This paper describes a case study in the use of the COCOMO\textsuperscript{1} cost estimation model as a tool to provide an independent prognosis and validation of the schedule of a software project at IBM UK Laboratories Ltd, Hursley. Clearly case studies have the danger of being anecdotal however software engineers often work in situations where sufficient historical data is not available to calibrate models to the local environment. It is often necessary for the software engineer to attempt to use such tools on individual projects to justify their further use. This case study describes how we began to use COCOMO and concentrates on some of the problems and benefits which were encountered when trying to use COCOMO in a 'live' development environment.

The paper begins by discussing some problems in mapping the COCOMO phases on to the IBM development process. The practical aspects of gathering the development parameters of the model are described and the results of the work are presented in comparison to a schedule assessment using other prognosis techniques and the planned schedule at other milestones in the project's history. Some difficulties experienced in interpreting the data output from the model are discussed. This is followed by a brief comparison with other schedule analysis techniques used in quality assurance. We hope this case study shows that despite the problems in trying to use models such as COCOMO there are significant benefits in helping the user understand what is required to use such tools more effectively to improve software development cost estimates in the future.

**Keywords:** cost estimation, cost models, software development, metrics, risk assessment, project management

1. Introduction

The mission of product assurance was 'to provide general management with an independent technical assessment of the progress, schedules, quality, function versus customer requirements and risks associated with a product under development'. Assessment of product schedules was attempted by trying to model parts of the software development process using commonly accepted data. Our effectiveness was measured by our ability to provide early and accurate identification of potential risk and so help management make well-informed, reasoned decisions. Cost estimation models could be a useful tool to provide regular schedule assessments. We chose to do a case study on one project with COCOMO because

(1) COCOMO had a high visibility in the industry and had a local users' group;  
(2) COCOMO was in the public domain and was the best documented model (Boehm, 1981);

\textsuperscript{1}COCOMO is the 'CO\textit{n}structive CO\textit{S}t MO\textit{d}el' developed by Barry Boehm at TRW, Inc (Boehm 1981).
(3) There were several computer programs based on COCOMO which made it easy to use;
(4) COCOMO had been tried previously with some promise.

COCOMO was intended for use in validating the cost (in terms of manpower, allocated time and necessary financial commitments) of a project before the project had started. In this way the results could be used to support management decisions in any trade-off analysis at the proposal stage. It was clear, however, that this sort of modelling could equally be used during the project implementation phases at major project milestones to provide a schedule prognosis of the end of each project stage. In fact, Boehm recommends an iteration and comparison step as one of the seven basic steps in software cost estimation. If schedule assessments could be based on some standard parameters accepted by project managers as good measures of a product's progress then this would enable us to provide more equitable risk assessments on whether products are able to meet their committed completion dates. Using COCOMO in this way could also provide a 'sanity check' of the results from other methods of schedule assessment.

Clearly, case studies have the danger of being anecdotal, however software engineers often have to work in an environment where complete sets of historical data are unavailable and so development of new models or calibration of existing models is impossible. Unlike experimental science, which often takes place in a controlled research environment with no real delivery constraints, the software engineering laboratory is the live project with defined customer commitments. The software engineer therefore often has to pick up existing tools and use them 'as is' to determine whether further work to collect and analyse the data required to develop better models would bring a reasonable return on the investment. Further, much of the literature, including Boehm's book itself, requires a high degree of understanding of terminology and does not address some of the practical problems of trying to use tools such as COCOMO. This paper tries to describe some of these problems to help others anticipate how much effect is involved in trying to use COCOMO for schedule assessment.

*Software Engineering Economics* by Boehm (1981) was used extensively to plan and do this study and any further references to Boehm can be taken to imply the use of this book.

---

2. Mapping COCOMO phases to a 'real world' development process

To make it possible to compare the COCOMO results with the project plan it was necessary to define the IBM development process in terms of the COCOMO phases. The correlation between the IBM process and the COCOMO phase definitions proved to be harder than we imagined for the following reasons.

(1) With increased business competition many companies are moving towards using testable function units with overlapped phases. This makes it more difficult to determine the end of a phase since many phases could be in progress at any one time on more than one functional unit.

(2) The development and testing part of the waterfall model used in COCOMO did not include

---

2There is an argument that this does not reduce cycle time but increase it because any change in dependent modules after testing can lead to the need to do more regression testing.