mean Magnitude of Relative Error) of 60%, ... too inaccurate to be of any use …” (Jørgensen 1997)

• Combination with other estimation approaches
  – Expert judgments, case based reasoning, regression models, Bayesian network ...

• Milestone budget and schedule for project stakeholder negotiation and expectations management

• Basis of software maintenance planning and control
Agenda

- Economic aspects of software maintenance
- Economic measures of maintenance activities
- Cost models of maintenance phase
- Value driven prioritization of maintenance task
Value-based software maintenance

• The decision to perform maintenance at any time is based on cost benefit analysis and economic measures

• Earned-value system
  • A set of maintenance tasks
  • Planned Value (PV) in money and time of the tasks
  • Quantification of Earn Value (EV)
Return on Investment (ROI)

ROI = (benefit – cost) / cost

• Benefit = Value of Changed System - Value of Original System
• Cost = Adjusted size / productivity x Cost driver factors

Example:
- The benefit of the current system is 250,000 USD
- The benefit of the adapted system is 300,000 USD
- The estimated cost of change: 14,000 USD

The benefit of the maintenance task is:
300,000 – 250,000 = 50,000 USD

ROI = (50,000 – 14,000) / 14,000 = 2.57
Net present value (NPV)

\[ PV_{O&M} = \sum_{i=n_1+1}^{n_1+n_2} \frac{c_2}{(1+r)^i} \]

- Development phase requires \( n_1 \) years with a cost of \( c_1 \) at the end of each year.
- Operation and maintenance (O&M) phase requires \( n_2 \) years with a cost of \( c_2 \) at the end of each year.
- Cost of capital is at a change rate of \( r \% \)
Management of technical debts

- Metaphor indicating possibly of significant economic consequences in software maintenance
- Technical debts occur when making the change worked as quickly as possible with as few resources as possible

Increased productivity, lowered development costs of the current release

Increased maintenance costs, debt out of controls
Management of technical debts

• Principal = cost of paying off the debt
  – Maintenance task effort estimation
• Interest benefit of paying off
  – Interest probability x interest amount
• Which debts to pay off?
• Applications of technical debts
  – Project budget allocation
  – Prioritizing maintenance tasks
Agenda

- Economic aspects of software maintenance
- Economic measures of maintenance activities
- Cost models of maintenance phase
- Value driven prioritization of maintenance task
Focus on low-cost, high-value changes

- Technical debt pay-off
- Test case triage
- Defect prioritization
- Feature reduction
- ...

Perceived value

Effort-to-change

High

Low

Low

High
Deciding customer’s perceived value

• Approaches
  – Multi criteria decision analysis (MCDA): Analytic hierarchy process (AHP), PROMETHEE, etc
  – Agile: MoSCoW, planning game, quality house, hundred-dollar test, etc

• Expert judgment, no model based approach
Triage and prediction models

- Modification request (MR)
- History of commits
- Communication data
- Project context
- SIZE & COMPLEXITY
- Regression + machine learning

- Post-release defect triage
- Issue resolution time estimation
- Change proneness
- Defect proneness
- Auto task assignment
Indicators of change effort

SIZE & COMPLEXITY

Modification request (MR)

History of commits

Communication data

Project context

Regression + machine learning

Code metrics
- Cyclomatic complexity
- Object oriented metrics
- Entropy of code changes
- Churn metrics
- Code delta
- Code smell

Project context

Regression + machine learning

Communication data

History of commits

Modification request (MR)
Indicators of change effort

- Communication data
  - No. of comments
  - No. of commenters
  - Communication frequency
- Project context
  - No. of locations
  - No. of developers
  - CMMI levels
  - Team experience levels
  - ...
Indicator of change effort

Figure from Zhang et al. 2013

Figure from Nguyen-Duc et al. 2011
Reflection on model building

• Local vs. global prediction
• Implicit assumptions of models
• Universal indicators of software changes
• Data missing and preprocessing
• Compounding impacts
Summary

- Cost-benefit concern of software maintenance for vendors and buyers
- Measuring economic value of software maintenance activities
- Limited applicability of traditional maintenance costs
- Integration of code, process, human based metrics in estimating and triaging maintenance tasks
Reference

BOOK

Reference

PAPER (1)

Reference

PAPER (2)


Q&A