Empirical Estimation of COCOMO I and COCOMO II Using a Case Study

Muhammad M. Albakri and M. Rizwan Jameel Qureshi

Department of Information Technology, Faculty of Computing and Information Technology, King Abdul-Aziz University,

P.O. BOX 80221 Jeddah 21589, Saudi Arabia

+966 556 338283, +966 536 474921

Email: hiaimer@hotmail.com, anriz@hotmail.com

***Abstract*- There are several software estimation models such as Line of Code, Function Point and COnstructive COst MOdel (COCOMO). The original COCOMO model is one of the most widely practiced and popular among the software development community because of its flexible usage. It is a suite of models i.e., COnstructive COst MOdel I and COnstructive COst MOdel II. In this paper, we are evaluating the both models, to find out the level of efficiency they present and how they can be tailored to the needs of modern software development projects. We are applying COCOMO models on a case study of an E-Commerce application, that is built using HTML and JavaScript. We will also shed light on the different components of each model, and how their Cost Drivers effect on the accuracy of cost estimations for software development projects.**

***Keywords*- COCOMO I; COCOMO II; Software Cost Estimation; Software Cost Drivers' Assessment; Trade-off Analysis; Component Composition.**

I. Introduction

The main stimulus for the COCOMO I model is to help people understand the cost consequences of the decisions they will make in developing and supporting a software product. COCOMO II not only offers a cost estimation tool, but also provides a great amount of parameters which explain what the model is estimating, and why it produces the estimates it does. COCOMO I is actually a hierarchy of three sub-models and each sub-model is progressively more detailed than the other. This paper will present our results and findings after applying two of COCOMO's sub-models. The First sub-model is 'Basic COCOMO'. It is a single-valued model and calculates the software development cost and effort of a program by measuring lines of code(LOC). Basic COCOMO itself is divided into three modes based on the nature of the software project. First is 'Organic Basic COCOMO', it is used in small-sized simple software projects developed by small teams with good application experience. Second is 'Semidetached Basic COCOMO', it is used in medium-size software projects developed by teams with diversified levels of experience. Third is 'Embedded Basic COCOMO', that is used in massive software projects with strict resource constraints developed by multiple teams acquiring immense levels of experience, and sophistication. The second sub-model is 'Intermediate COCOMO', it is simply 'Basic COCOMO' plus a set of subjective 'Cost Drivers'. Those drivers are used to assess product, computer, personnel, and project attributes of a software project. The evaluator uses a six-level scale to decide where each attribute fall. When an attribute is assessed, it produces what is called an Adjustment Factor. After all adjustment factors are multiplied together, they give an Effort Adjustment Factor(EAF), that is usually equal to a value between 0.9 and 1.4. The EAF is then mathematically applied on all Basic COCOMO's formulas. Third sub-model is Detailed COCOMO, as the name indicates, it produces the most accurate estimation of all three sub-models of COCOMO I. It combines Basic and Intermediate COCOMO together, boosted by an assessment of every Cost Driver's impact on each stage of 'Barry Boehm's software engineering process'. COCOMO II model on the other hand, is divided into four sub-models. Each sub-model is based on different inputs and estimates the effort of different activities of a software project. 'Application Composition' is the first sub-model. It estimates the effort of prototype systems developed using scripts, database programming, etc. And it uses application points as an input. Second sub-model is 'Early Design', it calculates initial effort based on system requirements and design options, and uses function points as an input. Third sub-model is 'Reuse', it estimates the effort of integrating reusable automatically generated components and uses generated line of code as an input. Fourth sub-model is 'Post Architectural', it estimates the development effort of system design specifications and uses lines of source code as an input.

The paper is further organized as: section 2 covers related work. Section 3 defines the research problem. Section 4 describes the brief case study design. Section 5 illustrates the evaluation. Section 6 covers the discussion.

2. Related Work

Boehm et al. [1] proposed evaluation criteria for the validity of the process models and they provided effective results. This article also explained the strengths and weaknesses of various cost estimation techniques for the period of 1965 to 2005 (40 years). COCOMO-II [2] was an excellent model up to 2005 but it did not enfold the new requirement and development styles for the reuseness or estimation of cost. COCOMO-II directed the software experts to create and designed new models such as the Chinese government version of COCOMO (COGOMO) and the Constructive Commercial-off-the-Shelf Cost Model (COCOTS) etc. Different future challenges were discussed for the invention of new model/methods and tools.