SEER-SEM to COCOMO II Factor Convertor

Anthony L. Peterson, Derrick Tate, David A. Wyrick, Texas Tech University
Ricardo Valerdi, Massachusetts Institute of Technology

Abstract

Previously Rosetta Stones have been developed for converting COnstructive COst MOdel COCOMO II estimate inputs into corresponding SEER-SEM or True S inputs, or vice-versa. Most of those Rosetta Stone mappings between factors are one-to-one, but some are one-to-many. These factors often have disproportionate definitions. Additionally, each model contains factors that are unique to that model. [Madachy-Boehm 2006]

This paper describes a tool that provides automation of factor transformations based on an refinement of one of those Rosetta Stones, specifically, the mapping from SEER-SEM to COCOMO II. Techniques are described to map between disproportionate factor definitions. Subsequent to specifying the three Effort Multipliers not mapped from SEER-SEM (Personnel Continuity, Data Base Size, and Documentation) a composite Effort Multiplier is obtained.

Following specifying the five COCOMO II Scale Factors and the size of the software development effort in ESLOC (equivalent source lines of code) the tool produces a COCOMO II estimate. If schedule compression is detected, the final EM, SCED (Required Schedule), is computed using a new technique of interpolating between an uncompressed schedule (SCED = 1.00) and 75% schedule compression (SCED = 1.43) using a 4^{th} degree polynomial. The SCED EM is then applied to arrive at the final adjusted COCOMO II estimate.

Introduction

A key component of a successful engineering bid is realistic labor cost estimates. Software engineering has several mature parametric engineering labor cost estimation models such as COCOMO II, SEER-SEM, and True S to support their software cost and estimation processes. These models have also helped software engineering to reason about the cost and schedule implications of their development decisions, investment decisions, client negotiations and requested changes, risk management decisions, and process improvement decisions. [Boehm et al 2005] The result has been that software engineering labor estimates are automated via parametric models calibrated with industry model norms or better still local program experience. The resultant estimates have greater cost realism with superior basis of estimate than estimates generated with rules of thumb heuristics or by time consuming bottoms-up process.

Sometimes an organization needs to estimate a software effort using more than one parametric model. Automating the development of a second estimate would provide not only a cost savings for multiple rounds of costing but also a reduction in errors introduced. This paper discusses one such automation –

that of translating an estimate using SEER-SEM into a COCOMO II Post-Architecture model estimate. Building on the work of Madachy and Boehm [Madachy-Boehm 2006], Table 1 depicts a top-level Rosetta Stone which maps to COCOMO II Effort Multiplier cost drivers to SEER-SEM cost factors. While most of the mappings are one to one, two different COCOMO II effort multipliers are mapped to multiple SEER-SEM factors. SEER-SEM has factors for platform volatility ratings split into host and target as well as factors for platform experience split into development and target. The final Effort Multiplier, SCED, is a factor based on the required development schedule and the COCOMO II estimate of nominal development schedule duration. A new technique is offered to compute SCED based on model inputs.

Table 1: COCOMO II Effort Multiplier to SEER-SEM Cost Factor

COCOMO II Effort Multiplier	ID	SEER−SEM Cost Factor
PRODUCT ATTRIBUTES		
Required Software Reliability	RELY	Specification Level - Reliability
Data Base Size	DATA	No Equivalent
Product Complexity	CPLX	Complexity (Staffing)
Required Reusability	RUSE	Reusability Level Required
Documentation	DOCU	No Equivalent
PLATFORM ATTRIBUTES		
Execution Time Constraint	TIME	Time Constraints
Main Storage Constraint	STOR	Memory Constraints
Platform Volatility	PVOL	Host System Volatility
		Target System Volatility
PERSONNEL ATTRIBUTES		
Analyst Team Capabilities	ACAP	Analyst Capabilities
Programmer Team Capabilities	PCAP	Programmer Capabilities
Personnel Continuity	PCON	No Equivalent
Applications Experience	APEX	Analyst Application Experience
Platform Experience	PLEX	Development System Experience
		Target System Experience
Language and Tools Experience	LTEX	Programmer’s Language Experience
PROJECT ATTRIBUTES		
Use of Software Tools	TOOL	Automated Tools Use
Multi-Site Development	SITE	Multiple Site Development

COCOMO II Effort Multiplier	ID	SEER−SEM Cost Factor
Required Schedule	SCED	Development Schedule

COCOMO II

The COCOMO (COnstructive COst MOdel) cost and schedule estimation model was originally published in [Boehm 1981]. The COCOMO II research effort was started in 1994, and the model continues to be updated at USC, the home institution of research for the COCOMO model family. COCOMO II has three submodels: Applications Composition, Early Design and Post-Architecture. [Boehm et al. 2000] The focus of this paper is the Post-Architecture model.

The Post-Architecture model is used when top level design is complete and detailed information about the project is available and the software architecture is well defined. It uses Source Lines of Code and/or Function Points for the sizing parameter, adjusted for reuse and breakage; a set of seventeen effort multipliers and a set of five scale factors that determine the economies/diseconomies of scale of the software under development. [Madachy, Boehm 2006]

SEER-SEM

SEER-SEM is a product offered by Galorath, Inc. This model is based on the original Jensen model [Jensen 1983], and has been available for over 20 years. Its parametric modeling equations are proprietary. Descriptive material about the model can be found in [Galorath-Evans 2006].

The scope of the model covers all phases of the project life-cycle, from early specification through design, development, delivery and maintenance. It supports a variety of environmental and application configurations, and models different development methods and languages. Development modes covered include object oriented, reuse, COTS, spiral, waterfall, prototype and incremental development. Languages covered are 3rd and 4th generation languages (C++, FORTRAN, COBOL, Ada, etc.), as well as application generators. [Madachy, Boehm 2006]

The SEER-SEM cost model allows probability levels of estimates, constraints on staffing, effort or schedule, and it builds estimates upon a knowledge base of existing projects. Estimate outputs include effort, cost, schedule, staffing, and defects. Sensitivity analysis is also provided. [Madachy, Boehm 2006] See the Galorath Inc. website at http://www.galorath.com.

SEER-SEM to COCOMO II Detailed Rosetta Stone

Table 2 shows the detailed correspondence between COCOMO II and SEER-SEM factors with guidelines to convert ratings between the two models for applicable factors. In some cases the SEER-SEM factors cover different ranges than COCOMO II EM factors and some of the conversions in Table 2 are best approximations. Not all factors have direct corollaries. The settings of the SEER-SEM factors may be defaulted according to project type and domain choices in the knowledge bases. [Madachy, Boehm 2006]

Table 2 depicts the detailed Rosetta Stone mapping from SEER-SEM cost factors to COCOMO II EM factors. The defined COCOMO II EM factors [Boehm et al. 2000] are in bold fort. There are some cases where an entire mapping is not possible. Some of the transformations in Table 2 are only approximations. For example, in some instances COCOMO II EM factors cover different ranges than SEER-SEM cost factors. SEER-SEM cost factors also have a greater fidelity. They can be set "between" the defined values for Extra High, Very High, High, Nominal, Low, and Very Low. In order to provide a mapping of those "between" settings, a linear interpolation of COCOMO II Effort Multipliers is employed. This is a refinement of the method presented in an unpublished Galorath draft. [Galorath, 2003]

To illustrate consider the SEER-SEM "Analyst's Application Experience" factor with a range from <4 months to 10 years mapping to the COCOMO II "Applications Experience" EM with a range between 2 months and 6 years. This results in mapping a SEER-SEM experience rating of 6 or more years to the COCOMO II experience EM rating of 6 years. Linear interpolation is used to map the gaps between the "Very High" and "High" EM factors and between "High" and "Normal" EM factors. For example a SEER-SEM "Low" rating and a COCOMO II EM "Nominal" rating for experience corresponds to 1 year awhile a SEER-SEM "Nominal" rating and a COCOMO II EM "High" rating corresponds to 3 years. Since SEER-SEM also includes the "between" ratings of "Low+" and "Nominal–" and the EM rating for "High" experience is 0.88 the corresponding "between" EM rating for experience of "Nominal+" is 0.96 and of "High–"is 0.92.

Not all mappings are as straightforward. Again considering the experience cost factor, a SEER-SEM "Very Low" rating corresponds to <4 months while a COCOMO II EM "Low" rating corresponds to 6 months and a COCOMO II EM "Very Low" rating corresponds to 2 months. Thus the closest mapping of SEER-SEM "Very Low" rating is the COCOMO II EM "Very Low" rating of 1.22 and the closest mapping of SEER-SEM "Very Low+" rating is the COCOMO II EM "Low" rating of 1.10. The remaining SEER-SEM "Low–" rating is then mapped between COCOMO II EM "Nominal" and COCOMO II EM "Low" for a rating of 1.05.

Table 2: Detailed SEER-SEM Factor to COCOMO II Effort Multiplier Mapping

SEER−SEM Cost Factor Definitions	SEER−SEM Rating	COCOMO II Effort Multiplier (EM) Definitions	COCOMO II EM Rating	
PERSONNEL CAPABILITIES & EXPERIENCE				
Analyst Capabilities		**Analyst Team Capabilities (ACAP)**		
Near Perfect Functioning Team (90th percentile)	VHi	90th Percentile	**VH**	**0.71**
	VHi−		VH−	0.76
	Hi+		H+	0.80
Extraordinary (75th Percentile)	Hi	75th Percentile	**H**	**0.85**

SEER−SEM Cost Factor Definitions	SEER−SEM Rating	COCOMO II Effort Multiplier (EM) Definitions	COCOMO II EM Rating	
	Hi−		H−	0.90
	Nom+		N+	0.95
Functional and Effective (55th Percentile)	Nom	55th Percentile	**N**	**1.00**
	Nom−		N−	1.06
	Low+		L+	1.13
Functional with Low Effectivity (35th Percentile)	Low	35th Percentile	**L**	**1.19**
	Low−		L−	1.27
	VLo+		VL+	1.34
Poorly functioning team (15th Percentile)	VLo	15th Percentile	**VL**	**1.42**
Non-functional team (5th Percentile)	Vlo−		VL	1.42
Analyst's Application Experience		**Applications Experience (APEX)**		
> 10 years	VHi		VH	0.81
	VHi−		VH	0.81
	Hi+		VH	0.81
6 years	Hi	6 years	**VH**	**0.81**
	Hi−		VH−	0.84
	Nom+		H+	0.87
3 years	Nom	3 years	**H**	**0.90**
	Nom−		H−	0.93
	Low+		N+	0.97
1 year	Low	1 year	**N**	**1.00**
	Low−		L−N	1.05
	VLo+	6 months	**L**	**1.10**
< 4 months	VLo	2 months	**VL**	**1.22**
Programmer Capabilities		**Programmer Team Capabilities (PCAP)**		
Near Perfect Functioning Team (90th percentile)	VHi	90th Percentile	**VH**	**0.76**
	VHi−		VH+	0.80
	Hi+		H+	0.84
Extraordinary (75th Percentile)	Hi	75th Percentile	**H**	**0.88**
	Hi−		H−	0.92
	Nom+		N+	0.96
Functional and Effective (55th Percentile)	Nom	55th Percentile	**N**	**1.00**
	Nom−		N−	1.05
	Low+		L+	1.10
Functional with Low Effectivity (35th Percentile)	Low	35th Percentile	**L**	**1.15**
	Low−		L−	1.21
	VLo+		VL+	1.28
Poorly functioning team (15th Percentile)	VLo	15th Percentile	**VL**	**1.34**
Non-functional team (5th Percentile)	Vlo−		VL	1.34
Programmer’s Language Experience		**Language & Tools Experience (LTEX)**		
> 4 years	EHi	6 years	**VH**	**0.81**
	EHi−		VH−	0.84
	VHi+		H+	0.87
3 years	VHi	3 years	**H**	**0.90**
	VHi−		H	0.90

SEER−SEM Cost Factor Definitions	SEER−SEM Rating	COCOMO II Effort Multiplier (EM) Definitions	COCOMO II EM Rating	
	Hi+		H−	0.93
2 years	Hi		N−H	0.95
	Hi−		N+	0.97
	Nom+		N	1.00
1 year	Nom	1 year	**N**	**1.00**
	Nom−		L−N	1.06
	Low+	6 months	**L**	**1.12**
4 months	Low		L−	1.15
	Low−		VL+	1.17
	VLo+	2 months	**VL**	**1.20**
None	VLo		VL	1.20

Development (Host) System Experience		**Platform Experience (PLEX)**		
> 4 years	EHi	6 years	**VH**	**0.85**
	EHi−		VH−	0.88
	VHi+		H+	0.89
3 years	VHi	3 years	**H**	**0.91**
	VHi−		H	0.91
	Hi+		H−	0.94
2 years	Hi		N−H	0.96
	Hi−		N+	0.97
	Nom+		N	1.00
1 year	Nom	1 year	**N**	**1.00**
	Nom−		L−N	1.05
	Low+	6 months	**L**	**1.09**
4 months	Low		L−	1.12
	Low−		VL+	1.16
	VLo+	2 months	**VL**	**1.19**
None	VLo		VL	1.19
Target System Experience		**Platform Experience (PLEX)**		
> 4 years	EHi	6 years	**VH**	**0.85**
	EHi−		VH−	0.87
	VHi+		H+	0.89
3 years	VHi	3 years	**H**	**0.91**
	VHi−		H	0.91
	Hi+		H−	0.94
2 years	Hi		N−H	0.96
	Hi−		N+	0.97
	Nom+		N	1.00
1 year	Nom	1 year	**N**	**1.00**
	Nom−		L−N	1.05
	Low+	6 months	**L**	**1.09**
4 months	Low		L−	1.12
	Low−		VL+	1.16
	VLo+	2 months	**VL**	**1.19**
None	VLo		VL	1.19
Practices & Methods Experience		**No Equivalent**		
DEVELOPMENT SUPPORT ENVIRONMENT				
Modern Development Practices Use		**No Equivalent**		
Automated Tools Use		**Use of SW Tools (TOOL)**		
Advanced fully integrated toolset (Integrated CASE, full Ada APSE)	VHi	strong, mature, proactive life-cycle tools; well integrated with processes, methods, reuse	**VH**	**0.78**
	VHi−		VH−	0.82
Modern fully automated application development environment, including requirements, design, and test analyzers	Hi+		H+	0.86
Modern visual programming tools, automated CM, test analyzers plus requirements or design tools	Hi	strong, mature, proactive life-cycle tools; moderate integration	**H**	**0.90**

	Hi−		H−	0.93
Visual programming, CM tools and simple test tools	Nom+		N+	0.97
Interactive, programmer work bench (Ada minimal ASPE)	Nom	Basic life-cycle tools, moderate integration	**N**	**1.00**
Interactive (Programmer Work Bench)	Nom−		N−	1.03
	Low+		L+	1.06
Base batch tools (compiler, editor)	Low	Simple front end/back end tools, CASE. little integration	**L**	**1.09**
	Low−		L−	1.12
	VLo+		VL+	1.14
Primitive tools (bit switches, dumps)	VLo	edit, code, debug	**VL**	**1.17**
Logon through Hardcopy Turnaround		**No Equivalent**		
Terminal Response Time		**No Equivalent**		
Multiple Site Development		**Multiple Site Development (SITE)**		
Single site & single organization	Nom	Fully colocated	**XH**	**0.80**
	Nom+		XH−VH	0.83
	Hi−	Same building or complex	**VH**	**0.86**
Single site & multiple organization	Hi		VH−H	0.90
	Hi+	Same city, same company, wideband electronic communications	**H**	**0.93**
	VHi−		N−H	0.97
Multiple sites, same general location, or mixed clearance levels	VHi	Multi-site, multi-company; narrowband e-mail	**N**	**1.00**
	VHi+	Multi-site, multi-company; single phone, FAX	**L**	**1.09**
	EHi−		VL−L	1.16
Multiple sites, located 50 miles or more from each other, or international participation	EHi	International, some phone calls	**VL**	**1.22**
Resource Dedication		**No Equivalent**		
Development System Volatility		**Platform Volatility (PVOL)**		
Major Change Each 2 Weeks, Minor 2 Times A Week	EHi	Major Change Each 2 Weeks, Minor 2 Times A Week	**VH**	**1.30**
	EHi−		VH−	1.25
	VHi+		H+	1.20
Major Change Each 2 Months, Minor Each Week	VHi	Major Change Each 2 Months, Minor Each Week	**H**	**1.15**
	VHi−		H−	1.10
	Hi+		N+	1.05
Major Change Each 6 Months, Minor Each 2 Weeks	Hi	Major Change Each 6 Months, Minor Each 2 Weeks	**N**	**1.00**
	Hi−		N−	0.96
	Nom+		L+	0.91
Major Change Each 12 Months, Minor Each Month	Nom	Major Change Each 12 Months, Minor Each Month	**L**	**0.87**
	Nom−		L	0.87
	Low+		L	0.87
No Major Changes, Minor Change Each Year	Low		L	0.87
Process Volatility		**No Equivalent**		
PRODUCT DEVELOPMENT REQUIREMENTS				
Requirements Volatility (Change)		**No Equivalent**		

Specification Level - Reliability		**Required S/W Reliability (RELY)**		
Highest Reliability, Public Safety Requirements. Documentation is required for all aspects of system development, including architecture, design, programming, testing, and interface specifications. Most documentation will follow rigorous guidelines for content and format.	VHi	Error could cause loss of life. Some examples are space shuttle systems control software, medical machine control systems, nuclear reactor control systems, defense command and control systems.	**VH**	**1.26**
	VHi−		VH−	1.21
	Hi+		H+	1.15
Major Financial Loss. Typical Mil-Specs or other standards with full documentation.	Hi	Errors could cause major financial loss.	**H**	**1.10**
	Hi−		H−	1.07
	Nom+		N+	1.03
Moderate Loss, Recover Without Extreme Penalty; Typical Mil-Spec or other standards tailored to include complete essential documentation.	Nom	Errors could cause moderate financial loss.	**N**	**1.00**
	Nom−		N−	0.97
	Low+		L+	0.95
Low, Easily Recoverable From Loss. Minimal documentation	Low	The effect of failure is a low level, easily recovered loss to the users.	**L**	**0.92**
	Low−		L−	0.89
	VLo+		VL+	0.85
Slight Inconvenience. Documentation is not dictated or required. Any specifications created are incidental.	VLo	Impact of failure is usually an inconvenience to the developer. Many errors can be simply ignored.	**VL**	**0.82**
Test Level		**No Equivalent**		
Quality Assurance Level		**No Equivalent**		
Rehost from Development to Target		**No Equivalent**		
PRODUCT REUSEABILITY REQUIREMENTS				
Reusability Level Required		**Required Reusability (RUSE)**		
Mission Software Developed With Full Reusability Required. All Components of the Software Must Be Reusable. Reusability Is a Primary Objective of the Development Organization.	EHi	Designed for reuse planned and applied across multiple product lines.	**XH**	**1.24**
	EHi−		XH−	1.21
	VHi+		VH+	1.18
Software Will Be Reused Within A Single Product Line (Single Product Line: Multiple Contractors; Multiple/Single Customers)	VHi	Designed for reuse planned and applied across this product line.	**VH**	**1.15**
	VHi−		VH−	1.12
	Hi+		H+	1.10
Software Will Be Reused Within A Single Application Area (Single Application Area: Single Contractor; Multiple/Single Customers)	Hi	Designed for reuse across program (multiple projects).	**H**	**1.07**
	Hi−	Designed for reuse planned and applied across this project.	**N**	**1.00**
	Nom+		L−N	0.98

No Reusability Requirement	Nom	No concerted or planned attempt to design reusable components on this project.	**L**	**0.95**
Software Impacted By Reuse		**No Equivalent**		
DEVELOPMENT ENVIRONMENT COMPLEXITY				
Language Type (Complexity)		**No Equivalent**		
Development System Complexity		**No Equivalent**		
Application Class Complexity		**No Equivalent**		
Process Improvement		**No Equivalent**		
TARGET ENVIRONMENT				
Special Display Requirements		**No Equivalent**		
Memory Constraints		**Main Storage Constraint (STOR)**		
Complex Memory Management and Economic Measures	EHi	95% < Storage	**XH**	**1.46**
	EHi−		XH−	1.36
	VHi+		VH+	1.27
Extensive Overlaying or Segmentation	VHi	85% < Storage < 95%	**VH**	**1.17**
	VHi−		VH−	1.13
	Hi+		H+	1.09
Some Overlaying or Segmentation	Hi	70% < Storage < 85%	**H**	**1.05**
	Hi−		H−	1.03
	Nom+		N+	1.02
No Memory Constraints	Nom	Storage <70%	**N**	**1.00**
Time Constraint		**Execution Time Constraint (TIME)**		
	N/A	95% < expected CPU utilization	**XH**	**1.63**
75% of Code is Time Constrained	EHi	85% < expected CPU utilization	**VH**	**1.29**
	EHi−		VH−	1.23
	VHi+		H+	1.17
50% of Code is Time Constrained	VHi	70% < expected CPU utilization	**H**	**1.11**
	VHi−		H−	1.07
	Hi+		N+	1.04
25% of Code is Time Constrained	Hi	60% < expected CPU utilization	**N**	**1.00**
	Hi−		N	1.00
	Nom+		N	1.00
No Time Constraints	Nom		N	1.00
Real Time Code		**No Equivalent**		
Target System Complexity		**No Equivalent**		
Target System Volatility		**Platform Volatility (PVOL)**		
Major Change Each 2 Weeks, Minor 2 Times A Week	EHi	Major Change Each 2 Weeks, Minor 2 Times A Week	**VH**	**1.30**
	EHi−		VH−	1.25
	VHi+		H+	1.20
Major Change Each 2 Months, Minor Each Week	VHi	Major Change Each 2 Months, Minor Each Week	**H**	**1.15**
	VHi−		H−	1.10
	Hi+		N+	1.05
Major Change Each 6 Months, Minor Each 2 Weeks	Hi	Major Change Each 6 Months, Minor Each 2 Weeks	**N**	**1.00**
	Hi−		N−	0.96
	Nom+		L+	0.91
Major Change Each 12 Months, Minor Each Month	Nom	Major Change Each 12 Months, Minor Each Month	**L**	**0.87**
	Nom−		L	0.87

	Low+		L	0.87
No Major Changes, Minor Change Each Year	Low		L	0.87
Security Requirements		**No Equivalent**		
SCHEDULE AND STAFFING CONSIDERATIONS				
Required Schedule		**Time to Develop (TDEV)**		
Start Date		**No Equivalent**		
Complexity (Staffing)		**Product Complexity (CPLX)**		
Development primarily using micro code for the application, for example, a signal processing system with extremely complex interfaces and control logic. Staffing will increase very slowly.	EHi	Multiple resource scheduling with dynamically changing priorities. Microcode-level control. Distributed hard real-time control. Difficult and unstructured numerical analysis: highly accurate analysis of noisy, stochastic data. Complex paralleli-zation. Device timing-dependent coding, micro-programmed operations. Performance-critical embedded systems. Highly coupled, dynamic relational and object structures. Natural language data management. Complex multimedia, virtual reality, natural language interface.	**XH**	**1.74**
	EHi−		XH−	1.61
	VHi+		VH+	1.47
New systems with significant interfacing and requirements for interaction within a larger system structure. Examples include operating systems & real-time applications with significant logical code. Additional staff can only be added slowly.	VHi	Reentrant and recursive coding. Fixed-priority interrupt handling. Task synchronization, complex callbacks, heterogeneous distributed processing. Single-processor hard real-time control. Difficult but structured numerical analysis: near-singular matrix equations, partial differential equations. Simple parallelization. Routines for interrupt diagnosis, servicing, masking. Communication line handling. Performance-intensive embedded systems. Distributed database coordination. Complex triggers. Search optimization. Moderately complex 2D/3D, dynamic graphics, multimedia.	**VH**	**1.34**
	VHi−		VH−	1.28
	Hi+		H+	1.23
Applications with significant logical complexity, perhaps requiring changes to the host operating system, minor real-time processing or special displays and hardware. Staff may not be added quickly to the project.	Hi	Highly nested structured programming operators with many compound predicates. Queue and stack control. Homogeneous, distributed processing. Single processor soft real-time control. Basic numerical analysis: multivariate interpolation, ordinary differential equations. Basic truncation, round-off concerns. Operations at physical I/O level (physical storage address translations; seeks, reads, etc.). Optimized I/O overlap. Simple triggers activated by data stream contents. Complex data restructuring. Widget set	**H**	**1.17**

		development and extension. Simple voice I/O, multimedia.		
	Hi−		H−	1.11
Typical Command & Control Program	Nom+		N+	1.06
New standalone systems developed on firm operating systems. Minimal interface problems with the underlying operating system or other system parts. Complexity characteristics also map into processing as follows: Control Operations: mostly simple nesting,	Nom	Mostly simple nesting. Some intermodule control. Decision tables. Simple callbacks or message passing, including middleware-supported distributed processing. Use of standard math and statistical routines. Basic matrix/vector operations. I/O processing includes device selection, status checking and error processing. Multi-file input and single file output. Simple structural changes, simple edits. Complex COTS-DB queries, updates. Simple use of widget set.	**N**	**1.00**
	Nom−		N−	0.96
	Low+		L+	0.91
Software of low logical complexity using straightforward I/O and primarily internal data storage (typical of CSC and CSU work elements) . Additional staff may be added easily to the project.	Low	Straightforward nesting of structured programming operators. Mostly simple predicates. Evaluation of moderate-level expressions: e.g., D=SQRT(B**2-4.*A*C). No cognizance needed of particular processor or I/O device characteristics. I/O done at GET/PUT level. Single file subsetting with no data structure changes, no edits, no intermediate files. Moderately complex COTS-DB queries, updates. Use of simple graphic user interface (GUI) builders.	**L**	**0.87**
	Low−		L−	0.82
	VLo+		VL+	0.78
Extremely simple software with primarily straightforward code, simple I/O, and internal storage arrays (typical of CSC and CSU work elements). Staffing may increase rapidly.	VLo	Straight-line code with a few non-nested structured programming operators: DOs, CASEs, IF-THEN-ELSEs. Simple module composition via procedure calls or simple scripts. Evaluation of simple expressions: e.g., A=B+C*(D-E). Simple read, write statements with simple formats. Simple arrays in main memory. Simple COTS-DB queries, updates. Simple input forms, report generators.	**VL**	**0.73**
Staff Loading		**No Equivalent**		
Min Time vs. Opt Effort		**No Equivalent**		
STAFFING CONSTRAINTS				
Staff Level		**No Equivalent**		
REQUIREMENTS				
Requirements Complete at Start		**No Equivalent**		
Requirements Definition Formality		**No Equivalent**		
Requirements Effort After Baseline		**No Equivalent**		

SYSTEM INTEGRATION	
Programs Concurrently Integrating	**No Equivalent**
Concurrency of I&T Schedule	**No Equivalent**
Hardware Integration Level	**No Equivalent**

Tool Description

An excel tool has been built to automate conversion from SEER-SEM cost factors to COCOMO II EM factors using the Rosetta Stone depicted in Table 2. The concepts used had been previously implemented by Gary Thomas and Gary Constantine while they were at Raytheon. The SEER-SEM cost factors can be either pasted in or entered directly. The tool utilizes a center-weighted PERT to evaluate the three rating values mapped from each SEER-SEM cost factor: $\frac{\text{Lowest} + 4 \cdot \text{Most Likely} + \text{Highest}}{6}$. The converted ratings should be reviewed and adjusted as needed. The three EM factors not mapped from SEER-SEM cost factors must be entered manually. Next the five COCOMO II Scale Factors are entered. Finally COCOMO II ESLOC must be entered since SEER-SEM uses a different algorithm for ESLOC.

COCOMO II estimates the person-months required for the design, code, unit test, and software integration of a project. An initial estimate of person–months for a nominal schedule is generated based on the project size in Equivalent Lines of Code (ESLOC), sixteen Effort Multipliers, five Scale Factors and calibration constants. Note that labor for requirement activities are not estimated by COCOMO II. An adjustment is thus necessary prior to comparison with a SEER-SEM estimate. Equation 1 estimates person-months for a nominal schedule. [Boehm et. al. 2001]

$$PM_{NS} = A \cdot Size^{E} \cdot \prod_{i=1}^{16} EM_i \quad where\ E = B + 0.01 \sum_{j=1}^{5} SF_j \qquad Eq\ 1$$

- PM_{NS} – Effort in person-months for nominal schedule
- $Size$ – Equivalent source lines of code
- A – Effort (productivity) coefficient constant – 2.94 (for COCOMO II.2000 calibration)
- B – Scaling effort exponent base constant – 0.91 (for COCOMO II.2000)
- E – Scaling exponent expressing diseconomy of scale
- EM_i – Effort Multipliers from cost factors specific to Project estimated
- SF_j – Scale Factors from cost factors specific to Project estimated

COCOMO II also estimates nominal development schedule duration or Time to Develop, $TDEV_{NS}$ (Equation 2). The percentage of schedule compression, SP, is the required development schedule divided by the estimated nominal development schedule (Equation 3). If schedule compression exists, the seventeenth Effort Multiplier, Required Schedule (SCED), imposes a penalty for schedule acceleration with respect to a nominal schedule. The COCOMO II model specifies a SCED of 1.14 for a 85% schedule compression and a SCED of 1.43 for a 75% schedule compression. [Boehm et. al. 2001] To model this behavior, this tool introduces a 4th degree polynomial to compute SCED by interpolating between an uncompressed schedule and 75% schedule compression (Equation 4). With the final Effort Multiplier SCED computed, the tool produces a final estimate $\mathbf{PM_{CS}}$ which includes the effects of schedule compression by multiplying the initial estimate $\mathbf{PM_{NS}}$ of person–months for a nominal schedule with SCED (Equation 5).

$$TDEV_{NS} = C \cdot (PM_{NS})^{F} \quad where\ F = D + 0.2 \cdot 0.01 \sum_{j=1}^{5} SF_j = D + 0.2(E - B) \qquad Eq\ 2$$

$$SP = \frac{TDEV_{NS}}{SDEV} \qquad Eq\ 3$$

$$SCED \cong \begin{cases} 1.00 & SP \geq 1 \\ 16.444 \cdot SP^4 - 65.778 \cdot SP^3 + 104.52 \cdot SP^2 - 77.482 \cdot SP + 23.297 & 0.75 < SP < 1 \\ 1.43 & SP \leq 0.75 \end{cases} \qquad Eq\ 4$$

$$PM_{CS} = SCED \cdot A \cdot Size^{E} \cdot \prod_{i=1}^{16} EM_i = SCED \cdot PM_{NS} \qquad Eq\ 5$$

- $TDEV$ – Time to develop in calendar months
- B – Scaling effort exponent base constant – 0.91 (for COCOMO II.2000)
- C – Time to Develop calibration constant – 3.67 (for COCOMO II.2000)
- D – Schedule compression exponent base scaling constant – 0.28 (for COCOMO II.2000)
- E – Scaling exponent expressing diseconomy of scale
- F – Schedule compression adjusted scaling exponent expressing diseconomy of scale
- $SDEV$ – Project schedule calendar months
- SP – Schedule percentage of acceleration or stretch-out with respect to a nominal schedule
- PM_{CS} – Effort in person-months (adjusted if required schedule compressed)

Conclusions

This paper has presented a Rosetta Stone for converting SEER-SEM cost factors to COCOMO II Effort Multipliers. Techniques were detailed which map between disproportionate factor definitions. Following specifying the three Effort Multipliers not mapped from SEER-SEM (Personnel Continuity, Data Base Size, and Documentation) a composite Effort Multiplier is obtained.

A tool which automates cost factor transformations and produces a COCOMO II labor estimate was also described. A new technique has also been offered to calculate the final EM, SCED (Required Schedule), by interpolating between an uncompressed schedule (SCED = 1.00) and 75% schedule compression (SCED = 1.43) using a 4^{th} degree polynomial. The tool applies the SCED EM to compute the adjusted COCOMO II estimate.

References

[Boehm 1981] Boehm B., *Software Engineering Economics*. Englewood Cliffs, NJ, Prentice-Hall, 1981

[Boehm et. al. 2001] Boehem B, Abts C, Brown A, Chulani S, Clark B, Horowitz E, Madachy R, Reifer D, Steece B *Software Cost Estimation with COCOMO II*, Upper Saddle River, NJ, Prentice Hall, 2000

[Boehm et. al. 2005] Barry W. Boehm, Ricardo Valerdi, Jo Ann Lane, & A. Windsor Brown, "COCOMO Suite Methodology & Evolution", Crosstalk: The Journal of Defense Software Engineering, April 2005

[Galorath, 2003] "Converting SEER-SEM 6.0 to COCOMO II", unpublished Galorath Inc. draft, 2003

[Galorath-Evans 2006] Galorath D, Evans M, *Software Sizing, Estimation, and Risk Management*, Auerbach Publications, 2006

[Jensen 1983] Jensen R, "An Improved Macrolevel Software Development Resource Estimation Model", Proceedings of 5th ISPA Conference, 1983

[Madachy-Boehm 2006] Madachy R, Boehm, B, "Comparative Analysis of COCOMO II, SEER-SEM and True-S Software Cost Models", 21st International Forum on COCOMO and Software Cost Modeling, 2006