

Comparative Analysis of Software Maintenance Modeling in the COCOMO II, SEER, SLIM and True S Cost Models

Donald J. Reifer, Reifer Consultants, Inc.

Jill Ann Allen, U. S. Army

Brian Fersch, U. S. Air Force

Barbara Hitchings, SAIC

James Judy, Office of Deputy Assistant Secretary of the Army (Cost and Economics) and

Wilson Rosa, U.S. Air Force Cost Analysis Agency

Abstract: This paper provides a snapshot of the findings, conclusions and recommendations a year-long Joint U.S. Army and Air Force study on the topic of software life cycle maintenance and cost modeling. The paper starts by defining the work that government life cycle support centers typically perform as part of their normal jobs using a Work Breakdown Structure (WBS). Next, the paper rates how well the popular COCOMO II, SEER, SLIM and True S cost models estimate each of the work tasks in this WBS. Gaps uncovered are then highlighted and rules of thumb for filling them are offered as the paper concludes with recommendations for providing fuller coverage for these important operations, maintenance and support activities.

1. Introduction

During the past year, our study team has investigated in-depth how software maintenance was being conducted across the aircraft, missile and space domains [1]. The investigation has canvassed over thirty major weapons systems projects via questionnaires and interviews to develop conclusions and recommendations. As part of this effort, we tried to understand what tasks software maintenance personnel actually perform day-to-day as part of their jobs. Based on these tasks, we next looked at how the work involved was estimated, budgeted and managed. We also looked at the metrics and cost estimating models used for this purpose. As our final thrust, we tried to gather hard data to bound the actual costs of maintenance and identify what the primary drivers were that influenced software productivity and product quality.

This paper highlights our findings relative to cost estimating models for maintenance. It looks at the scope of the models relative to the activities government software life cycle support centers perform in an attempt to determine where additional coverage is warranted. The paper also provides recommendations for improvement based on the results of our investigations.

2. Software Maintenance Study Results

During the past year, we have been investigating the work that government software life cycle support centers perform to keep weapons systems operational. We have visited numerous facilities and interviewed over thirty weapons systems projects to develop our conclusions. The results of these efforts are summarized in the form of the Software Maintenance Work Breakdown Structure (WBS) that appears as Table 1. Each major activity performed in this life cycle stage is broken down into its component tasks within this Table whenever possible. The purpose of this breakdown is to identify what maintainers do in each of the major types of activities that are being performed.

This Table is important because it highlights the fact that software people do a great deal more than just software maintenance as part of their tasking in these government software life cycle support centers. This is important because the cost models used for estimating these costs do not always generate predictions that cover the full scope of work involved. For this reason, we have

included definitions for key software maintenance terms in Appendix A. We provided these because we found that these standard definitions of maintenance as defined in ISO/IEC 14764.2006 and IEEE standards [2] only covered the block release and retirement processes.

WBS	Title	Description
1.0	Operations, Maintenance & Support	This entry collects the total cost associated with maintaining a system after it has been accepted by the customer (DD-250).
1.1	Maintenance	This entry summarizes the costs for updating and repairing elements of system.
1.1.1	Release Planning	This entry records the costs associated with developing plans, budgets and schedules for block releases.
1.1.2	Hardware Defect Repair	This entry records the costs for repairing hardware defects. Such costs include engineering and test ¹ .
1.1.3	Software Defect Repair	This entry records the costs for repairing software defects. Such costs include engineering and test.
1.1.4	Hardware Enhancements	This entry records the costs associated with developing hardware enhancements and making perfective changes. Such costs include both engineering and test ² .
1.1.5	Software Enhancements	This entry records the costs associated with developing software enhancements and making perfective changes. Such costs include both engineering and test.
1.1.6	Release Testing and Delivery	This entry records the costs for acceptance test and delivery of the release including the costs of integration and verification.
1.2	Sustaining Engineering	This entry summarizes the costs associated with sustaining operations in the field. It includes the costs of analysis and studies, emergency repairs and user handholding and support.
1.2.1	Analysis and Studies	This entry records the costs associated with the conduct of analysis and studies stemming from operational issues and problems.
1.2.2	Emergency Repairs	This entry records the costs associated with emergency repairs including those associated with development and delivery of patch releases to the field.
1.2.3	User Training & Support	This entry records the costs associated with providing user support and training.
1.2.3.1	Help Desk	This entry records the costs for manning and operating a help desk.
1.2.3.2	User Group Support	This entry records the costs for coordinating and conducting user group meetings and blogs and for maintaining a web site.
1.2.3.3	User Training	This entry records the costs for providing the training needed to maintain and operate the system as it evolves.
1.3	Independent Test & Verification	This entry summarizes the costs associated with independently verifying and validating the system as releases are prepared and released typically by third parties. Such verification activities can range from independent testing to detailed analysis of both designs and code on a separately maintained test-bench.
1.3.1	Test Planning	This entry records the costs associated with preparing test plans.
1.3.2	Test Preparation	This entry records the costs associated with developing test cases and scenarios and the related test tools needed to run them.
1.3.3	Test Conduct	This entry records the costs associated with conducting the tests, capturing results, verifying release requirements are satisfied and developing regression test baselines for use in revalidating the system when future changes are made.
1.3.4	Independent Analysis & Verification	This entry records the costs associated with performing the detailed analysis of designs and code needed to provide additional confirmation that requirements including those for security and safety have been satisfied.
1.4	Product Support	This entry summarizes the costs associated with maintaining the overall quality of the processes, products and supplier networks used by the system in operations, maintenance and support.
1.4.1	Configuration Management	This entry records the costs associated with configuration management including those associated with CCB operations and tracking configurations, spares, licenses and parts among various operational and support sites.
1.4.2	Quality Assurance	This entry records the Quality Assurance costs aimed at ensuring the quality and

		integrity of the processes used by the system for maintenance and support.
1.4.3	Supplier Management	This entry records the costs associated with maintaining liaison with suppliers including those that provide parts, spares and software licenses.
1.5	Information Assurance	This entry summarizes the costs associated with information assurance including those associated with product and computer network protection.
1.5.1	Protection Services	This entry records the costs associated with product protection including any associated with anti-tamper.
1.5.2	DIACAP	This entry records the costs associated with the periodic conduct of DIACAP accreditation and certification for those computer networks used to maintain and operate the system.
1.6	Acquisition Support	This entry summarizes the costs associated with providing oversight and support for acquisition management activities. Such costs often occur when managing third parties doing maintenance activities.
1.7	Operations Support	This entry summarizes the costs associated with supporting operations in the field including those costs associated with database maintenance, configuration and system administration.
1.8	Facility Support	This entry summarizes the costs needed to ready and maintain those development and test facilities needed to maintain and sustain the system in the field.
1.8.1	Maintenance Facility Sustainment	This entry records the costs related to readying and maintaining a maintenance facility that can be used to develop and sustain the system once it is fielded.
1.8.2	SIL Sustainment	This entry records the costs associated with readying and maintaining a System Integration Lab (SIL) that is used to test and evaluate new releases destined for the field under realistic operating conditions (using tactical hardware in the loop).
1.8.3	Network Operations & Administration	This entry records the costs associated with managing and maintaining the networks used for maintaining, operating and supporting the system.
1.8.4	Security	This entry records the costs associated with security including those associated with implementing the security plan and maintaining any Sensitive Compartmented Information Facilities (SCIFs).
1.9	Field Support	This entry summarizes the costs conducting field support including both labor and travel components.
1.10	Management	This entry summarizes the costs associated with managing release and sustaining engineering activities and conducting metrics analysis.
1.10.1	Release Management	This entry records the costs associated with managing the generation of releases.
1.10.2	Sustaining Engineering Management	This entry records the costs associated with managing the sustaining engineering efforts for the system including those associated with independent testing; independent verification; acquisition, product, field, operations facility support; and information assurance.
1.10.3	Metrics Analysis	This entry records the costs associated with metrics data collection and analysis activities.
1.11	Parts	This entry summarizes the costs of acquiring, packaging, transporting and storing replacement parts, components and subassemblies.
1.12	Spares	This entry summarizes the costs of acquiring, packaging, transporting and storing spares.
1.13	Licenses	This entry summarizes the costs for software licenses. Such costs include the costs for the license along with those needed to maintain market watch and vendor liaison functions.
1.14	Cost Item General	This entry summarizes all other costs not tied to categories within this breakout.

Table 1 – Software Operations, Maintenance and Support WBS

Notes

- 1 Many times the maintenance team must first determine whether the trouble report is a hardware or software problem. When it is hardware, the team often winds up fixing it.
- 2 Often, software must be optimized when new hardware is retrofit into the configuration. Such costs are normally not included as part of the hardware upgrade.

Like most Work Breakdown Structures, the one shown in Table 1 should be viewed as a laundry list of tasks that can be performed by your software life cycle support center. Some may perform all of the tasks listed. Others may not. Some of the activities may involve new requirements that you may be considering like information operations. Other tasks like manning a help desk and providing user support may not be applicable. From an estimating point-of-view, models or heuristics must be made available to help us predict the costs of each of these activities just in case they are applicable. Otherwise, the budgets you will receive may be insufficient for you to get the work done in a timely manner. Equally important is the necessity to account for all of the labor that must be expended to get the total and not just part of the job done.

3. Popular Software Cost Models

The four most popular software cost estimating models used today seem to be COCOMO II [3], SEER [4], SLIM [5] and True S [6]. Each of these models has different scopes, mathematical formulations and assumptions. However, as Table 2 illustrates, each of them works in a similar manner when it comes to estimating software maintenance cost. For the most part, they assume that software maintenance is a subset of development. New releases are born when requirements change as new features and functionality are added to the software by the user. These releases are planned, developed, tested and fielded in order to incorporate these new requirements, fix bugs and make perfective changes that make the software operate quicker and better. Estimates are developed in the same manner for software maintenance as for new developments. The only difference is that the forecast is based on how much of the code changes (the code fragment) to accommodate requirements changes, fixes and perfective changes. As the estimate is prepared, estimators may also change some of the cost drivers to reflect their current situation. In some cases, maintenance estimates are derived like those for new developments. In others, they are estimated stand-alone in a native maintenance mode. In all of the models analyzed, you can combine development and maintenance estimates to provide a full life cycle cost estimate. In most of the models, you can also export the estimate to either spreadsheet or planning packages and allocate the estimated effort to tasks. Some of the vendors even offer multiple models; some of which handle one or more of the tasks in our WBS.

The maintenance capabilities of the four models studied are as follows:

- **COCOMO II** – The scope of software maintenance within the COCOMO II model includes adding new capabilities and fixing or adapting existing capabilities. It excludes major product rebuilds changing over fifty percent of the existing software. The model assumes that maintenance estimates have the same general cost and scale driver attributes as software development costs. There are special considerations listed in reference [3] for the schedule (SCED), reuse (RUSE) and software reliability (RELY) cost drivers. The maintenance size for the software is obtained using the following formula when the base code size and the percentage of change to the base code is known:

$$\text{Size} = (\text{Size Added} + \text{Size Modified}) (\text{MAF})$$

Where: MAF = Maintenance Adjustment Factor = $1 + (\text{SU}/100 \times \text{UNFM})$

SU = Software Understanding (see reference [3] for rating scale)

UNFM = Software Familiarity (see reference [3] for rating scale)

As the formula indicates, the model takes into account the effort required to understand and become familiar with the system being estimated. In addition, a companion model called COPLIMO can be used to extend the COCOMO II model to take into account the life cycle effects of software product line investments.

WBS Element	COCOMO II	SEER-SEM	SLIM	True S
1.1 Maintenance	Code fragment model	Code fragment model or dedicated maintenance model	Code fragment model	Dedicated model for software maintenance driven by defects
1.2 Sustaining Engineering	Not addressed for software maintenance	Can be addressed for software maintenance*	Can be addressed for software maintenance ⁺	Addressed by maintenance model for software as LOE task
1.3 Independent Test & Verification	Not addressed for software maintenance	Not addressed for software maintenance	Can be addressed for software maintenance ⁺	Can be addressed for software maintenance [@]
1.4 Product Support	Partially addressed for software maintenance	Partially addressed for software maintenance*	Can be addressed for software maintenance ⁺	Partially addressed for software maintenance
1.5 Information Assurance	Not generally addressed as on-going requirement	Can be addressed via security parameter settings	Not generally addressed as on-going requirement	Can be addressed via security parameter settings
1.6 Acquisition Support	Not addressed for software maintenance	Not addressed for software maintenance	Not addressed for software maintenance	Not addressed for software maintenance
1.7 Operations Support	Not addressed for software maintenance	Can be addressed for software maintenance*	Can be addressed for software maintenance ⁺	Partially addressed by dedicated maintenance model for software
1.8 Facility Support	Not addressed for software maintenance	Can be addressed for software maintenance*	Not addressed for software maintenance	Can be addressed for software maintenance [^]
1.9 Field Support	Not addressed for software maintenance	Can be addressed for software maintenance*	Not addressed for software maintenance	Partially addressed by dedicated maintenance model for software
1.10 Management	Partially addressed as percentage effort	Partially addressed as percentage effort	Partially addressed as percentage effort	Can be addressed for software maintenance [@]
1.11 Parts	Not addressed for software maintenance	Not addressed for software maintenance	Not addressed for software maintenance	Not addressed for software maintenance
1.12 Spares	Not addressed for software maintenance	Not addressed for software maintenance	Not addressed for software maintenance	Not addressed for software maintenance
1.13 Licenses	Not addressed for software maintenance	Can be addressed for software maintenance*	Not addressed for software maintenance	Partially addressed by dedicated maintenance model for software
1.14 Cost Item General	Catch-all	Catch-all	Catch-all	Catch-all
Notes:		* You can estimate these tasks using a 2 nd model called SEER-IT. But, SEER-SEM does not perform this task.	+ You can allocate effort to these tasks via a separate package called SLIM MasterPlan.	@ You can estimate these tasks using a 2 nd model for system/product management. ^ You can estimate these tasks using a facilities model.

Table 2 – Cost Model Mappings to Software Maintenance WBS^{1,2}

Notes

- 1 We have based our results on publically available information contained in manual and training materials.
- 2 We have tried to be accurate and fair with our ratings by having the cost model developers review the entries to this Table.

The COCOMO II model assumes that code can either be added or modified, but not deleted in the maintenance version. Effort in person months for maintenance is determined on an annual basis using the same set of formulas used to predict development costs with updated size and cost driver ratings. The resulting maintenance effort can be allocated to other activities like those in Table 1 by passing estimate results to a spreadsheet.

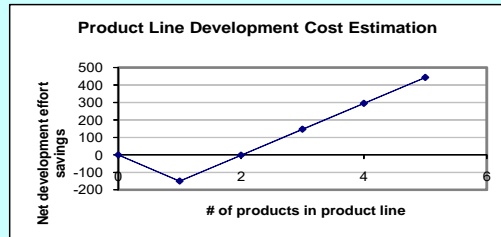
- **SEER-SEM** - This model estimates software maintenance costs using a set of maintenance specific parameters. In addition to entering the size to be maintained, you need to enter the number of years you want your maintenance estimate to cover. There are several other maintenance specific parameters which may also be set to fine tune the estimate. These include total size growth over life, maintenance rigor, maintenance personnel experience, maintenance tools and practices, and annual change. You may also set maintenance specific labor rates and include other maintenance costs. Maintenance estimates may be computed based on total size, effective size or any specific fraction of size. The maintenance size and parameters are then used to compute a required maintenance staff. Maintenance efforts are allocated into four types of change: corrective, adaptive, perfective and enhancements. The SEER-SEM model covers the maintenance of software systems. The SEER-IT version estimates the cost of supporting deployed systems. SEER-IT estimates activities such as help desk, network administration, database and application administration, backup/recovery operations, end user support, troubleshooting, training, documentation and content updates. This type of support falls under the general heading of ongoing support in the SEER-IT package which is a separate software package that must be acquired in order to estimate the scope of the activities listed in Table 1.
- **SLIM** – The SLIM model predicts effort and duration for a project using a unique set of math formulas which call for rating just a few drivers. You select the life cycle, put in size and a productivity index and away you go. You can fine tune the estimate by specifying things like industry sector, application type, language, reliability and other environmental factors. The package then generates estimates for maintenance via the SLIM MasterPlan Tool. MasterPlan is a project aggregation tool that must be acquired in order to estimate the scope of the activities listed in Table 1. Typically the program being estimated may have several major development releases followed by one or multiple years of “software maintenance.” The SLIM model views software maintenance as a combination of (1) major enhancements, (2) minor enhancements, and (3) baseline support (emergency fixes, help desk, infrastructure upgrades, operational support, small research projects, and other activities). The SLIM model is flexible in that it allows you to add or delete categories depending on the specific customer environment and its unique cost and scheduling requirements. The SLIM MasterPlan Tool allows model users to bolt a one to five year maintenance solution to the development estimate for life cycle costing. SLIM MasterPlan provides users flexibility in that it can be customized to many customer situations.
- **True S** – Estimating with the True S model has been likened to peeling an onion. You refine your estimates as you add more and more detail to them. Like COCOMO and SEER, you can cut into the onion by calibrating factors. The more factors, the more refined your estimate becomes. Estimating maintenance is done differently in True S because it assumes that software maintenance is determined by the latent defects in a release rather than as a function of how much of the code is modified. In addition, True S assumes that maintenance can involve more than just a release estimate. For example, it can assume that there can be sustaining engineering activity that is estimated solely using a fixed head count. Like with other models, the maintenance costs associated with modeling of new features and their subsequent integration with existing capabilities can be accomplished through specifying the amount of new, adapted, reused and deleted code. Finally, the model can be used to compute the costs for deployment activities using the number of client installations as its base. Low volume distributions are assumed to be highly customized with their deployment costs a function of size and applications complexity.

Maintenance screen shots for each of these four popular software cost models are provided as Figure 1 (COPLIMO extension of COCOMO II), Figures 2 and 3 (SEER-SEM), Figure 4 and 5 (SLIM), and Figure 6 and 7 (True S). These screen shots were included to illustrate just some of the power that each of these software maintenance estimating packages currently provide to its users through their base model and companion packages. We suggest those interested in gathering more facts check with the vendors or go to their web sites because all four models are continually being updated to incorporate new features.

COPLIMO Estimation Summary

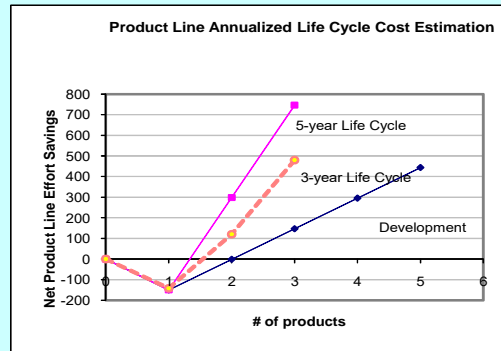
Part I: Product Line Development Cost Estimation Summary:

# of Products Effort (PM)	0	1	2	3	4	5
No Reuse	0	294	588	882	1176	1470
Product Line	0	444	589	735	881	1026
Product Line Savings	0	-150	-1	147	295	444
ROI	0	-1.00	-0.01	0.98	1.97	2.96



Part II: Product Line Annualized Life Cycle Cost Estimation Summary:

# of Products	0	1	2	3	4	5
AMSIZE-P	0	8.1	16.2	24.2	32.3	40.4
AMSIZE-R	0	6.1	6.1	6.1	6.1	6.1
AMSIZE-A	0	6.1	7.7	9.3	11.0	12.6
Total Equiv. KSLLOC Effort (AM) (*2.94)	0	20.2	29.9	39.6	49.3	59.1
5-year Life Cycle PM	0	296.9	439.8	582.6	725.4	868.3
PM(N, 5)-R (+444)	0	740.9	883.7	1026.5	1169.4	1312.2
PM(N, 5)-NR	0	590.9	1181.9	1772.8	2363.8	2954.7
Product Line Savings (PM)	0	-149.9	298.2	746.3	1194.4	1642.5
ROI	0	-1.00	1.99	4.98	7.97	10.96
Devel. ROI	0	-1.00	-0.01	0.98	1.97	2.96
3-year Life Cycle	0	-142.0	120.0	480.0		



AMSIZE: Annually Maintained Software Size

Figure 1 – COPLIMO Maintenance Screen

In Figure 1, the COPLIMO extension to the COCOMO II model provides users with a life cycle support profile for an example product line investment. Because COCOMO II estimates software development costs only, such extensions have proved valuable when trying to bound operations and support costs. Maintenance costs are estimated separately on an annual basis over a variable time period and added to the development estimate to get total life cycle cost.

As Figures 3 and 4 illustrate, the SEER-SEM and SEER-IT estimating models provide the user with a powerful array of reports about maintenance starting with an annual projection and continuing with program rollup and project estimates. These packages also provide insight to on-going support costs in the SEER-IT view, which must be acquired separately.

The SLIM model estimates software maintenance costs on an annual basis over a variable time period. These costs are then allocated across the project schedule using the separately acquired SLIM MasterPlan Tool to provide detail like in the screens illustrated in Figures 4 and 5, SLIM resource summary and maintenance screens, respectively. The two estimates can then be summed to develop a total cost projection across the total life cycle.

The impact of major life cycle cost drivers on software are shown for the True S model in the next set of Figures. Figure 6 predicts how increases in new code size leads to increases in deployment, maintenance and adaptation of the software. Figure 7 illustrates how higher values of operating specification (the platform on which the software operates) lead to decreases in defects via implied parallel increases in testing rigor.

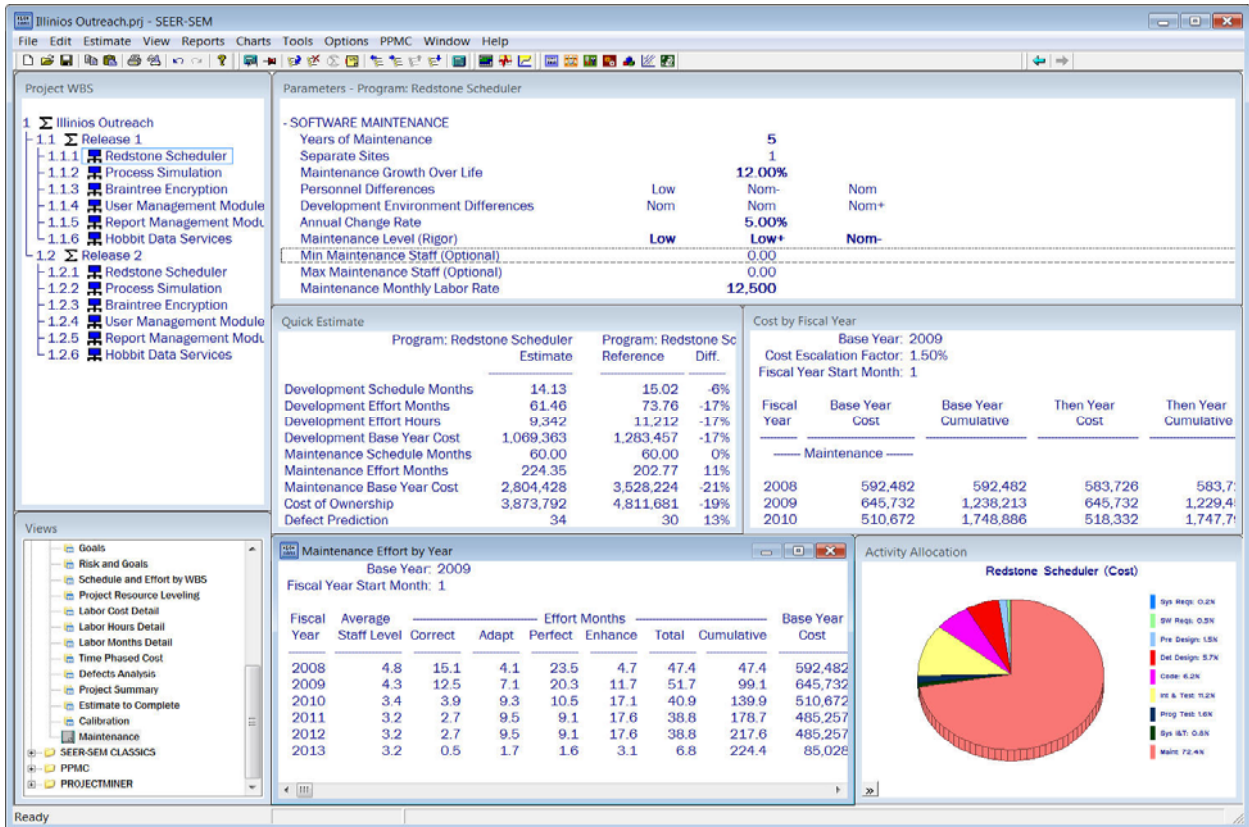


Figure 3 – SEER-SEM Maintenance Effort by Year Report

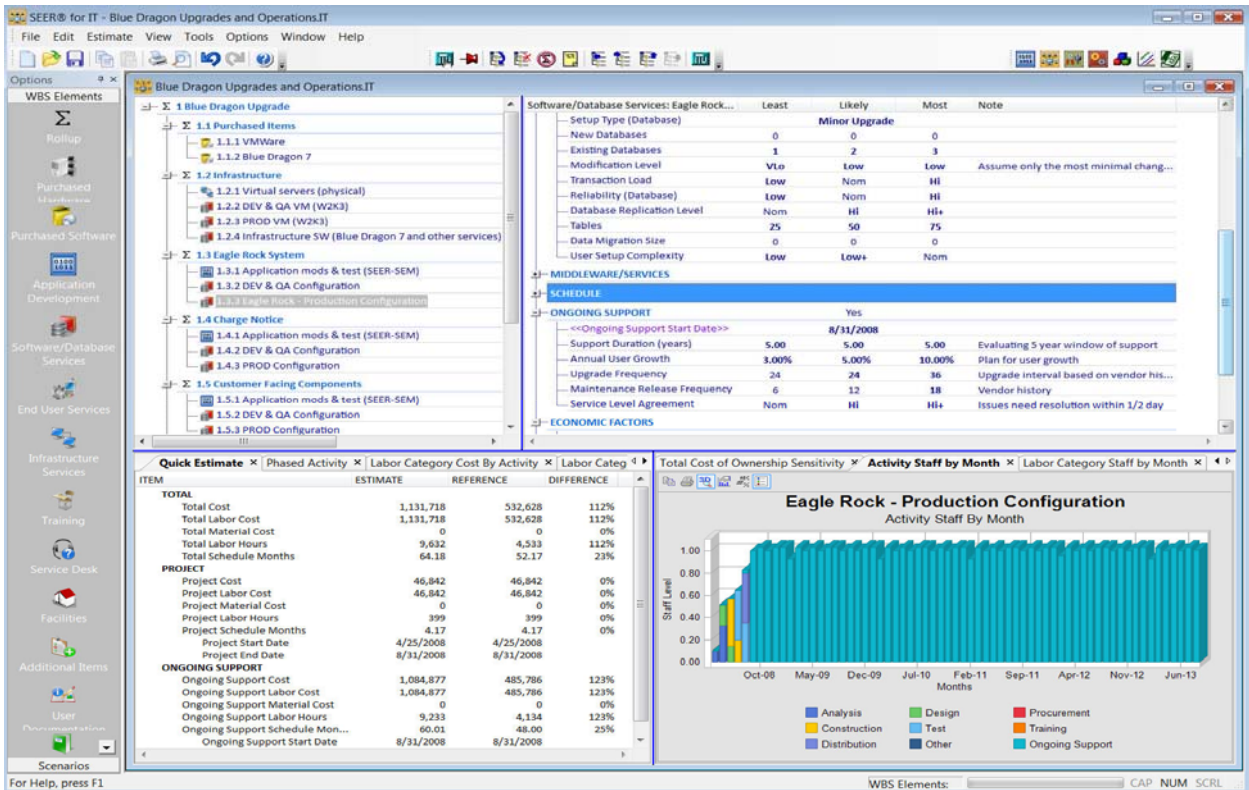


Figure 4 – SEER-IT On-Going Support Example

Resource-Cost-Schedule View

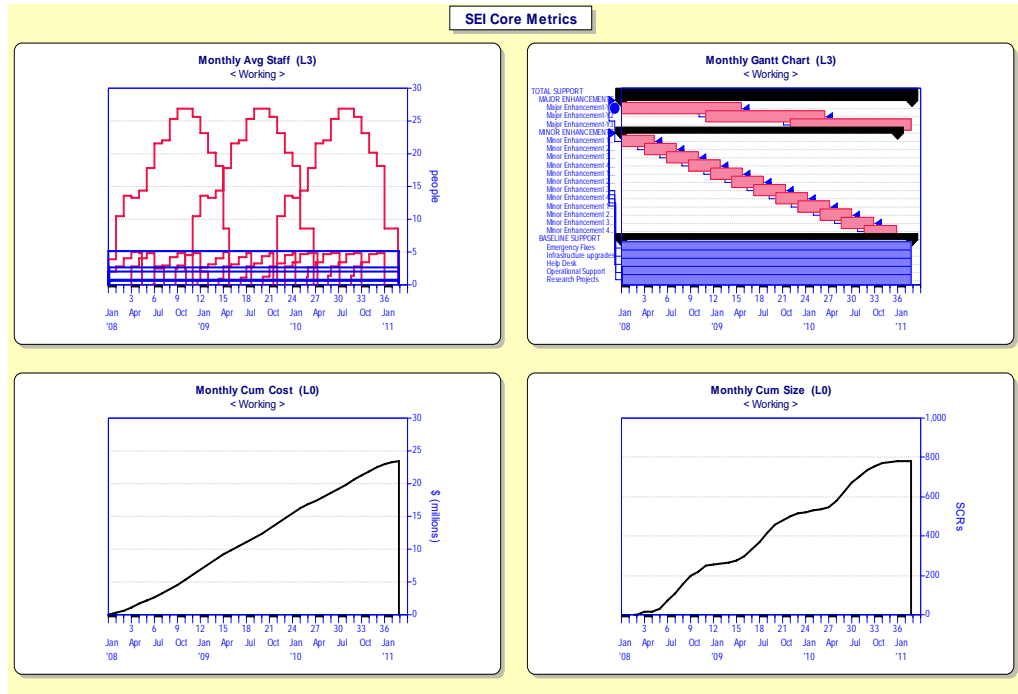


Figure 4 – SLIM Estimate Summary Screens

3 Year Maintenance Summary Report

Program Summary Report
Maintenance Model 2

Task	Task Description	Start Date	End Date	Elapsed Months	MM	(\$1000)
TOTAL SUPPORT	Summary Task	1/1/2008	2/23/2011	37.82	1,445.23	23,413
MAJOR ENHANCEMENTS	Summary Task	1/1/2008	2/23/2011	37.82	856.14	13,870
Major Enhancement-Y1	SLIM-Estimate Subsystem (Majo...	1/1/2008	4/23/2009	15.77	285.38	4,623
Major Enhancement-Y2	SLIM-Estimate Subsystem (Majo...	11/30/2008	3/21/2010	15.71	285.38	4,623
Major Enhancement-Y3	SLIM-Estimate Subsystem (Majo...	10/31/2009	2/23/2011	15.85	285.38	4,623
MINOR ENHANCEMENTS	Summary Task	1/1/2008	12/31/2010	36.00	157.09	2,545
Minor Enhancement 1-Y1	SLIM-Estimate Subsystem (Mino...	1/1/2008	5/13/2008	4.42	13.61	221
Minor Enhancement 2-Y1	SLIM-Estimate Subsystem (Mino...	3/31/2008	8/9/2008	4.52	13.05	211
Minor Enhancement 3-Y1	SLIM-Estimate Subsystem (Mino...	6/27/2008	11/4/2008	4.27	13.09	212
Minor Enhancement 4-Y1	SLIM-Estimate Subsystem (Mino...	9/23/2008	1/31/2009	4.27	13.11	212
Minor Enhancement 1-Y2	SLIM-Estimate Subsystem (Mino...	12/19/2008	4/25/2009	4.25	13.03	211
Minor Enhancement 2-Y2	SLIM-Estimate Subsystem (Mino...	3/13/2009	7/21/2009	4.29	13.04	211
Minor Enhancement 3-Y2	SLIM-Estimate Subsystem (Mino...	6/9/2009	10/16/2009	4.25	13.04	211
Minor Enhancement 4-Y2	SLIM-Estimate Subsystem (Mino...	9/4/2009	1/11/2010	4.25	13.04	211
Minor Enhancement 1-Y3	SLIM-Estimate Subsystem (Mino...	11/30/2009	4/7/2010	4.27	13.01	211
Minor Enhancement 2-Y3	SLIM-Estimate Subsystem (Mino...	2/24/2010	7/2/2010	4.24	13.04	211
Minor Enhancement 3-Y3	SLIM-Estimate Subsystem (Mino...	5/22/2010	10/1/2010	4.35	13.03	211
Minor Enhancement 4-Y3	SLIM-Estimate Subsystem (Mino...	8/21/2010	12/31/2010	4.35	13.02	211
BASELINE SUPPORT	Summary Task	1/1/2008	2/23/2011	37.82	432.00	6,998
Emergency Fixes	Custom Task	1/1/2008	2/23/2011	37.82	31.42	509
Infrastructure upgrades	Custom Task	1/1/2008	2/23/2011	37.82	78.54	1,272
Help Desk	Custom Task	1/1/2008	2/23/2011	37.82	196.36	3,181
Operational Support	Custom Task	1/3/2008	2/25/2011	37.83	102.11	1,654
Research Projects	Custom Task	1/3/2008	2/25/2011	37.83	23.56	382
Overall Program	Custom Task	1/1/2008	2/23/2011	37.82	1,445.23	23,413

Figure 5 – SLIM Maintenance Screen

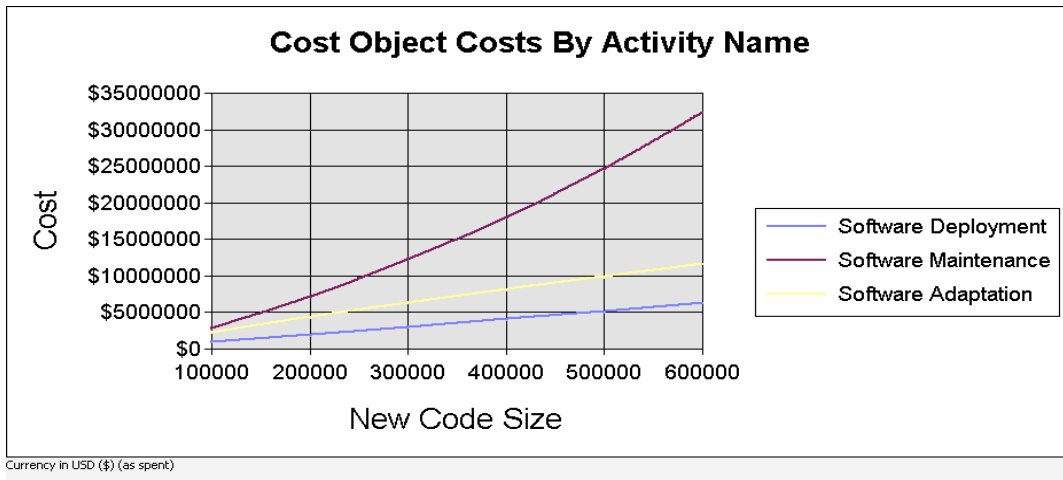


Figure 6 – Impact of New Code Size on Deployment, Maintenance and Adaptation Predicted by True S

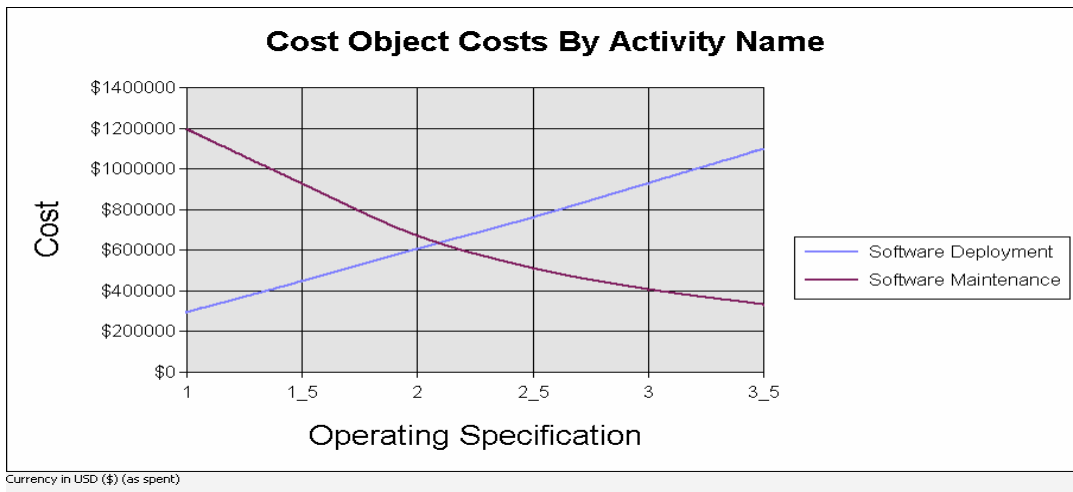


Figure 7 – Impact of Operating Specification on Deployment Costs Predicted by True S

4. Suggested Maintenance Estimating Guidelines

To address the estimating shortfalls illustrated in Table 2 (i.e., Cost Model Mappings to Software Maintenance WBS), we have developed a set of model-independent estimating guidelines for use in estimating software maintenance and support costs. These guidelines, which are summarized in Table 3, were developed for the four types of projects we found that government software life cycle support centers typically performed, which include the following:

- Sustaining Support** – As much as eighty percent of the work in government software life cycle support centers is typically performed on a Level-Of-Effort (LOE) basis by small teams of ten or less. These teams are tasked with doing everything necessary to keep the system operational in the field. They prioritize their updates based on criticality of the repair and perform whatever tasks are needed including hardware maintenance and field support to keep the system working. They may incorporate enhancements, repair problems and may take needed action to address performance issues. If necessary, they may even travel to patch and repair systems in the field. However, in most cases, their primary focus is addressing software trouble reports, many of which are taken from their current backlog.

WBS Element	Sustaining Support	Acquisition Support	Medium-Scale Enhancement	Large-Scale Enhancement
1.1 Maintenance	Performed LOE – estimate at 50% of the staff assigned	Performed by others – estimate at 4 to 8% of contractor staff assigned	Estimate using one of the software cost models	Estimate using one of the software cost models
1.2 Sustaining Engineering	Performed LOE – estimate at 50% of the staff assigned	Optional task - when performing IT&V, add 10 to 30% of WBS # 1.3 effort to the test team, assume no independent test tools are maintained	Secondary task – estimate at 20 to 50% of maintenance staff size depending on tasks involved	Secondary task – estimate using past experience based on tasks involved and their difficulty
1.3 Independent Test & Verification (IT&V)	Part of the staff assignment ¹	Performed LOE – estimate at 10 to 20% of contractor staff assigned	Part of staff assignment ¹	Part of staff assignment ¹
1.4 Product Support	Part of the staff assignment ¹	Not applicable (handled by others)	Part of staff assignment ¹	Part of staff assignment ¹
1.5 Information Assurance	Requires further study	Requires further study	Requires further study	Requires further study
1.6 Acquisition Support	Not applicable	Not applicable	Optional task – estimate 8 to 10% of staff being managed	Optional task – estimate 8 to 10% of staff being managed
1.7 Operations Support	Performed LOE – reassign people as needed to do work	Not applicable (handled by others)	Part of staff assignment, if required	Part of staff assignment, if required
1.8 Facility Support	Part of the staff assignment ¹	Applicable only if IT&V performed – addressed in sustaining engineering WBS item 1.2.	Part of staff assignment ¹	Part of staff assignment ¹
1.9 Field Support	Performed LOE – reassign people as needed to do work	Not applicable (handled by others)	Part of staff assignment ¹	Part of staff assignment ¹
1.10 Management	Performed LOE – assume one working manager/project	Important task – assume a 8 to 12% management load	Important task – assume a 8 to 12% management load	Important task – assume a 8 to 12% management load
1.11 Parts	Performed LOE – reassign people as needed to do work	Not applicable (handled by others)	Part of staff assignment ¹	Part of staff assignment ¹
1.12 Spares	Performed LOE – reassign people as needed to do work	Not applicable (handled by others)	Part of staff assignment ¹	Part of staff assignment ¹
1.13 Licenses	Not applicable (handled by others)	Not applicable (handled by others)	Optional market watch task, maintain relationship, conduct annual census for licenses ²	Optional market watch task, maintain relationship, conduct annual census for licenses ²
1.14 Cost Item General (catch-all)	Performed LOE – reassign people as needed to do work not thought of	Performed LOE – reassign people as needed to do work not thought of	Performed LOE – reassign people as needed to do work not thought of	Performed LOE – reassign people as needed to do work not thought of

Table 3 – Software Maintenance Cost Estimating Guidelines

Notes

- 1 This task is a normal part of the work performed
- 2 Licensing costs can be substantial especially if COTS products are used

- **Acquisition Support** – We found that as much as thirty percent of the work done in government software life cycle support centers was aimed at providing acquisition management support to Program Offices managing contractors that were tasked to do the maintenance. Staff assigned to these teams provided oversight and direction to the contractors. Sometimes they performed special studies and other times they performed varying degrees of independent testing. This testing was in all cases aimed at providing the Program Office with additional assurance that the system being maintained would satisfy its requirements and perform as expected in its operational environment. Such testing ranged from independent testing to Independent Verification and Validation (IV&V). IV&V differs from independent testing in that it provides a more comprehensive full maintenance cycle review of products as they are engineered (i.e., verifies the design meets requirements, ensures the code satisfies the design, verifies the code is defect free and validates the release against requirements established for it). In addition, there may be specialized testing conducted for the purposes of security or nuclear certification as well.
- **Medium-Scale Enhancements** – Enhancements are traditional maintenance projects where a new release is developed by a government software life cycle support center team based on requirements. The in-house software life cycle support team proceeds through a traditional development cycle to update the software primarily to incorporate changes that address these requirements, repair known problems and make needed performance improvements (perfective changes). However, these teams do a great deal more work than meets the eye. They also sustain and maintain the facilities necessary to develop and test the new releases. These facilities often incorporate more equipment than those devoted to development because operational hardware can and is often placed in the loop, whenever possible. Medium-sized enhancements are those updates that can be accomplished in about a year by teams ranging in size from ten to fifty people. The trick in estimating for such medium-sized enhancements is to make sure that the budgets available are adequate to perform the sustaining engineering and support tasks. Because budgets for these projects are frequently provided as separate line items, they are often insufficient to do the job.
- **Large-Scale Enhancements** – Large-scaled enhancements differ from what we described for medium-sized projects in in-house life cycle support team size and duration. Because these multi-year projects employ over fifty people, they have the economies of scale to reallocate staff to perform the full-range of tasks needed to maintain and sustain the software (and the hardware). They have greater autonomy than the other types of projects. This allows them a great deal more freedom of action. The trick in estimating for such large-sized enhancements is to retain the flexibility to reallocate staff from one task to another, when the need arises. For example, they might need to take staff off of working a new release and reassign them to sustaining engineering tasks when major new operating system releases were being incorporated to their platform. It should be noted that this task would be handled differently because platform maintenance would be performed by a separately funded group.

The guidelines in this Table were developed based on data that we collected during interviews with government software life cycle support center project leads and their staffs. We are in the process of validating and expanding these guidelines. During the next six months, we plan to interview subject matter experts and ask them for their opinions and data relative to how best to estimate these tasks. One such polling of the experts we hope will occur at a workshop on software maintenance metrics and models that we will hope to hold at the 24th International Forum on COCOMO and System/Software Cost Modeling at the Massachusetts Institute of

Technology (MIT) in November 2009. Of course, we also plan to separately engage the cost model vendors and ask to confirm our findings as well. Finally, we plan to mine the software maintenance project cost database that we have accumulated to-date to determine whether the data supports our conclusions. These data have been gathered throughout the last year as we interviewed projects at government software life cycle support centers about what tasks they performed as part of the topic of software maintenance. While noisy, they should provide us some insight into whether or not our guidelines are properly directed.

5. Terms of Reference

Terms of reference for maintenance terminology used in this article follow:

- **Acquisition Management (Support)** - the process of supporting the program office by acting as their trusted agent to provide independent analysis, contractor oversight and management support on an as-needed and directed basis
- **Independent Verification and Validation (IV&V)** – the process of determining if a software deliverable meets its specifications and fulfills its intended purpose. Verification involves evaluating the product as it is developed step-wise and in-depth, while validation is conducted to close the loop towards the end of the development cycle and ensure that the product meets its requirements.
- **Independent Verification and Test (IV&T)** – the process of determining if a software deliverable meets its specification and can operate effectively in its intended environment. The IV&T process differs from IV&V in that the software is typically analyzed in a System Integration Lab using operational environment using actual equipment, signals and threats during the testing of the software.
- **Information Assurance** –the process of protecting and defending information systems by ensuring their availability, integrity, authentication, confidentiality and non-repudiation.
- **Interoperability Engineering** - the process of engineering two or more systems so that they can exchange information and use it effectively to accomplish their desired missions.
- **Maintenance (Software)** – the process of modifying a software system or component after delivery to correct faults, improve performance or other attributes, or adapt to a modified environment (ISO/IEC 14764). Maintenance typically generates new versions of the software that are periodically released to the field.
- **Post Deployment System Support** – the process of modifying software or firmware resident in a weapon system to provide for repairs, corrective coding, upgrades, growth, enhancements and modernization. Associated modifications to the development environment and test facilities are an inherent part of the job.
- **Sustaining Engineering** – the process of sustaining the infrastructure and environment needed to maintain the software. The infrastructure revolves around configuration and distribution management practices along with those used for emergency repairs and customer support. The environment includes those facilities and tools used for releases of new and patch versions of the software to the field.

6. Summary and Conclusions

We began this paper by summarizing our findings about the work software life cycle support centers perform under the heading of software maintenance. We next looked at the popular software cost estimating models to determine which of these tasks were within their scope. Finally, we provided recommended software maintenance cost estimating guidelines by task in an attempt to assist those in the community developing estimates for their systems.

We would also like to extend our appreciation to the following leaders in the software cost estimating field who reviewed our results for accuracy: Barry Boehm of USC, Karen McRitchie of Galorath, Doug Putnam of QSM, and David Seaver of Price Systems.

It is important to note that the findings, conclusions and recommendations within this paper are directed towards weapons systems. Based on our extensive analysis of recently released ISBSG (International Software Benchmarking Standards Group) data on 350+ maintenance and support projects [8], these findings are not applicable to mainframe-based data processing projects. The differences in the underlying nature of the projects reported are the root cause of the disparity.

The study team is indebted to those who participated during its investigations. Their openness, support and candid appraisals of current software maintenance practices are acknowledged. So is the support of senior management and the staff located at the U.S. Army and Air Force's software life cycle support centers and headquarters organizations.

References

- [1] D. Reifer, J. Allen, B. Fersch and B. Hitchings, "Software Maintenance Myths and Malpractice," draft, September 2009.
- [2] International Standards Organization (ISO), ISO/IEC 14767:2006, *Software Engineering – Life Cycle Processes – Maintenance*, 2006.
- [3] B. Boehm, C. Abts, A. Brown, S. Chulani, B. Clark, E. Horowitz, R. Madachy, D. Reifer and B. Steece, *Software Cost Estimation with COCOMO II*, Prentice-Hall, 2000.
- [4] D. Galorath and M. Evans, *Software Sizing, Estimation, and Risk Management*, Auerbach Publications, 2006.
- [5] QSM, *SLIM Estimate for Windows 6.1 User's Guide*, 2003.
- [6] R. Madachy and B. Boehm, *Comparative Analysis of COCOMO-II, SEER-SEM and True S Software Cost Models*, Report USC-CSSE-2008-816, University of Southern California, 2008.
- [7] R. Stutzke, *Estimating Software-Intensive Systems*, Addison-Wesley, 2005.
- [8] International Software Benchmarking Standards Group (ISBSG), *Special Report: Managing Your Maintenance and Support Environment*, available for a fee from www.isbsg.org, 2009.