Estimating Software Maintenance

Arun Mukhija

Contents

- What is Software Maintenance?
- Facts and Figures
- Maintenance Activities and Costs
- Maintenance Estimation Models
- Conclusion and Discussion

What is Software Maintenance?

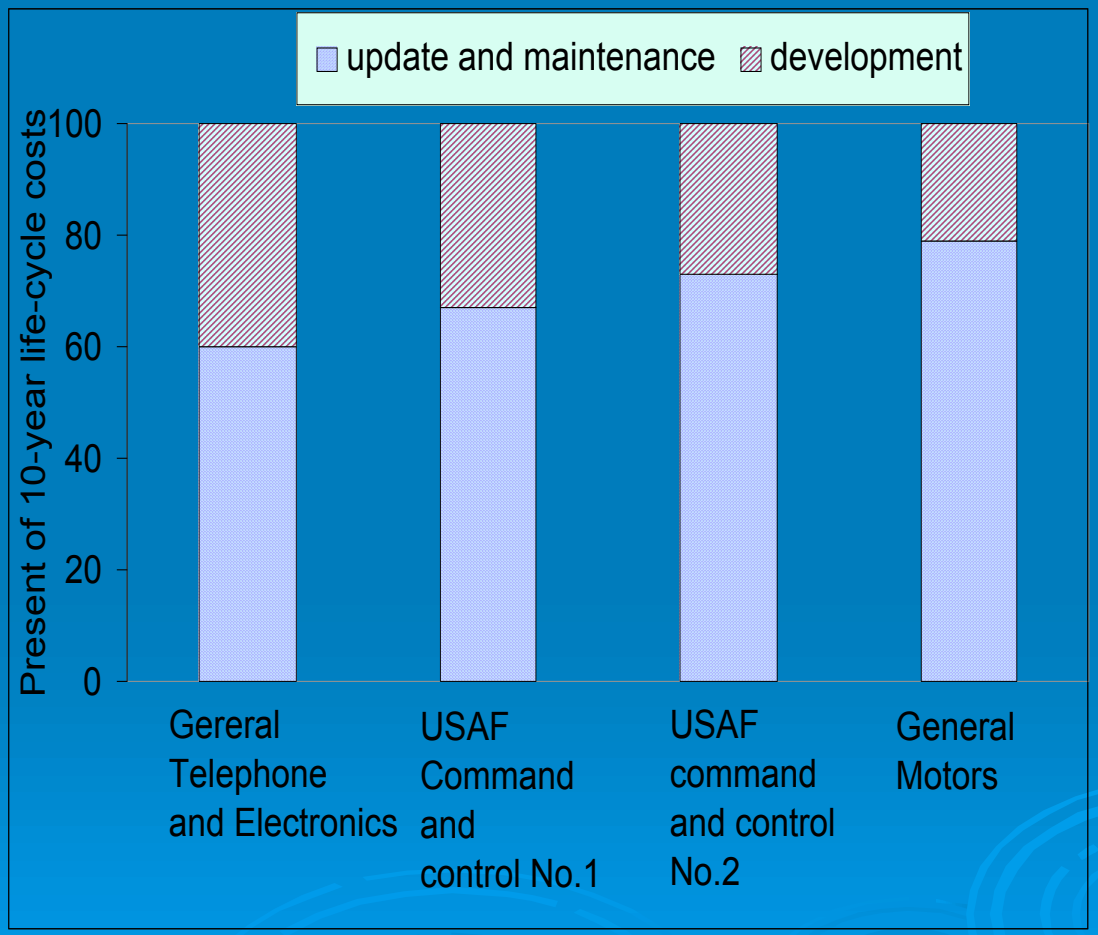
- "Changes that have to be made to computer programs after they have been delivered to the customer or user." *
- Software maintenance includes:
  - Corrective maintenance
  - Adaptive maintenance
  - Perfective maintenance
  - Enhancements (Although technically they are not a part of software maintenance but, being a post-release activity, are often considered a part of it)

* Martin J. and McClure G. "Software Maintenance: The Problem and its Solutions", Prentice Hall (1983).

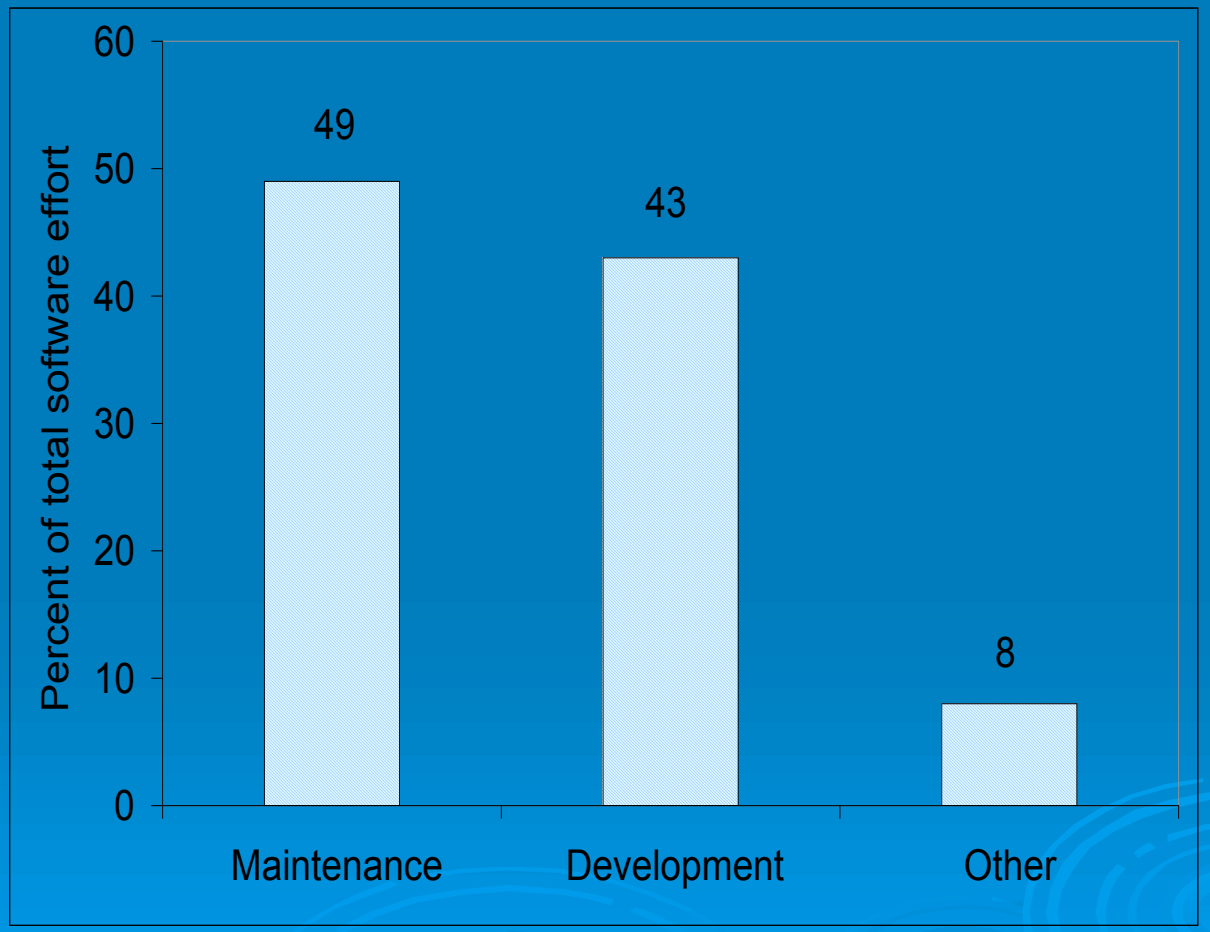
Facts and Figures

- Software maintenance costs around 50% of total software life-cycle cost.
- But relatively little is known about the software maintenance process and the factors that influence its cost.

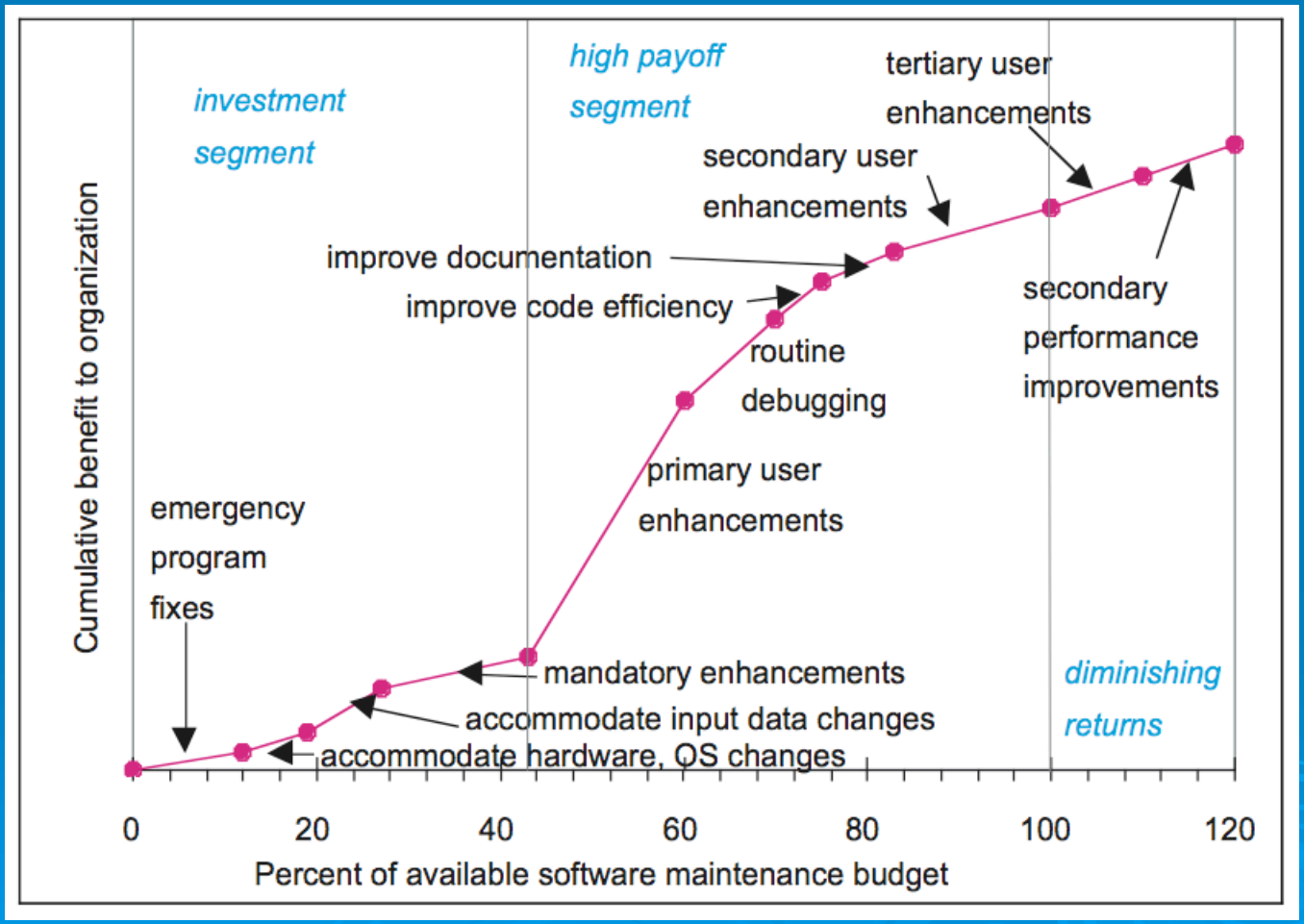
Software Development and Maintenance Costs in Large Organizations [Boehm81]



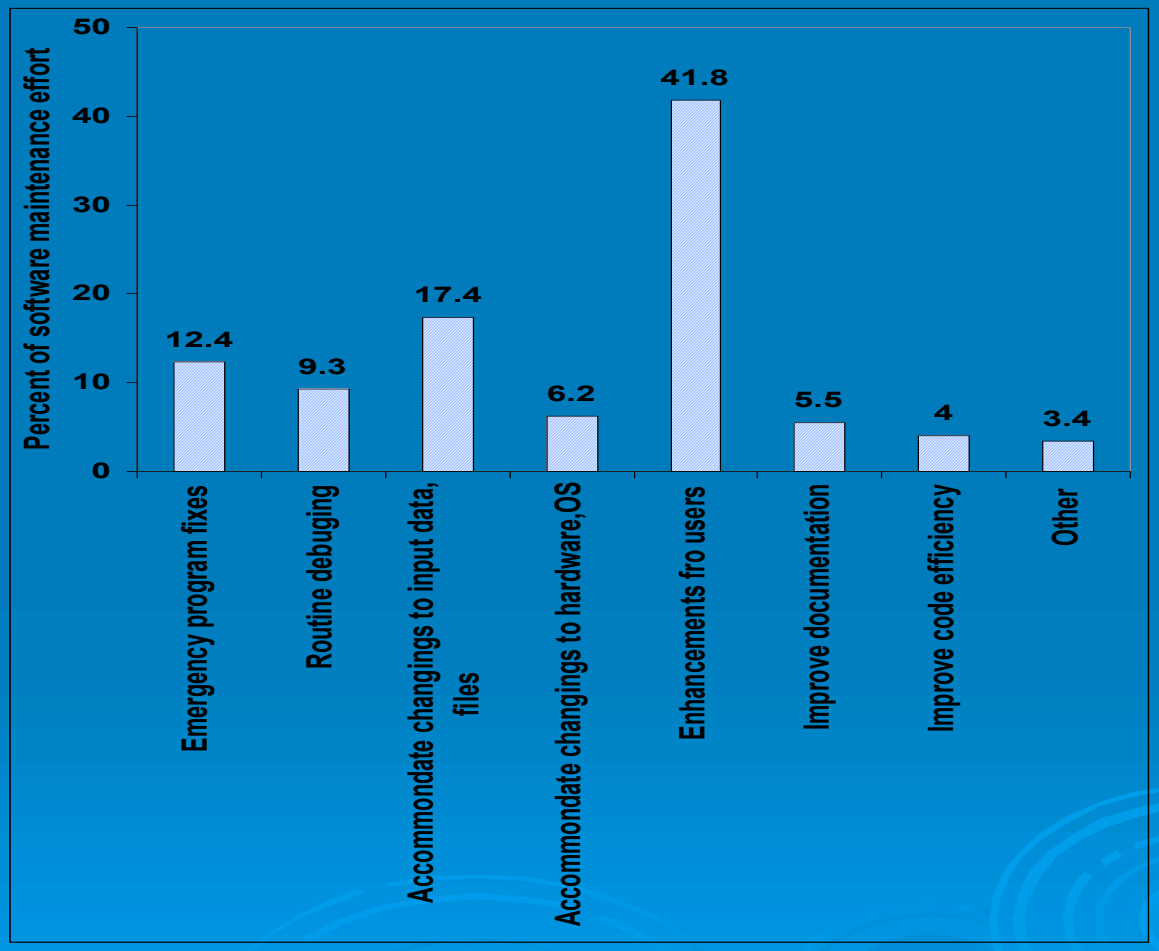
Software Development and Maintenance Costs in 487 Organizations [Boehm81]



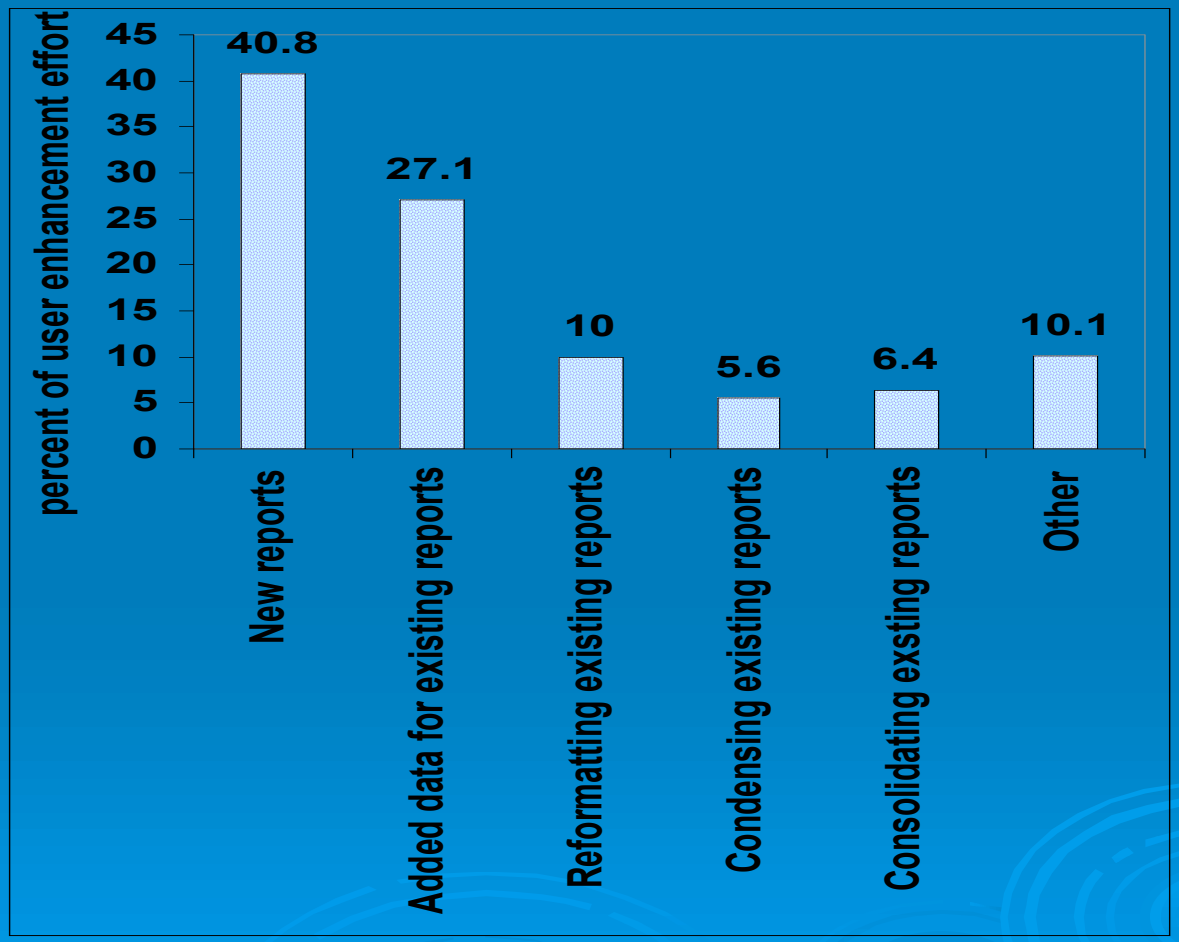
Software Maintenance Production Function [Boehm81]



Distribution of Software Maintenance Effort [Boehm81]



Distribution of User Enhancement Effort [Boehm81]



Maintenance Activities and Costs

Defect repairs

- keep software in operational condition
- costs absorbed by software supplier
- low pre-release defect removal efficiency (~85%)
- productivity = 8 defect repairs per month
  (can be higher with experienced personnel and defect-tracking tools etc.)

Factors influencing defect repairs:

- Abeyant defects (10%) - based on unique combination of events
- Invalid defects (15%) - misdiagnosed errors
- Bad fix injection (7%) - derivative errors
- Duplicate defects - multiple complaints about the same error

- Error-prone module removal
  - concentration of errors in particular modules
  - common among large poorly-structured systems
  - expensive to maintain, due to high bad fix injection rate
  - 500% more expensive than normal modules
- Customer support
  - interface between clients and defect repair teams
  - effort depends on number of users
    - with phone contact, 1 customer support person for 150 users
    - with electronic contact, 1 customer support person for 1000 users

Code restructuring

- done by automated tools to lower complexity levels
- lowering complexity eases maintenance
- precursor to other maintenance activities

Migration across platforms

- from one OS or hardware to another
- with well-documented specifications,
  migration speed = 50 FP per month
- with missing or obsolete specifications,
  migration speed = 5 FP per month

- Conversion to new architectures
  - changes to interface or file structure of apps.
  - quality of specifications affects productivity
  - reverse engineering may need to be performed to extract missing design info.
- Mandatory changes
  - in response to changes in law or policy
  - involve high costs and tight schedules
  - difficult to predict in advance

- Performance optimization
  - to minimize delays in transactions
  - improving performance at trouble spots
- Enhancements
  - adding new features as per user request
  - funded by user
  - annual rate = 7% increase in FP total of an app.
  - high integration and testing costs for poorly structured apps.

Maintenance Estimation Models

COCOMO Maintenance Model

for software maintenance effort estimation

$$(MM)_{AM} = (ACT)(MM)_{DEV}$$

$(MM)_{AM}$: annual maintenance effort in man-month

$(MM)_{DEV}$: development effort in man-month

ACT : annual change traffic (fraction of software that undergoes change during a year)

For intermediate and detailed COCOMO,

$$(MM)_{AM} = (EAF)_M(ACT)(MM)_{NOM}$$

$(EAF)_M$: maintenance effort adjustment factor

Maintenance/Development Cost Ratio

$$(MM)_M = (M/D)(MM)_{DEV}$$

$(MM)_M$: overall life-cycle maintenance effort in man-month

$(MM)_{DEV}$: development effort in man-month

M/D : maintenance/development cost ratio

Value of M/D ranges from 0.67 to 4.5, depending on application type.

Cards-per-person ratio

origin: number of cards each software person can maintain

$(KDSI/FSP)_M$: KDSI maintained per full-time software person

$$(FSP)_M = \frac{(KDSI)_{DEV}}{(KDSI/FSP)_M}$$

$(FSP)_M$: number of software maintenance personnel required

$(KDSI)_{DEV}$: size of software in KDSI

Value of $(KDSI/FSP)_M$ ranges from 3 to 132, depending on application type.

The annual maintenance effort $(MM)_{AM}$ is then simply

$$(MM)_{AM} = 12\ (FSP)_M$$

Maintenance Productivity Ratio

$$(DSI)_{MOD/YR} = (ACT)(DSI)_{DEV}$$

$$(MM)_{AM} = \frac{(DSI)_{MOD/YR}}{(DSI/MM)_{MOD}}$$

$(DSI_{)MOD/YR}$: number of source instructions modified per year

$(DSI)_{DEV}$: size of software in source instructions

$(MM)_{AM}$: annual maintenance effort in man-month

ACT : annual change traffic

$(DSI/MM)_{MOD}$: maintenance productivity ratio (number of source instructions modified per man-month of maintenance effort)

Average value of ACT is 0.092 and of $(DSI/MM)_{MOD}$ is 241, based on a survey.

Conclusion and Discussion

- “Software processes must produce software that can be gracefully evolved at reasonable costs. The choice of software architecture significantly influences modifiability and hence maintainability.” *

- Estimating maintenance is complex because of the relationship between base application and changes being made. Moreover predicting adaptive maintenance and enhancements in advance is very difficult.

* Richard D. Stutzke. “Software Estimating Technology: A Survey”, CrossTalk (1996).

References

- [Jones98] Jones T.C. “Estimating Software Costs”, McGraw Hill (1998).
- [Boehm81] Boehm B. “Software Engineering Economics”, Prentice Hall (1981).