COCOMO-SCORM

Interactive Courseware Project Cost Modeling

Roger Smith & Lacey Edwards
SPARTA Inc.
13501 Ingenuity Drive, Suite 132
Orlando, FL 32826
Roger.Smith, Lacey.Edwards @Sparta.com

Abstract. The U.S. Department of Defense has made significant investments in interactive courseware designed to train personnel distributed around the world. As these projects have multiplied, so too have the different approaches to estimating the schedule and staffing to create them. This paper presents the current status of a project that is creating a cost estimation algorithm for interactive courseware based on the COCOMO II family of software cost models. Our project focuses on specializing the COCOMO algorithm to account for the important variables to courseware development projects. We also will calibrate the algorithm to match historical data available from completed SCORM courseware projects.

Introduction

Today's learning society has developed an insatiable need for learning materials that are not limited to the traditional instructor-student-classroom design. This has led to the creation of new forms of knowledge transfer that are accessible to unlimited numbers of students at times and locations that are convenient to the student. Within the U.S. government, this has generated laboratories that focus on advanced methods for distributing learning content around the globe. It has also led to the creation of a standard to facilitate reuse and delivery of instructional content and functionality – the Shareable Content Object Reference Model (SCORM).

U.S. government organizations are creating courseware products that are a complex combination of custom software, commercial tools, and digital media content. The creation of such packages has led to a significant challenge for project managers and sponsors in calculating the expected level of effort, duration, and cost of such projects. In most cases, a specific company or government office will base these estimates on their internal experience with previous similar projects. However, as the need for courseware has grown, the number and diversity of estimation methods has grown with it. As a result, different organizations' estimates on similar projects can vary widely.

Our task has been to develop a reliable cost estimation algorithm for interactive courseware development. We selected the Constructive Cost Model II (COCOMO II) as the basis for the new algorithm. The algorithm we are developing focuses on the processes used and products created when developing courseware in compliance with the government Sharable Content Object Reference Model (SCORM) specification. Therefore, the algorithm has been labeled COSCOMO in keeping with a standing tradition of naming COCOMO-inspired cost models. While each agency, company, or department in the courseware community has evolved its own method for estimating the schedule and staffing for courseware development projects, the COSCOMO tool will be a shared asset in military offices to enable them to create consistent web-based SCORM project estimates based on industry practices and historical analysis of previous projects.

Cost Estimation Foundation

The Constructive Cost Model (COCOMO I) was first published in 1981 by Barry Boehm. Its reliability in estimating software projects led to its wide adoption in the software development community. Continued research to improve the algorithm led to the creation of COCOMO II in 1998 (Boehm 2000).

For this project, we will use the COCOMO II algorithm as a template from which to create a new model called COSCOMO for estimating courseware development costs. This model will build upon the COCOMO II approach by defining two types of cost drivers: Scale Factors and Effort Multipliers, described below.

Effort Multipliers. These factors account for required characteristics of the proposed product; characteristics of the delivery platform; capability and experience of the development team; and details about the content development environment and schedule. These variables, 17 in the COCOMO II model, capture the characteristics of a project that affect its level of effort.

Scale Factors. These variables focus on determining whether a project exhibits economies or diseconomies of scale. They measure the uniqueness of the project, the level of flexibility that a developer has, architectural complexity and risk management, the cohesion of the team, and the maturity of the process the developers will use.

Each of these cost drivers is part of the Effort Equation, which generates person-months of labor, and the Schedule Equation, which generates project duration. Together these two equations determine the staffing levels for a project. Models in the COCOMO family are independent of labor rates. They estimate only staffing levels and durations for projects. This allows organizations the flexibility to apply their own labor rates outside of the model, rather than attempting to integrate those rates inside the model for each unique customer. Other communities have adopted the COCOMO II method as a basis for their own tools. One COCOMO family tree is shown in Figure 1, adapted from (Boehm, et. al. 2005).

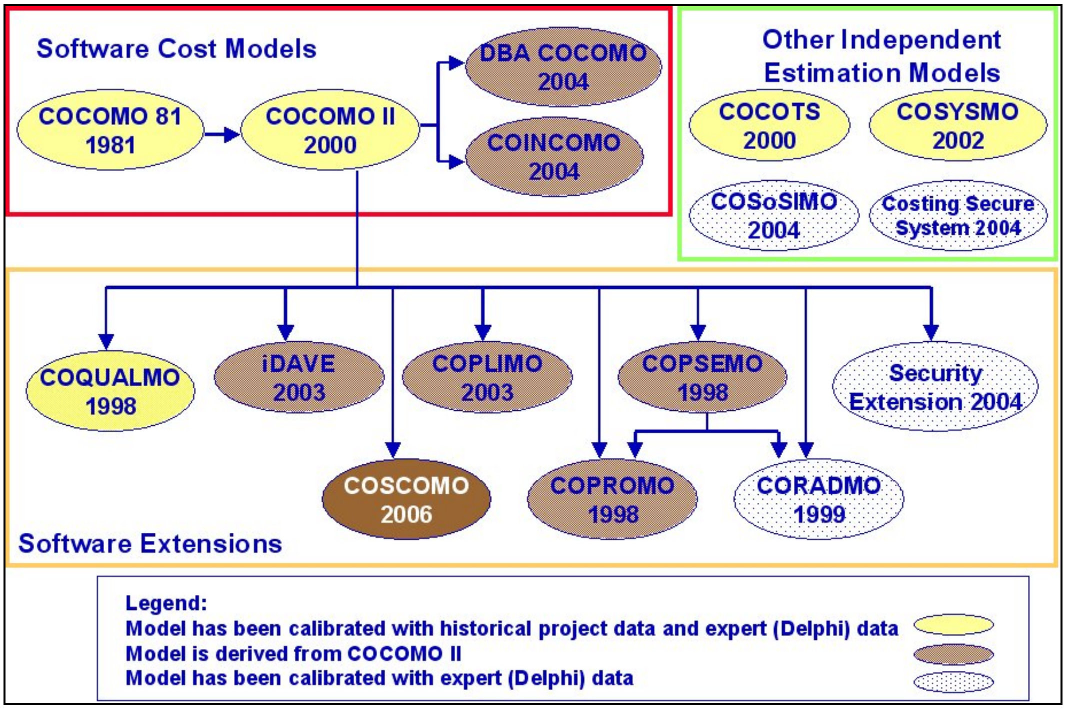


Figure 1. COCOMO Derivative Models.

Each community has taken a unique approach to modifying the COCOMO algorithm or to using the concepts to create an entirely new model. (Boehm 2005) classifies these models as (1) software cost models, (2) software extensions, and (3) other independent models. New models can be created by:

- Modifying existing variables,
- Adding or removing variables,
- Post-processing the results of a COCOMO model, or
- Decomposing a problem and applying multiple models.

We have added the new COSCOMO model to this diagram in the Software Extension category since it is a modification of the COCOMO II algorithm.

Characterizing Interactive Courseware Development

Interactive courseware products combine multimedia content, testing exercises, custom software, and commercial software tools to deliver instructional materials to students from any location on the globe. Figure 2 shows the three major categories of components.

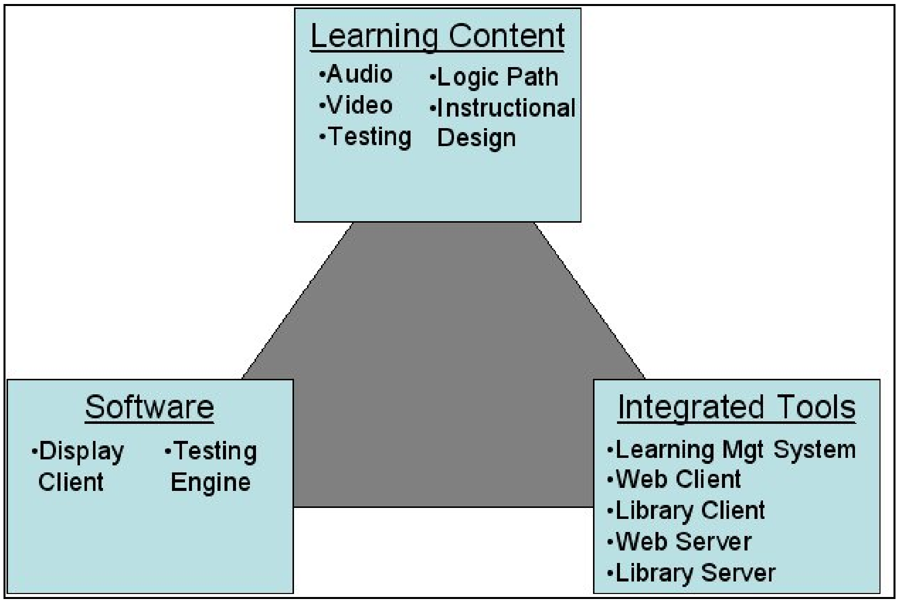


Figure 2. Major Components of Interactive Courseware.

The COCOMO II algorithm estimates the cost for development of software. Web-based courses often contain unique software for displaying data on the screen, passing data over a network, or calculating the results of a test or exercise, but software is only one part of courseware development.

The second part of providing the learning environment is integrating an existing system of software tools and computer hardware to deliver the information. One of the primary pieces of the system is a Learning Management System (LMS). This system can maintain a database on each student: past courses taken, courses required based on job function, performance on previous courseware, and suggestions for improvement. The LMS may also be able to operate as a content server, delivering the course to a student who selects it from a menu. For SCORM courseware, this delivery is often via the web as training becomes globally accessible. In this situation, the student client application is often a web browser configured with specific permissions (e.g. running Active X Components) or plug-in applications (e.g. Flash, Shockwave).

The third component is learning content. Unlike many pure software development or integration projects, interactive courseware requires a large volume of course content in the form of text, digital artwork, audio, video, and simulated worlds. This represents a significant investment of time and labor on the project. The projects also require the application of instructional design expertise that is used to determine how best to present information and to evaluate whether the student is learning as intended.

Courseware Development Process

The five-phase ADDIE Model – Analysis, Design, Development, Implement, and Evaluate – was developed in 1975 at Florida State University. The U.S. Armed Services adopted ADDIE, illustrated in Figure 3, as a valid Instructional Systems Design approach for training development. ADDIE has also become the framework for the Department of Defense Instructional Systems Design and Standards Approach to Training (DoD 2002).

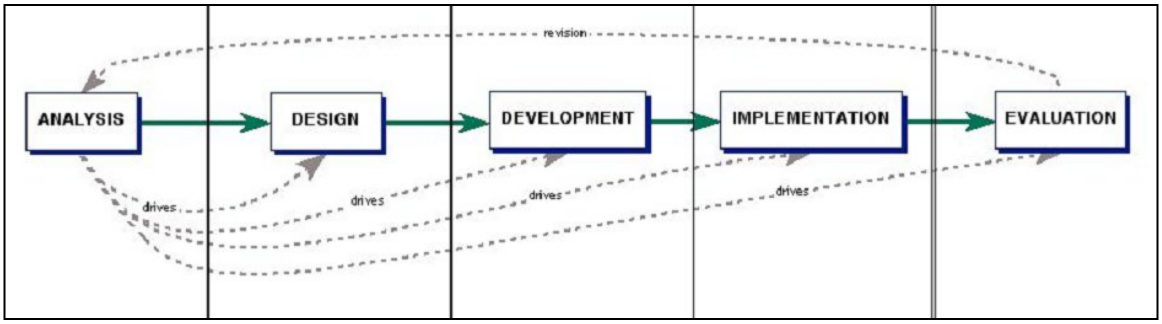


Figure 3. ADDIE Model for Instructional System Design.

As a discipline, instructional system design is based upon a set of values, specialized knowledge, intellectual skills, and methodologies that pinpoint performance problems and provide a means of solving them through training and other human performance solutions. The ADDIE model has become the most widely used model for describing and directing courseware development. Most other models contain phases that correspond closely to the five ADDIE phases (see Figure 3).

Government organizations and contractors often share the responsibility for completing the tasks and deliverables included in each phase of the ADDIE Model. For example, government personnel or independent specialists developing a Statement of Objectives (SOO) or Statement of Work (SOW) may conduct a preliminary training needs analysis, part of the ADDIE analysis phase. When this occurs, the contractor's entry point into the ADDIE Model and decision points within each phase change to reflect this new scope and schedule.

Model Adaptation

The COCOMO II model estimates the cost incurred for the development of a software product. The COSCOMO algorithm will do the same for courseware development. COCOMO II is driven by estimates of the size of a software product measured in source lines of code (SLOC) or software function points (FP), five scale factors, 17 effort multipliers, and two calibration constants. Our analysis of the SCORM courseware domain has led to modifications of this algorithm. In this section, we will explain the variables that drive the new COSCOMO algorithm, but will not provide an exhaustive explanation of the COCOMO II model. For a complete description of COCOMO II variables and their derivations please reference (Boehm et. al. 2000c) or the brief introduction in (Boehm 2000a). In calibrating the parameter rating scales and values, we will focus our efforts on those parameters that have the most significant effect on the model. Errors in some parameters will have a significantly greater effect on the accuracy of the estimate than errors in other parameters. For example, the COCOMO II model can tolerate much higher errors in the Scaling Factors than in the Size and Effort Multiplier values (Musilek 2002). In some cases, it may be appropriate to replace less significant parameters with constant values.

Modeling Process. To develop the new algorithm, we followed the COCOMO II Modeling Methodology (Figure 4) as closely as possible.

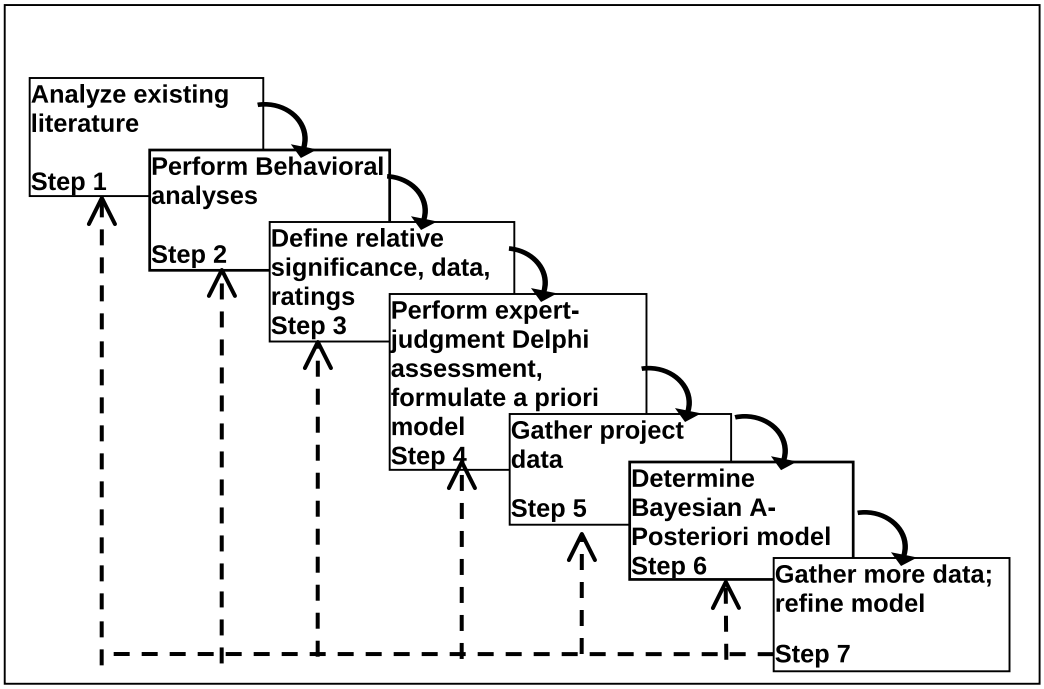

Analyze existing literature
Step 1
Perform Behavioral analyses
Step 2
Define relative significance, data, ratings
Step 3
Perform expert-judgment Delphi assessment, formulate a priori model
Step 4
Gather project data
Step 5
Determine Bayesian A-Posteriori model
Step 6
Gather more data; refine model
Step 7

Figure 4. COCOMO II Modeling Methodology

Many of the COCOMO II parameters carried over into COSCOMO with little or no modification. Factors such as the precedent for the products to be developed; the cohesion, continuity, and collocation of the team; the extent to which products must be reusable in future projects; and the capability and experience of personnel all influence the cost of developing learning content in much the same way they influence software development costs.

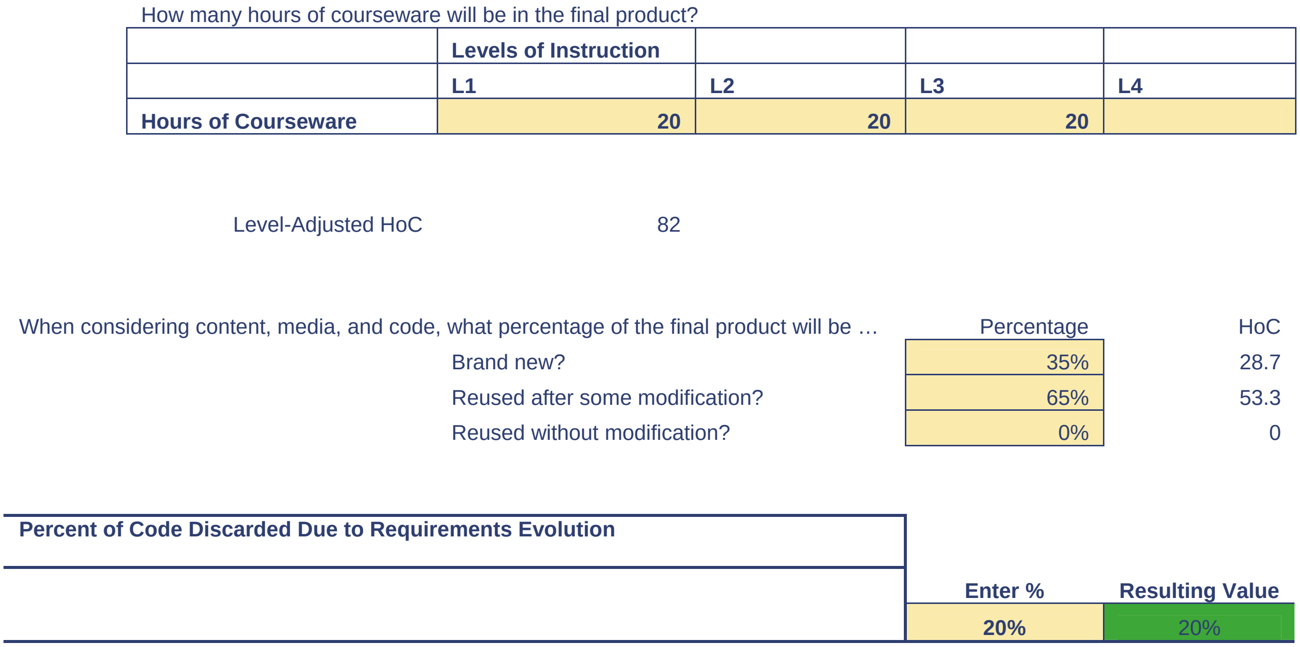
Size. In COCOMO II the size of a project is driven by the SLOC or FP's of a software product. FOR SCORM courses, most of the content is generated by tools or takes advantage of the capabilities of commercial products like the LMS or the web browser. The courseware community generally measures the "size" of a project by the number of contact hours with the student, known as "hours of courseware." Therefore, we have adopted this as the base measurement of the size of the project for COSCOMO.

As with software projects, in which code reuse is common, there are different sources for the courseware products. Developers may reuse some portion of prior projects, they may develop new courseware based upon the design of an existing course, or they may build a new course from scratch. Therefore, the size variable, hours of courseware, is modified based on the percentage of the contact time that is driven by the portion of that content that was built from a reused design.

Many SCORM courses are released as contracts or projects after the government has performed an initial analysis of the problem. Therefore, the "A" phase of the ADDIE model described above may or may not be part of the project estimate. COSCOMO will estimate the cost of executing a project, and will rely on a breakdown of ISD products and effort into the five ADDIE phases to extrapolate the total cost of an effort for all phases involved.

Given these categories of courseware size, the worksheet for entering the data into the model takes the form shown in Table 1.

Table 1: COSCOMO Size Drivers.



How many hours of courseware will be in the final product?

	Levels of Instruction			
	L1	L2	L3	L4
Hours of Courseware	20	20	20	

Level-Adjusted HoC 82

When considering content, media, and code, what percentage of the final product will be …	Percentage	HoC
Brand new?	35%	28.7
Reused after some modification?	65%	53.3
Reused without modification?	0%	0

Percent of Code Discarded Due to Requirements Evolution	Enter %	Resulting Value
	20%	20%

Effort Multipliers. The effort multipliers have the same function in COSCOMO that they do in the original COCOMO II model. They are qualitative measures that estimate variations in level of effort that are associated with the product, platform, personnel, and project. COCOMO II contains 17 of these multipliers. COSCOMO contains only 15. Three of the COCOMO II effort multipliers have been removed, one has been added, and the definitions of others have been changed. The resulting 15 COSCOMO effort multipliers are described in Table 2, organized by category.

Table 2: COSCOMO Effort Multipliers.

Product Multipliers

Required Reliability (RELY) *Modified from COCOMO II*	How reliable the product needs to be, and how much extra effort goes into the development process to ensure that level of reliability
Product Complexity (CPLX) *Modified from COCOMO II*	How complex the product is
Development for Reusability (RUSE) *Modified from COCOMO II*	How widely the product and its components will be shared in this or other projects, possibly throughout several services
Required Documentation (DOCU) *Modified from COCOMO II*	How much documentation is required compared to the amount of documentation that is necessary to support the product during its lifetime

Platform Multiplers

Platform Volatility (PVOL) *Modified from COCOMO II*	How often the deployment platform changes

Bandwidth Restrictions (BAND)
New in COSCOMO, replaces TIME and STOR from COCOMO II
The speed of the network across which the web-based training will be deployed

Personnel Multipliers

Senior ISD, Human Performance Team Capability (SCAP)
Modified from COCOMO II
The relative capability of the senior analysts, senior instructional system designers, and human performance factors team as compared to others in the industry

ISD, Development Team Capability (DCAP)
Modified from COCOMO II
The relative capability of the instructional system designers, programmers, and other development team members as compared to others in the industry

Personnel Continuity (PCON)
Modified from COCOMO II
How often the composition of the project team changes, measured in annual turnover

Courseware Applications Experience (APEX)
Reused from COCOMO II
Weighted average of the experience of the team in developing courseware applications

Platform Experience (PLEX)
Reused from COCOMO II
Weighted average of the experience of the team in developing applications for the deployment platform

Development Tools Experience (DTEX)
Reused from COCOMO II
Weighted average of the experience of the team in using the development tools chosen for the project

Project Multipliers

Availability of Lifecycle Tools (LIFE)
Modified from TOOL in COCOMO II
Level of development and lifecycle support tools available to the project team

Multi-site Development (SITE)
Reused from COCOMO II
How distributed the project team is, and how they communicate

Schedule Expansion (SCED)
Reused from COCOMO II
The relative length of the schedule for the project as compared to the schedule for a typical projects of the same effort

Figure 5 illustrates the effects of these effort multipliers. Each multiplier adjusts the total cost of the project up or down based on the numerical value associated with the selected rating, which varies from Extremely High to Very Low. In COCOMO II these multipliers have the ability to adjust the project costs down to 5.7% of a nominal project, or up as high as 115.6 times as high as a nominal project.

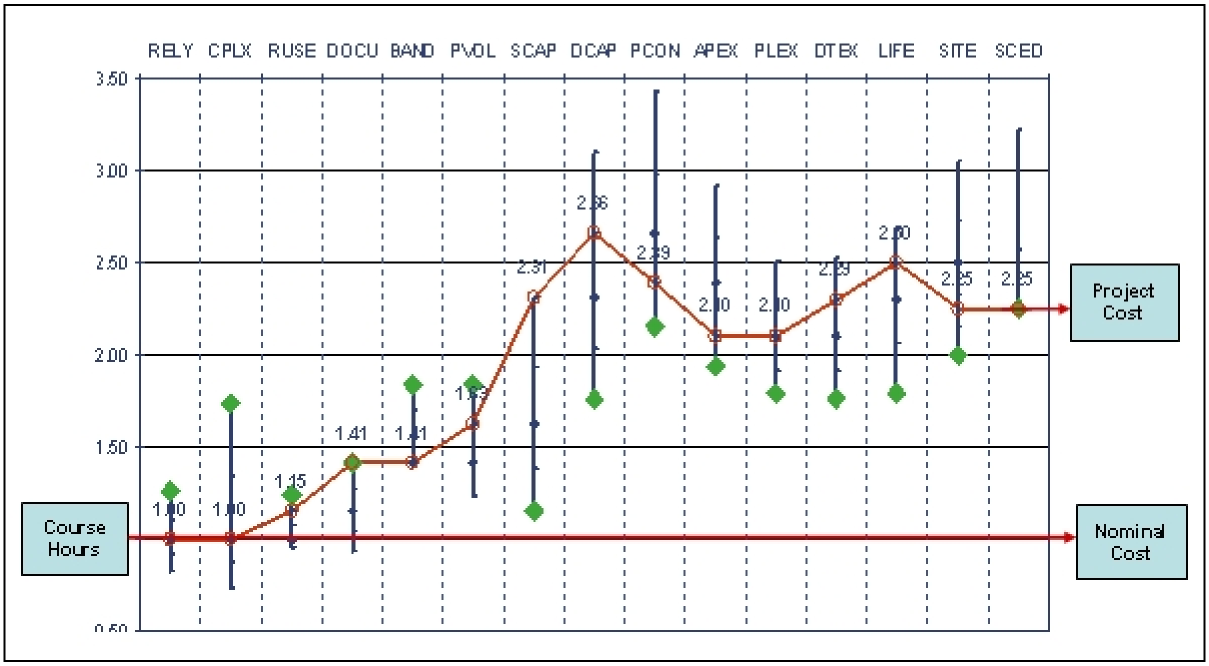


Figure 5. Impacts of Effort Multipliers on Cost Estimation.

Scale Drivers. The COSCOMO scale drivers are virtually identical to those used within COCOMO II. These five drivers make minor adjustments to the project size estimate, simulating economies or diseconomies of scale caused by the characteristics they rate. They account for the flexibility that the developers have in fulfilling project requirements; the precedentedness of the project; the clarity of requirements and resolution of risks; the cohesion of the team; and the maturity of the development process.

Constants. There are two calibration constants in the COCOMO II and COSCOMO models. These constants are conversion factors for turning the input in course hours into the output in person months of effort. For all COCOMO-based estimation models, these constants are derived from the analysis of historical project data. Our project is currently collecting this data from a number of military courseware development projects. The results of these development processes will serve as the calibration data set that allows us to determine the size of the two constants.

Current Model. The COSCOMO model is still under development. This paper is a description of the process that is being used to create it and the differences that have been identified between it and COCOMO II, more than it is a presentation of the final working model.

The current form of the model is given in the equation below. This model retains the fundamental form of the original COCOMO II model. Significant modifications may be necessary later in the project.

$$PM = A * (Size)^{E} * \prod_{i=1}^{15} EM_{i}$$

$$E = B + 0.01 * \sum_{j=1}^{5} SF_{j}$$

where
PM = effort in Person Months
A = calibration constant derived from historical project data
Size = adjusted number of courseware hours
EM_i = effort multiplier for the i^{th} cost driver in the equations
E = diseconomies of scale driven by five scale factors
B = calibration constant derived from historical project data. Can be customized based on a specific organization's historical project data.
SF_j = scale factor for the j^{th} scale driver. Provides project-specific adjustments to the size of the project.

While there may be many unknowns during the estimation phase, such as delivery medium, level of interactivity, level of instruction, instructional strategies, metadata requirements, and student tracking requirements, the COSCOMO user will be able to explore "what-if" scenarios for cost comparisons. Input data used during the estimation process may also be captured for future reference and used in the government's Request for Proposal (RFP) development process.

Validation. Model developers typically validate these models through comparison with historical data, interviews with experts, and the application of a Delphi process to achieve close agreement on the possible variable values (Boehm 2000).

For COSCOMO we intend to validate the algorithms with a number of historical projects that have been conducted by the Joint Advanced Distributed Learning Co-laboratory (Joint ADL Co-Lab) and by companies in the industry. Each project data set contains its own unique situations that need to be considered in applying the COSCOMO model. For example, as mentioned earlier, some projects do not include all of the ADDIE phases, while others do.

Validation has presented itself as a significant issue in determining the reliability of the model. Models with a large number of variables require a significant set of data points to validate them. COCOMO II contains 22 variables and COSYSMO contains 18. The current version of the COSCOMO model has 20 variables. The COCOMO project at USC has been collecting project data for over 20 years and is based on a database of 184 projects. COSYSMO has been under development for only four years and has a database of 42 projects (Valerdi 2005). We estimate that we will need approximately 80 to 100 historical data points to validate the model.

Future Work. The flexibility of the tool will allow it to be easily adapted for future technologies as they emerge. Future adaptations of the tool could include variations of SCORM projects such as life-cycle maintenance, new SCORM versions, and deployment on personal digital assistants (PDAs) or other alternative systems. Advanced simulations could be explored as well as blended learning projects.

Conclusion

At the conclusion of this project, the government will receive a working software product that can be used by the Joint ADL Co-Lab and its partners to generate staffing and schedule estimates for specific projects. The tool will query the user for the necessary information and provide sufficient explanatory information to allow a user to create estimates without becoming an expert in the COSCOMO algorithms – exactly as the COCOMO II products do for software project managers. Government and industry project managers will receive enough information through the interface to select ratings for specific variables with confidence.

The current version of COSCOMO is a first step in providing a cost estimation tool for the interactive courseware community. Like other such models, we expect that the model will continue to evolve toward two different goals. First, its ability to predict project costs accurately will improve as it is calibrated against more data sets. Second, it will be broadened to deal with some of the unique processes and products that occur among courseware projects.

References

Alexandria ADL Co-Laboratory, "The SCORM Implementation Guide: A Step by Step Approach", Release Date: November 27, 2002. Downloaded from `http://www.adlnet.org/downloads/187.cfm`, July 2003.

Boehm, B., *Software Engineering Economics.* Prentice Hall, New York, 1981.

Boehm, B., "Safe and Simple Software Cost Analysis," *IEEE Software*. IEEE, Cupertino, CA. Sept/Oct, 2000.

Boehm, B., "Software Estimation Perspectives," *IEEE Software*. IEEE, Cupertino, CA. Sept/Oct, 2000.

Boehm, B., Abts, C., Brown, A. W., Chulani, S., Clark, B. K., Horowitz, E., Madachy, R., Reifer, D., Steece, B., *Software Cost Estimation with COCOMO II*. Prentice Hall, New York, 2000.

Clark, B., Devnani-Chulani, S., Boehm, B., "Calibrating the COCOMO II Post-Architecture Model," *Proceedings of the 20th International Conference on Software Engineering*, 1998.

Boehm, B., Valerdi, R., Lane, J., Brown, A., "COCOMO Suite Methodology and Evolution," *Crosstalk*. Hill AFB, UT, May 2005.

Department of Defense, "Instructional Systems Design and Standards Approach to Training (MIL-HDBK-29612-2A)", 2002.

Department of Defense, "Strategic Plan for Transforming DoD Training," Office of the Under Secretary for Personnel and Readiness Director, Readiness and Training Policy Programs, 2002.

Musilek, P., Witold P., Giancarlo S.; Nan S., "On the Sensitivity of COCOMO II Software Cost Estimation Model," *Proceedings of the Eighth IEEE Symposium on Software Metrics*, 2002.

Biography

Roger Smith is a Chief Engineer with SPARTA Inc. working on next-generation simulation architectures and management tools. He is the creator and instructor for a series of simulation courses that have educated hundreds of simulation professionals. He serves on advisory boards for Sandia National Laboratories, University of Florida, and University of Central Florida, and on the editorial boards of the *ACM Transactions on Modeling and Computer Simulation* and the *IRI Research Technology Management*. He holds degrees in Mathematics, Statistics, Computer Science, and Management.

Lacey Edwards is a Systems Engineer with SPARTA Inc. investigating new combat system infrastructures and cost management algorithms. She earned her B.S. in Computer Engineering from the University of Florida and her M.S. in Management Science and Engineering from Stanford University.