Kingdom of Saudi Arabia
King Saud University
College of Computer and Information Sciences
Computer science Department

Software Cost Estimation:
A Tool for Object Oriented Console Applications

Presented by:
Amel Al hossan

supervisor
Prof. Dr. Ghazy Assassa
Software Cost Estimation Model

For

Object Oriented Software

Console Applications

Visual application = Visual part + Programming language part

The application does not use any internal visual components provided by any language

This part is our objective
Agenda

- Introduction
- Related work
- The proposed Cost Estimation Model
- The measurements in the estimation process
- Experimental data used to test our method.
- Experimental result
- Results notes
- Conclusion & future work

Demo for
The Cost Estimation Tool for the proposed model
Introduction

The cost estimation model need to be:

- Reliable and accurate
- The accuracy can be improved by the calibration
- Accurate estimation of size is vital
- The proper size measure effect the model accuracy.
Reliable & Accurate
Cost Estimation Model

Model Calibration

Proper Size Measure
The proposed cost model improve the accuracy through:

- Calibrate COCOMOII early design model for specific environment
- Use (OOFP) as a size measure with calibrated model.
- Apply Linear regression to relate OOFP with LOC
Related work

Several authors assert that a model's predictive accuracy can be improved by calibrating (adjusting) its default parameters to a specific environment.

Thibodeau [1] calibrated nine models using three databases.

the IITRI [2] study with original COCOMO

In 1997 COCOMO II Post-Architecture model was calibrated by Bernheisel [3]
Related work

These studies showed estimating accuracy improved with calibration.

In this study

COCOMO II Early Design Model calibrated.

Effort PM = A * (Size)^B * EM_1 * EM_2 * ............ * EM_7
Related work

A lot of researchers have proposed many methods for size measure such as:

- function point (FP)
- Object Point (OP)
- Predictive Object Point (POPs)
- Object Oriented Function Point (OOFPs)
- Object Oriented Design Function Points (OODFP)

COCOMOII models use only the traditional FPs & OP.
Related work

The addition in this study:

We combine the OODFP with calibrated COCOMOII early design model.

The Cost estimation tool will

- Calibrate COCOMOII early design model according to the historical data.
- Automate OODFP counting procedure [4].
- Estimate the software size in LOC.
- Estimate the software effort and cost.
The proposed Cost Estimation Model
The proposed Cost Estimation Model

1. Historical OO Projects information
   - OOFP count sub-system
     - Historical Projects size in OOFP
     - Historical Projects size in LOC
   - Linear regression sub-system
     - LOC = b₀ + b₁ OOFP

2. COCOMO II Early design Model Calibration sub-system
   - New value of A

Estimate New Project

1. New OO Project information
   - OOFP count sub-system
   - New Project size in OOFP
   - LOC = b₀ + b₁ OOFP
   - New Project size in LOC

2. New Project
   - Effort PM= A * (Size)B * EM₁* EM₂*...* EM₇
   - Cost = Estimated Effort * Labor Rate
the measurements in the estimation process
the measurements in the estimation process

1- COCOMOII Early design Model Calibration

Effort PM = A * (Size)^B * EM_1 * EM_2 * ... * EM_{17}

SW House information
Process Maturity level (Exp5)
Labor Rate

Historical Object Oriented Projects of Software House
Actual effort, size, Effort Multipliers (EMs1..17), Exponent factors (Exp1..4)

Calibration of COCOMOII Early Design Model Sub-system

Calibrated Cost Model

Algorithm
- Enter SW House information
- Enter Historical Object Oriented Project information
- B = 1.01 + 0.01 \* \sum Exp_{1..5}
- Q_i = (Size)^B \* \sum_{1..n} EMs/n

New Value
A = (\sum_{i=1..n} PM_i \ Q_i) / (\sum_{i=1..n} Q_i^2)
In object oriented software design:

Classes & Methods

- Data in a class can be mapped to a logical file.
- Each method mapped to a transactional function.
Class Complexity

The complexity of an **logical file** (class) depends on the DETs and RETs

- A **DET** is a simple data type such as int, char.
- A **RET** is an Object reference, a complex data type

\[
\text{DET}_{\text{s}} = \text{number of simple data type} + \text{number of inherited simple data type}
\]

\[
\text{RET}_{\text{s}} = 1 + \text{number of complex data type} + \text{number of inherited Complex data type}
\]
Method Complexity

The complexity of a transactional function (method) depends on the DETs and RETs.

- A single valued association is considered as a DET.
- Multi-valued association is considered as a RET.
- If a method does not have any arguments and return type, then its complexity is considered as one DET.

the measurements in the estimation process
2-OOPF measurement calculation

<table>
<thead>
<tr>
<th>Classes info.</th>
<th>Methods info.</th>
<th>OOFP count sub-system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compute</td>
<td>DETs &amp; RETs</td>
<td>DETs &amp; RETs</td>
</tr>
<tr>
<td>Determine</td>
<td>Classes</td>
<td>Method</td>
</tr>
<tr>
<td></td>
<td>Complexity</td>
<td>Complexity</td>
</tr>
<tr>
<td>Determine</td>
<td>UOOFP</td>
<td>UOOFP</td>
</tr>
<tr>
<td>Compute</td>
<td>Value Adjusted Factor (VAF) by Evaluate 14 General System Characteristics (GSCs)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>VAF = (TDI * 0.01) + 0.65.</td>
<td></td>
</tr>
</tbody>
</table>

\[ \text{OOFP} = \text{VAF} \times \text{UOOFP} \]
3- Build a Regression Model to relate OOFP with Project’s LOC

**Algorithm**

- Calculate OOFP for \((n)\) old projects
- Read Actual LOC for \((n)\) old projects
- \(b_1 = \frac{\sum (\text{LOC}_i \times \text{OOFP}_i) - [(\sum \text{LOC}_i)(\sum \text{OOFP}_i)/n]}{(\sum \text{OOFP}_i^2)-[\sum (\text{OOFP}_i)^2/n]}\)
- Compute the average of OOFP
- Compute the average of LOC
- \(b_0 = (\text{AVG of LOC}) - b_1 (\text{AVG of OOFP})\)

Regression Model

\[
\text{LOC} = b_0 + b_1 \times \text{OOFP}
\]
the measurements in the estimation process

4- Estimate New Project

Enter New OO Projects information

Estimate project size in OOFP

Estimate project size in LOC

LOC = $b_0 + b_1$ OOFP

Estimate project effort & Cost by calibrated model

Effort PM = $A \times (Size)^B \times EM_1 \times \ldots \times EM_7$

Project Effort

Cost = Estimated Effort $\times$ Labor Rate

Project Cost
Experimental data used to test our method
Experimental data used to test our method

In this project

The measurements (Cost drivers) & 14 General System Characteristics (GSCs) were collected for ten completed OO projects form a specific software house, for which both an OO design model and the final code (LOC) were available.

All were developed in the same environment, using the same language (C++ - console application)
Experimental Result

Show the results
Results Notes

The factors affect the model accuracy:

- The number of data points collected.
- The accuracy of data collected.
- The re-estimation process could be done as the OO design become more stable.
Results Notes

- the estimated values of the 7 effort multipliers.

- The regression model needs more than 10 data points to estimate the size of the project in LOC correctly.

- Some collected data are abnormal, so it will affect the calibration values and regression model parameters.
Conclusion

In this study we have propose a new cost estimation model for object-oriented software – written with C++ console application.

It based calibration of COCOMOII – early design Model that is a well-known and validated cost model.

The calibrated model use OOFP as size measure.

The size of the project in line of code (LOC) estimated by using linear regression model built from historical project of the organization.
Conclusion
A cost estimation tool is constructed to automate the following:

- Calibrate COCOMOII early design model for specific environment.
- Calculate OOFP for a new project and for all historical projects.
- Construct a linear regression model to relate OOFP of the project to LOC.
- Estimate the effort and cost for a new project
future work

I think it is better to improve the proposed model to estimate the cost for all OO C++ projects:

1- Console Application (No Generated code in the model)

\[ \text{Effort PM} = A \times (\text{Size})^B \times EM_1 \times EM_2 \times \ldots \times EM_7 \]

2- Windows Application (Visual component and Automatic generated code included in the model)

\[ \text{Effort PM} = A \times (\text{Size})^B \times EM_1 \times EM_2 \times \ldots \times EM_7 \]

\[ + \frac{((\text{ALOC}) \times (\text{AT}/100))}{\text{ATPROD}} \]

- **ALOC** is the number of automatically generated LOC
- **AT** the percentage of total system code which is automatically generated
- **ATPROD** is the productivity level for this type of code production.
Thank you
References


