

A COMPARATIVE STUDY OF MODEL CONTENT AND PARAMETER SENSITIVITY OF SOFTWARE SUPPORT COST MODELS

Daniel V. Ferens, Kevin L. Brummert, and Philip R. Mischler, Jr.

ABSTRACT

Software support costs frequently account for about seventy percent of software life cycle costs. This is because software support involves more than corrective support, or error correction (sometimes called “maintenance”). Software support also includes adaptive support, which includes modifying software for a new operating system or database, and perfective support, which includes enhancing the software’s performance or capabilities.

Despite the predominance of support costs in software life cycle costs, attempts to estimate these costs have historically been lacking. Most parametric cost models do not address software support costs to the same level of sophistication as development costs. Additionally, software support cost models have not yet been shown to be accurate (Ferens, 1984). Furthermore, as illustrated by Boulware, Nethery, and Turner, software cost models vary substantially in the way they address software support costs (Boulware, 1991).

In 1998, a thesis effort (Brummert, 1998) performed at the Air Force Institute of Technology (AFIT) rigorously compared the software support cost capabilities of five popular software parametric cost models: PRICE-S, SEER-SEM, SoftCost-OO, SoftEst (a COCOMO variant), and SPR KnowledgePLAN (a successor to the CHECKPOINT model). The objectives of the thesis were to determine how differences in models affect resulting support cost estimates, and whether the differences can be explained and, consequently, adjusted in order to “normalize” models to give equivalent estimates. A hypothetical case used by Coggins and Russell to assess development cost capabilities of selected cost models was also used for this study (Coggins, 1993). Accuracy of the models was not assessed.

The results of the thesis effort showed that the differences among the five models did significantly affect support cost estimates. Given equivalent inputs, the five models varied substantially in both cost estimates and in support profiles, or how effort is allocated annually during a twenty-year support period. As in the Coggins and Russell study for development costs, a conclusion of this thesis is that models can not be normalized for support costs. An analyst should use one or two models (and learn them well) instead of trying to normalize the results of several models.

INTRODUCTION

Software support costs comprise a substantial portion of most system costs and usually exceed software development costs. According to renowned software expert Dr. Barry Boehm, software support costs usually comprise from 60 to 75 percent of software life cycle costs (Boehm, 1981, Chapter 30). Lawrence Putnam, another prominent software authority, states that software support costs comprise about 61 percent of software life cycle costs (Putnam, 1992, Chapter 3). Noted software champion Lloyd Mosemann states that software support costs account for 60 to 80 percent of software life cycle costs (Mosemann, 1996).

Despite their dominant role in software life cycle costs (and sometimes in total system-level costs), software support costs have not been given adequate attention. Most commercial software cost models provide sophisticated development cost and schedule estimation capabilities, but vary widely with respect to software support cost estimating. Some models provide very little support cost capability; they only contain one or two support-unique inputs and appear to just treat support as an extension of development. Other models have more detailed support-unique inputs and outputs, but even their support cost features and capabilities pale in comparison with their development cost capabilities. Cost models also differ in how they define "support". Some models treat it as merely error correction, or "maintenance", while others include various enhancements and modifications.

This paper discusses the status of software support cost estimation, with particular attention to a thesis effort performed at the Air Force Institute of Technology (AFIT) in 1998 by Brummert and Mischler (Brummert, 1998). First, software support is briefly explained to show why it is costly. Next, a history of software support cost models is presented. Next, a summary of independent studies related to software support cost estimating is presented. Next, the 1998 thesis effort will be discussed in some detail. Finally, the present and future status of software support cost estimating is described.

SOFTWARE SUPPORT EXPLAINED

After software has been developed, it must be maintained or, more specifically, supported. Software support involves much more than traditional "maintenance", or correcting errors not discovered during testing. It also involves adding additional capabilities, deleting obsolete capabilities, modifying the software to address a change in the environment or to better interface with the host computer, and other activities that are necessary after software is developed. In fact, since software support usually involves reiterations of software developmental phases of requirements analysis, design, coding, and testing, software support is occasionally called "software redevelopment". This term is infrequently used, however, since many agencies such as the United States Air Force (USAF) fund development and support separately.

Efforts termed as re-development, therefore, would not receive needed support funding.

According to Glass and Noiseux, almost all software support can be partitioned into three categories: corrective, adaptive, and perfective. Figure 1 briefly explains each category and gives the relative percent of support effort expended for each category. Note that corrective support, the closest analogy to traditional "maintenance", accounts for only 17% of the overall support effort

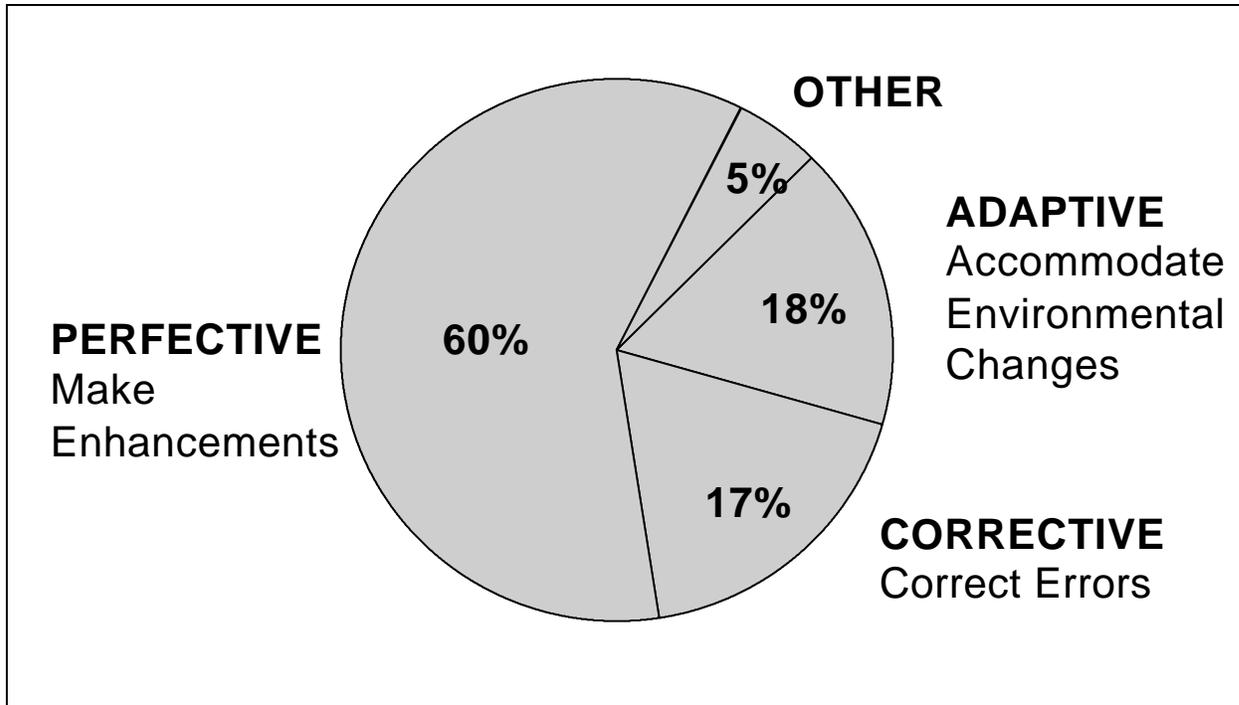


Figure 1: Categories of Software Support (Glass, 1981)

Altogether, these activities are quite extensive, and expensive. In addition, some agencies include preventive software support activities, such as restructuring, reverse engineering, and re-engineering, in software support costs. These activities are becoming popular out of necessity because of the current "Year 2000" enigma. It is a small wonder, then, that being able to accurately estimate and track software support costs would be a benison to many organizations.

A BRIEF HISTORY OF SOFTWARE SUPPORT COST ESTIMATION

Software parametric cost models were virtually unknown until 1977, when RCA PRICE Systems released their proprietary PRICE Software (PRICE S) model for general use on a time-sharing service. Shortly thereafter, in 1979, QSM Corporation introduced their proprietary Software Lifecycle Model (SLIM) for use on a time-sharing

basis. Although these models were very sophisticated in their ability to estimate software development costs and schedules, their ability to estimate software support costs was virtually non-existent.

In 1979, the Air Force Avionics Laboratory managed a four-year contracted effort to develop what eventually became the Avionics Software Support Cost Model (ASSCM). This model was based on ten historical support programs from Air Force Air Logistics Centers, and considered those factors which, in the opinion of those at the Air Logistics Centers, affected software support costs. The model was released in 1983 (SYSCON, 1983), but quickly evanesced into oblivion because cost estimating was de-emphasized in the Avionics Laboratory at that time. Meanwhile, RCA PRICE Systems released the PRICE-S Life Cycle (PRICE-SL) model in 1980, and, in 1981, Dr. Barry Boehm published the Constructive Cost Model for Maintenance (COCOMO-M) in his classic book, *Software Engineering Economics* (Boehm, 1981). QSM included a life cycle option in SLIM, although it was limited in several aspects. In 1983 there were several models available for software support cost estimation.

Although software support cost models were now available, their ability to accurately predict software support cost was unproven, at least from independent assessments. For example, SYSCON reported that ASSCM estimates were within 17% of actual effort for three projects not used in developing the model; however, an independent analysis did not demonstrate similar accuracy. This analysis, published in 1984, is described below.

Since 1984, several other software support cost models have become available. In the mid-1980s, SEER-SEM was initially marketed by Galorath Associates. SEER-SEM has a relatively extensive support cost capability, including more than ten support-unique inputs. The SoftCost-OO model, released in the mid 1980s by Reifer Consultants, Inc., had several support-unique inputs and had the capability to estimate blockchange-like applications. Here, effort and schedule are fixed and the amount of work that can be done (e.g., number of changes) is computed. The Software Architecture, Sizing, and Estimating Tool (SASET), developed by Martin Marietta Corporation for the United States Navy's Cost Analysis Center, had more than ten support-unique inputs. The CHECKPOINT model, released in the early 1990s by Software Productivity Research, also has a comprehensive software support cost capability with more than fifteen support-unique inputs. The new COCOMO 2.0 considers additional support factors to those in the original (1981) COCOMO, and CostExpert, a new software cost model marketed by Marotz, Inc. considers numerous factors for software support estimation.

While these models demonstrate that software support costs are important and consider many support-unique parameters, they still have not been shown to be accurate. Furthermore, the models differ considerably in their definitions of software support, the factors considered, their assumptions about staffing during the support

period, and their estimates of effort in dollars or person-months. Some independent assessments of software support cost models are now synopsized.

The 1984 “Quo Vadis” Study

As stated above, for the ASSCM model, independent analysis did not replicate SYSCON’s results. A study by Lt. Petersime of Air Force Logistics Command for three other programs did not demonstrate significant accuracy for ASSCM. The author then estimated the support effort of these three other programs using COCOMO-M and PRICE-SL to determine if these models were more accurate than ASSCM. Also, since software support may be “re-development”, the author ran PRICE-S for these programs. The results of this study, summarized in Table 1, were presented at the 1984 International Society of Parametric Analysts (ISPA) Conference and published in the *Journal of Parametrics* (Ferens, 1984).

Table 1
Software Support Cost Accuracy Study Results (Ferens, 1984)

<u>Program</u>	<u>Actual PM</u>	<u>Estimated PM</u>			
		ASSCM	COCOMO-M	PRICE-SL	PRICE-S
EW Receiver	44.85	39.61	15.30	98.98	250.70
EW Integrated	91.90	54.09	37.50	200.30	850.80
Navigation OFP	70.44	17.29	2.32	43.99	34.70

In Table 1, the three programs analyzed, the electronic warfare (EW) receiver program, the EW integrated programs (which has multiple EW functions), and the navigation operational flight program (OFP), were block changes managed at Air Logistics Centers. Block changes are blocks of time (for example, 18 months) where the assigned personnel make as many changes as possible to the software. The block changes in Table 1 had time periods were between 18 and 20 months, and sizes between 15,000 and 21,000 lines of assembler code.

Table 1 shows that the models were not accurate in predicting effort for the three programs studied. ASSCM was close for the EW receiver block change; however, further investigation revealed that ASSCM was inaccurate for individual support stages of this program, and the seemingly accurate result was merely from positive and negative errors canceling out. Actually, this study did not prove models to be inaccurate, since it was based on a small sample and the validity of the actual effort may be questionable. Furthermore, block changes are difficult to estimate. They usually have a fixed level of effort and schedule while the work performed varies. However, most software cost models, however, estimate effort based on a fixed level of work to be done. On the other hand, the study showed that accuracy could not be established at least for the programs investigated. Unfortunately, no other studies have refuted the findings of this one; they have not demonstrated accuracy for any software support cost models.

The 1991 Boulware, Nethery, and Turner Study

As stated above, software support cost models also treat support differently. These differences were first highlighted in a 1991 article by Boulware, Nethery, and Turner published in the *National Estimator* (Boulware, 1991). In this study, four software support cost models, SLIM, SASET COCOMO-M, and PRICE-SL, were compared as to (1) the contents of software support costs, and (2) the ways they assigned annual effort during a twenty-year support period. The models varied, sometimes substantially, in both of these considerations. Model accuracy was not addressed in this study.

For the contents comparison, the models address support costs as follows:

- a. SLIM: Assumes support costs are 45% enhancements, or major changes, 40% modifications to optimize or improve existing software, and 15% error correction.
- b. SASET: Assumes 42% error correction, 50% adaptation (to new hardware or system software), and 8% prevention. These are “in-scope” maintenance activities. Enhancements are treated as “out of scope” activities which are not normally included in a SASET “maintenance” estimate.
- c. COCOMO-M: According to Boehm, COCOMO-M assumes 21.7% for error correction, 41.8% for enhancements for users, 23.8% to accommodate changes to hardware or data files, 5.5% to improve documentation, 4.0% to improve code efficiency, and 3.4% for other changes (Boehm, 1981, p. 548).
- d. PRICE-SL: This model assumes all software support falls in one of three categories: maintenance, enhancements, and growth. Maintenance is correcting latent software defects; enhancements are improvements to the efficiency, performance, or portability of existing code; and growth is the addition of new software functions or capabilities beyond the original development specifications. The percentage of support for each category depends on user inputs for enhancement level, quality level, and anticipated growth rate.

Although there is some commonality among the four models, there are some significant differences in the definitions of support (or maintenance) and the percentages for the various categories of support. There are even more marked differences, however, in the way the models assume the distribution of effort during a support period. Figure 2 shows the effort in person-months (PM) by year for each model for a twenty-year support period for the same (sample) program.

Figure 2 shows the differing assumptions the models make about effort during the support period. SLIM assumes that effort is very high initially, then decreases rapidly after the first few years. SASET assumes a continuously declining level of

support throughout the period. COCOMO-M computes only annual support effort in PM, and assumes that effort remains constant during the entire support period. Finally, PRICE-S assumes that effort increases during the first several years, then decreases gradually.

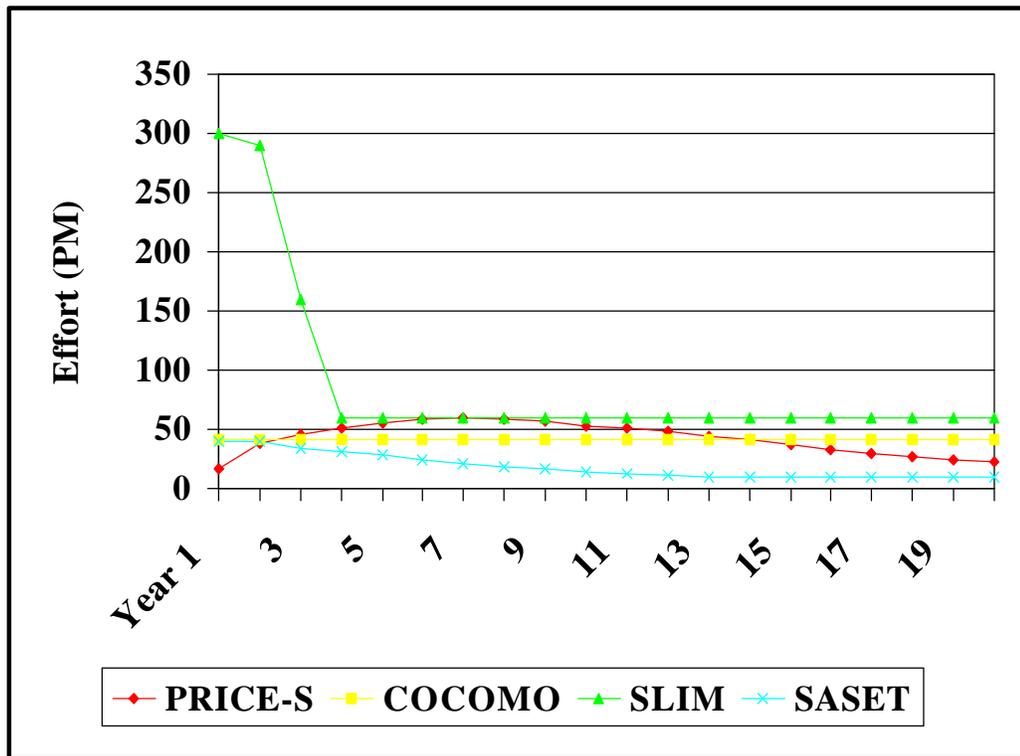


Figure 2: Software Support Staffing Profiles (Boulware, 1991)

These divergent profiles indicate that certain models may be more suitable for certain types of support. SLIM seems best suited to business applications where software is used for a few years, then another version replaces it. SASET may be most suitable where corrective support is predominant for a long period of time, such as a legacy program. COCOMO-M seems to be most suitable for situations where a constant level of support effort is envisioned, such as the blockchanges discussed earlier. PRICE-SL appears to be suitable for applications where deficiencies are corrected as they are discovered. The assumption appears to be that the early portion of the support period is a period of learning and discovering deficiencies, while the later portion assumes deficiencies will decrease.

None of the models' assumptions or definitions are incorrect; the models are just different. These differences reflect varying and, sometimes, divergent philosophies of software support. Therefore, the user of a model must consider the envisioned support environment and determine whether the model is suitable for that environment. Selecting a model without considering the assumptions is misuse of the model.

Boulware, Nethery, and Turner conclude their study with the following statement: "It appears that the more one knows about the various underlying model assumptions, the less comfortable an estimator feels about performing a maintenance estimate" (Boulware, 1991, p. 24). The study seemed to highlight more problems than solutions in the area of software support estimation.

The 1993 Coggins and Russell Study

This study (Coggins, 1993) investigated software development cost models and did not address software support costs. It is being included here, however, because it was the forerunner to the Brummert and Mischler effort discussed later. This study was an AFIT thesis effort sponsored by the Air Force Cost Analysis Agency (AFCAA).

The objective of this study was to analyze four prevalent software cost models as to the different assumptions made by the models, how these assumptions affected development cost or effort estimates, and whether these assumptions could be adjusted so the models would produce equivalent estimates. It was thought at that time that perhaps software model results could be "normalized", or made equivalent, such that, for example, a PRICE-S estimate could be directly compared or even equated to a SEER-SEM estimate. The four models analyzed were REVIC (a COCOMO variant managed at AFCAA), SASET, PRICE-S, and SEER-SEM.

With regard to software development effort, it was found that the four models differed notably in the software development phases considered, development activities considered, key attributes and cost drivers, effects of different programming languages, implications of size, effects of schedule compression and expansion, and model database characteristics. Furthermore, when a hypothetical test case was used, the four models gave significantly different estimates for the same case.

The test case was a project consisting of three computer software configuration items (CSCIs), which had all new design and code. The project was an avionics system using tailored documentation for a then-prevalent Department of Defense (DOD) Standard, DOD-STD-2167A. The waterfall model of software development was assumed, and an "average" level of integration for each CSCI was assumed. The project was estimated from system software requirements analysis through system testing. A workload of 152 person-hours per month was assumed. CSCI #1 was programmed in the Ada language by average personnel with nominal tools. It contained two components of 20,000 and 30,000 lines of Ada code. CSCI #2 contained 80,000 lines of assembly language, and was programmed by above average programmers with nominal software tools. CSCI #3 contained 45,000 lines of Ada code, and was programmed by average personnel with nominal software tools.

The four models were run using "equivalent" inputs from the test case. Table 2 summarizes the test case results for each model in development person-months (PM).

As is evident from Table 2, the estimates of the four models differed substantially in estimated effort values and in relative effort for individual CSCIs. For example SASET and PRICE-S gave higher estimates for CSCI #1 and #3 than SEER-SEM and REVIC, but lower estimates for CSCI #2.

Table 2
Development Cost Model Test Case Results (Coggins, 1993)

<u>Level</u>		<u>Estimated</u>	<u>PM</u>	
	REVIC	SASET	PRICE-S	SEER-SEM
Project	1648.5	2643.3	3161.5	2366.4
CSCI #1	416.0	1104.2	1429.5	719.4
CSCI #2	837.2	545.3	520.2	1013.0
CSCI #3	376.7	993.8	1211.8	634.0

The conclusion of the study was that models varied widely in many respects, and could not be “normalized”. Instead, analysts should become thoroughly familiar with one or two cost models, and use these models for software estimates.

Stukes’ 1996 Post Deployment Software Support (PDSS) Study

During the mid 1990s, Sherry Stukes, noted software estimating researcher and practitioner at MCR Corporation, performed a study for AFCAA on post-deployment software support (PDSS) estimation methods. Some of the highlights of this study, published in an MCR report (Stukes, 1996) and condensed in an article in the Summer, 1998 *National Estimator* (Stukes, 1998), are presented here.

The study first discussed the dilemmas associated with software PDSS cost estimation. It showed that, like the 1984 Quo Vadis study, some USAF agencies have fixed budgets and schedules and must estimate the number of changes that can be done within these constraints. Most software cost models, however, compute PDSS cost and (sometimes) schedule based on software technical and managerial characteristics such as size, complexity, and support environment. Furthermore, PDSS is defined differently by various organizations and cost models. Stukes proposed a standard definition for PDSS, at least for Government use. The overall definition is “the total cost of maintaining a software program for its entire life once it has been delivered to its operational site” (Stukes, 1998, p. 4). In addition, Stukes listed four components of PDSS costs: error correction, adaptation to requirements, perfective improvement, and enhancements.

Stukes then analyzed seven cost models and compared the ways they estimate PDSS costs. These models were REVIC, SASET, SEER-SEM, PRICE-S, SLIM, SoftCost-R, and CHECKPOINT. It was found that SEER-SEM, PRICE-S, and CHECKPOINT estimated costs for all four PDSS components, while the other models concentrated on error correction activities. Moreover, each model had unique inputs

and methods for computing PDSS costs. One important conclusion from this study is that the models do not sufficiently address the situations many organizations face, such as blockchanges, where effort and schedule are fixed.

THE 1998 BRUMMERT AND MISCHLER THESIS

This thesis effort, sponsored by AFCAA, was a follow-on effort to the Coggins and Russell study (Coggins, 1993) discussed earlier. This thesis addressed software support costs whereas the Coggins and Russell thesis examined development costs. The overall objectives of this thesis effort were to assess differences in software support cost models, and to determine to what degree the differences could be explained and adjusted for possible “normalization” among models. As in all the studies described above except the 1984 Quo Vadis study, accuracy was not addressed.

Thesis Methodology

The AFCAA requested that five models be analyzed for this study. The models were PRICE-S, SEER-SEM, Object-Oriented SoftCost (SoftCost-OO), SoftEst (a successor to REVIC), and KnowledgePLAN, a model developed by Software Productivity Research as a successor to CHECKPOINT. The researchers gained knowledge about the content and execution of the five models from published reports, user manuals, telephone calls and personal interviews, and hands-on use.

The researchers evaluated the models using a checklist containing several questions to be answered about each of the five models. These included the following:

- a. What are the unique input parameters that directly affect software support cost?
- b. What is the estimating methodology used?
- c. What are the underlying algorithms?
- d. What are the underlying bases for parameter values?
- e. Is support partitioned into sub-categories (e.g., adaptive, corrective, and perfective)
- f. What support time periods can be considered?

Since some of this information was not readily available for the proprietary PRICE-S and SEER-SEM models, especially regarding methodologies and algorithms, the researchers personally interviewed the responsible agencies for these models. The information presented in the findings below is partly based on these interviews.

The researchers then ran the models using the hypothetical scenario from the Coggins and Russell effort (Coggins, 1993) described earlier. This scenario was used because this effort was a follow-on to the Coggins and Russell effort, and the

researchers wanted to avoid accuracy comparisons among the models (which may be enticing if a real project was used).

Findings

Table 3 summarizes the primary support-unique cost factors and those developmental parameters that most strongly affected support costs for each of the five models. Table 3 shows that some models, such as SEER-SEM, had more support-unique cost factors than others such as SoftEst. Although the number of these factors may indicate the thoroughness of support considerations, the number of factors should not be used as a basis for model selection. As concluded in the Boulware, Nethery, and Turner study (Boulware, 1991) discussed earlier, different models may be more suitable for different situations.

Table 3 also shows that the different models consider different factors. Even when similar factors are present in two or more models, they may not affect costs in the same way. For example, security requirements are a very strong cost driver in SEER-SEM, but only have a maximum range of ten percent in SoftEst.

Table 3
Software Cost Model Support Factors (Brummert, 1998)

Model	Support-Unique Cost Factors	Developmental Factors Most Strongly Affecting Support Cost
PRICE-S	<ul style="list-style-type: none"> - Years of support - Productivity factors for maintenance, enhancements, and growth - Anticipated quality, enhancement, and growth levels - Number of installations 	<ul style="list-style-type: none"> - Size - Application - Platform - Utilization (machine constraints) - Language used - Internal integration
SEER-SEM	<ul style="list-style-type: none"> - Years of support - Separate sites - Support growth over period - Personnel and Environment Differences - Annual change traffic - Support level (Rigor) - Support staffing constraints - Percent supported - Whether total system is supported 	<ul style="list-style-type: none"> - Size - Security requirements - Rehosting requirements - Modern development practices use - Requirements volatility - Time constraints - Analyst and programmer capability - Analyst's applications experience - Development staffing complexity
SoftCost-OO	<ul style="list-style-type: none"> - Years of support - Annual change traffic - Sustaining engineering factor - Cost of money 	<ul style="list-style-type: none"> - Size - Technology usage factor - Scope of (developmental) support - Reuse benefits - Analyst capability
SoftEst	<ul style="list-style-type: none"> - Years of support - Annual change traffic - Software understanding* - Software assimilation effort* 	<ul style="list-style-type: none"> - Size - Required reliability - Requirements volatility - Product complexity - Virtual machine volatility
KnowledgePLAN	<ul style="list-style-type: none"> - Maintenance personnel staffing, experience, and education - Maintenance platform support - Release control methods - Problem tracking and reporting - Replacement and restructure planning - Central and field maintenance - Customer and delivery support - Software warranty coverage - Installation and production geography - Number of system installation and maintenance sites - Annual growth in installation sites - Program execution frequency - Current system status - Long range product stability 	<ul style="list-style-type: none"> - Size - Language - Base code origin - Age of base code - Support responsibility - Base code status

* These parameters were not functional in SoftEst version analyzed

Table 4 shows the types or categories of software support considered by each of the five models. These reflect some of the findings of the Boulware, Nethery, and Turner study, and the Stukes study (Stukes, 1998). Again, this information is for

comparison and not for rank-ordering the models. It is noteworthy, however, that the categories of SEER-SEM are those proposed in Stukes' "standard" definition.

Table 4
Support Categories Considered By Models (Brummert, 1998)

Model	Software Support Categories
PRICE-S	3: Maintenance, enhancements, and growth
SEER-SEM	4: Corrective, adaptive, perfective, and enhancements
SoftCost-OO	2: Maintenance and sustaining engineering
SoftEst	1: (Model does not partition cost into categories)
KnowledgePLAN	7: System deployment, defect reporting preparation, defect reporting execution, defect repair, field service, customer support, maintenance management

The models were run for the hypothetical scenario, described earlier, for the overall project and for each CSCI. Table 5 shows the project-level estimates for each of the five models for an entire 20-year support period and for the first 5 years of a 20-year support period. Since the version of KnowledgePLAN (Version 2.0) used in this study only computed support costs for a five-year period, only an estimate for the first five years is provided.

Table 5
Summary of Support Cost Estimates (Brummert, 1998)

Model	PRICE-S	SEER-SEM	SoftCost-OO	SoftEst	KnowledgePLAN
20-Year Costs (\$M)	34.9	101.3	57.6	91.2	(N/A)
First 5 Years Costs (\$M)	14.3	36.9	15.0	25.9	20.8

The results shown in Table 5 parallel those of the 1993 Coggins and Russell study (shown in Table 2) in that the models predict different costs for the same scenario. Also, the models apparently assume different ratios between support costs and development costs. For example, in Table 2, PRICE-S predicted higher costs than SEER-SEM for project development, but SEER-SEM support costs were much higher than those for PRICE-S.

Figure 3 shows the distribution of costs during the 20-year support period, and is similar to Figure 2 from the Boulware, Nethery, and Turner study. In Figure 3, PRICE-S and SEER-SEM had partial support duration in the first and twenty-first years; therefore, their support cost profiles are different than what would be expected for full years. Also, as discussed above, KnowledgePLAN only computed support costs for five years.

The results shown in Figure 3 are similar to those shown in Figure 2 for the Boulware, Nethery, and Turner study in that the models vary extensively in their assumptions about cost or effort distribution over time during software support.

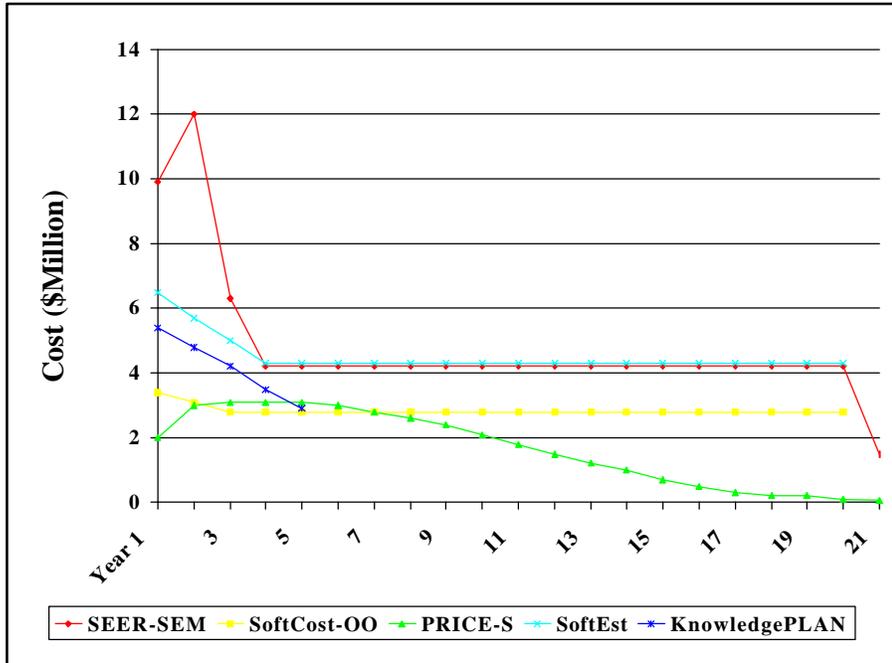


Figure 3: Software Support Cost Profiles (Brummert, 1998)

The overall results of this study, as summarized in Tables 3, 4, and 5, and in Figure 3, show that software support cost models differ in myriad ways, and a direct comparison, or normalization among the models is virtually impossible. The conclusion is similar to that of Coggins and Russell: It is best to learn one or two models well than to try to normalize the results of several models.

WHERE DO WE GO FROM HERE?

It should now be evident that the problems associated with software support are labyrinthine. Unfortunately, it does not appear that any climactic solutions are forthcoming. A recent assessment of software support cost estimation capabilities by DOD showed that our ability to estimate software support costs was color-rated “red” for DOD electronic, fixed-wing aircraft, and space systems (Balut, 1999). This rating means that the analyst can expect poor data, large errors, and the available models to be of little use. It should be noted that the rating for software development cost estimation was rated between “red” and “yellow”, which means that moderate to large errors could be expected for these estimates.

Software support data is an area that is especially problematic. Data sources are meager and, even when data is available, the quality and thoroughness of the data is questionable. For example, the Air Force’s Space and Missile Systems Center Software Database (SWDB) contains detailed information for over 2,600 Air Force

software programs, including developmental effort for more than 500 programs (Tinkler, 1997). However, the SWDB contains software support effort for only 13 programs, and the information for most of these programs is sketchy. In the comprehensive Air Force Total Ownership Cost (AFTOC) database, only coarse aircraft-level software “maintenance” cost data is available (Kunc, 1999). The contractor for the AFTOC database has admitted that even this data was difficult to obtain. Many other software cost databases do not contain any software support cost or effort data.

It does not appear that any breakthroughs will be achieved in the near future for software support cost estimation. The paucity of data, considerable differences among models, the lack of any demonstrated model accuracy, and the lack of agreement as to what constitutes software support are thwarting success in this arena. Furthermore, as the DOD study and a published article by the author (Ferens, 1998) showed, there are accuracy problems with software development cost models, for which progress is much more advanced than software support costs.

The situation is not entirely bleak, however. Detailed software support cost data was collected for the ASSCM model in the early 1980s, and the fact that some coarse data was collected for the AFTOC database and for 13 projects in the SWDB indicates that at least cursory attention is being given to collecting support cost data. Some recently released software cost models, such as Jensen’s SAGE model and the CostExpert model, are addressing software support costs in novel ways. Also, the studies discussed in this paper have revealed the issues that need attention, and an entrepreneurial software model developer or cost analyst may research these issues further in search of future profits. Furthermore, since software support cost issues are primarily managerial (Hughes, 1991), top-level management sponsorship of efforts to collect better data or develop better estimation techniques would be a benison. This sponsorship may occur as managers gain a more thorough understanding of the importance of software support and its currently fuliginous state. Still, it may be some time before we witness a quantum leap or “silver bullet” in the area of software support costs.

REFERENCES

- Balut, Steven, "Update on the Status of DOD's Capabilities to Estimate the Costs of Weapon Systems", Panel presentation at the Annual DOD Cost Analysis Symposium, Williamsburg, VA, February 2-5, 1999.
- Boehm, Barry W., *Software Engineering Economics*, Englewood Cliffs, NJ, Prentice-Hall, 1981.
- Boulware, Gary W., Belinda J. Nethery, and Bryan D. Turner, "Maintenance Software Model Assumptions Versus Budgeting Realities", *National Estimator*, Spring, 1991, pp. 13-25.
- Brummert, Kevin L., and Philip R. Mischler, Jr., *Software Support Cost Estimating Models: A Comparative Study of Model Content and Parameter Sensitivity*, AFIT/GCA/LAS/98S-3, Air Force Institute of Technology, Wright-Patterson AFB, OH, 1998.
- Coggins, George A., and Roy C. Russell, *Software Cost Estimating Models: A Comparative Study of What the Models Estimate*, AFIT/GCA/LAS/93S-4, Air Force Institute of Technology, Wright-Patterson AFB, OH, 1993.
- Ferens, Daniel V., "Software Support Cost Models: Quo Vadis?", *Journal of Parametrics*, Volume IV, Number 4, December, 1984, pp. 64-79.
- Ferens, Daniel V., and David S. Christensen, "Calibrating Software Cost Models to Department of Defense Databases - A Review of Ten Studies", *Journal of Parametrics*, Volume XVIII, Number 2, November, 1998, pp. 55-74.
- Glass, Robert L., and Ronald A. Noiseux, *Software Maintenance Guidebook*, Englewood Cliffs, NJ, Prentice-Hall, 1981.
- Hughes, Max, "Abolish the Word 'Maintenance' ", *Software Maintenance News*, Volume 9, Number 7, July 1991, pp. 4-6.
- Kunc, Wendy, "Air Force Total Ownership Cost, Presentation at the Annual DOD Cost Analysis Symposium, Williamsburg, VA, February 2-5, 1999.
- Mosemann, Lloyd K., II, et al, *Guidelines for Successful Acquisition and Management of Software Intensive Systems*, Hill AFB, UT, Software Technology Support Center, June, 1996.
- Putnam, Lawrence H., and Ware Myers, *Measures of Excellence*, Englewood Cliffs, NJ, Prentice-Hall, 1992.

Stukes, Sherry, et al, *Air Force Cost Analysis Agency Software Estimating Model Analysis Final Report*, Thousand Oaks, CA, MCR Federal, Inc., September, 1996.

Stukes, Sherry, "Post Deployment Software Support (PDSS) Estimating Methods", *National Estimator*, Summer, 1998, pp. 3-7.

SYSCON Corporation, *Avionics Software Support Cost Model* (AFWAL-TR-82-1173), Washington, DC, SYSCON, 1 February 1983.

Tinkler, Shirley, and Sherry Stukes, *Space and Missile Systems Center Software Database User's Manual, Version 3.0*, Thousand Oaks, CA, MCR, 1997.

BIOGRAPHIES

Daniel V. Ferens is an Engineering Analyst at Air Force Research Laboratory at Wright-Patterson AFB in Dayton, Ohio, where he directs cost analysis efforts for Laboratory programs. He is also an Adjunct Associate Professor of Software Systems Management at the Air Force Institute of Technology (AFIT), Graduate School of Logistics and Acquisition Management, at Wright-Patterson AFB in Dayton, Ohio, where he teaches courses in software estimation and software management in general. He was the Advisor for the theses described in this paper. He is an active member of the Society of Cost Estimating and Analysis and a lifetime member of the International Society of Parametric Analysts. Mr. Ferens has a Master's Degree in Electrical Engineering from Rensselaer Polytechnic Institute, and a Master's Degree in Business from the University of Northern Colorado.

Kevin L. Brummert is a Captain in the United States Air Force assigned to the Airborne Warning and Control System (AWACS) Program Office as a weapon systems analyst at the Electronic Systems Center, Hanscom AFB, Massachusetts. He currently develops cost estimates in support of Air Combat Command's Modernization Investment Planning Process (MIPP) which is used for life cycle decisions impacting the \$8 billion multi-national AWACS program. Captain Brummert also performs earned value management analysis on contractor cost data, and provides in-depth financial management and cost expertise as a member of AWACS Communication and Navigation Integrated Product Team (IPT). Captain Brummert graduated in May 1995 from The University of Akron with a Bachelor of Science degree in Accountancy. In September 1998 He received a Master of Science degree in Cost Analysis from the Air Force Institute of Technology's Graduate School of Logistics and Acquisition Management.

Philip R. Mischler, Jr., is a Captain in the United States Air Force assigned to the Air Force's Space and Missile Systems Center at Los Angeles AFB, California. He received a Bachelor of Science degree from North Carolina A&T University in 1985, and was commissioned as an Air Force Officer during the same year. Captain Mischler served two years as a Financial Manager at Aeronautical Systems Center at Wright-Patterson AFB, Ohio. In September 1998 He received a Master of Science degree in Cost Analysis from the Air Force Institute of Technology's Graduate School of Logistics and Acquisition Management