Cost Estimation for Secure Software & Systems

Ed Colbert Dr. Barry Boehm
Center for Systems & Software Engineering, University of Southern California,
941 W. 37th Pl., Sal 328, Los Angeles, CA 90089-0781
ecolbert@usc.edu

Background

The Center for Software Engineering (CSSE) at the University of Southern California (USC) is extending the widely–used *Constructive Cost Model* version 2 (*COCOMO II*) [4] to account for developing secure software. CSSE is also developing a model for estimating the cost to acquire secure systems (emphasizing space systems), and is evaluating the effect of security goals on other models in the COCOMO family. We will present the work to date.

Research Opportunity: Costing Secure Software & Systems

Engineering security in software–intensive system is now a high–priority objective (a) for the Aerospace Corporation, which is supporting this research, and the U.S. Federal Aviation Administration (FAA), which supported this research; (b) for the U.S. Government generally, and (c) for many industries. Prudent management is concerned about the life–cycle cost of security.

While it is widely held that engineering security will substantially raise software–project cost, different models vary in estimating the added cost. For example, Bisignani and Reed [3] estimates that engineering highly–secure software will increase costs by a factor of 8; the 1990's Softcost-R model estimates a factor of 3.43 [9].

Both of these models are based on the 1985 U.S. Department of Defense (DoD) Standard 5200.28-STD, "Trusted Computer System Evaluation Criteria" (called the "Orange Book") [8]. However, security engineering has changed since the Orange Book was considered the best practice. Since 1985, the software technology has changed to include features like distributed and mobile components, commercial off-the-shelf ("COTS") components, and the Internet. Developers must now consider high-risk threats not only from government agencies or amateur individuals but also from well-funded and possibly well-trained terrorist groups. The Orange Book, and cost models based on it, are obsolete. the Orange Book has been replaced by (a) the ISO Standard *Common Criteria for Information Technology Security Evaluation* (*CC*) [5-7] for commercial use; (b) the DoD 8500 series [1] for the U.S. military; (c) NIST 800 series [2] for non-DoD U.S. government agency.

The USC COCOMO II model [4] provides an excellent base for developing and calibrating a software cost driver for security. Its advantages include:

- A completely open description of its cost driver parameters and algorithms;
- Documented statistical calibration of its parameters to a body of carefully collected project data points;
- A proven methodology for extending the model to refine or add parameters;
- Compatibility with the current USC analysis to provide a COTS costing model (COCOTS).

In the first 1-year phase of research, USC developed a model for extending COCOMO II for development of secure systems. In the second 1.5-year phase, USC continued the behavior analysis needed for the COCOMO II model; analyzed the behavior for the Acquisition of Secure Cost Estimation model; ran 6 workshops to gather opinions from developers of software-intensive systems and experts in security; and gave presentations at conferences on general software development as well conferences on secure software development. As a result of these activities, USC created the following products.

- An update to the COCOMO II security extension ("COSECMO") that includes a new cost driver for security assurance activities ("SECU"), guidance on estimating the size of security functions, and guidance on setting levels of existing COCOMO drivers that are affected by the additional requirements for secure systems.
- A model for estimating total system acquisition costs, including life-cycle operations, maintenance, and disposal activity. This model is based on the FAA's work-breakdown structure and additional activities needed for secure systems.
- Two prototype tools, one for COSECMO and the other for the total acquisition cost.

The July 2003 FAA project workshop showed that the FAA could not wait through a long gestation period for this revised model, so USC has been developing these models incrementally (see table below). USC-CSSE has thus been able to identify tasks that can be done reasonably well with available information now, while USC-CSSE lays useful foundations for more powerful cost estimation later.

In December 2005, representatives of Aerospace Corporation reviewed both the COSECMO model and the System Acquisition model, resulting in a proposal to fund research in the application of these models to secure space systems.

Summary of Work Completed

Increment 1

During Increment 1, USC-CSSE developed a simple "rule of thumb" cost, based on simple cost-drivers like system precedents and how critical security will be. USC-CSSE also developed a Work Breakdown Structure (WBS) for security-related sources of effort, and identified the likely best types of Cost-Estimating Relationship (CER). This showed the scope for sources of cost to be estimated by the COCOMO II security extension. USC-CSSE also developed a secure-product taxonomy of product elements, and identified which of them should be addressed by the COCOMO II security extensions or elsewhere.

Increment 2

During Increment 2, USC-CSSE experimented with and refined the results of Increment 1. USC-CSSE (a) re-scoped and refined the initial COCOMO II security extensions, (b) revised the product taxonomy by identifying security requirements frequently used in each product type (based on the 149 *Security Targets* registered with the National Information Assurance Partnership (NIAP) Website); and (c) defined a model for costing the acquisition of secure systems. USC-CSSE reviewed the models with multiple groups of industry experts on security and system costing. USC-CSSE developed prototype tools that allowed reviewers to see the effect of different values for the cost drivers, and demonstrated them to FAA personnel. Experimenting with the tools, and getting FAA feedback showed us where the models matched the experts' expectations well, and where the models need to be refined to provide better CER's. USC-CSSE delivered copies of the two prototype cost-estimation tools.

Increment 3

During Increment 3, USC-CSSE presented the models to both security and costing experts at 3 workshops, hosted by USC-CSSE, and at conferences that focused on either security or costing. As a result of the feedback, USC-CSSE refined the models in a new way that has thus far satisfied the expectations of the experts, and has been calibrated with the first data point.

Developed as an extension of COCOMO II that is based on the Common Criteria v2, the new model appears to be adaptable to other models in the COCOMO family, other costing models, and other security standards (e.g. the Department of Defenses 8500 series; the National Institute of Standards and Technologies 800 series). The new

model appears to be flexible enough that it can be easily refined as the Cost-Element Relations are better understood. USC-CSSE delivered copies of the updated prototypes.

The "COSECMO Overview" section summarizes the security extension for COCOMO II.

Table: Revised Increment Plan for COCOMO II Security Extensions Project

Task Element	Increment 1 (February 2004 - July 2004)	Increment 2 (August 2004 - February 2006)	Increment 3 (March 2006 - February 2007)	Increment 4 (March 2007 - February 2008)*
1. Early Estimation Model for Acquisition of Secure Systems	• Prototype model	• Refine model • Develop prototype costing tool	• Collect & analyze expert opinion • Experimental use and refinement	• Evolution; integration with other models
2. Sources of Cost (Products, Activities, Services)	• Identify, define, scope sources of cost • Relate sources of cost to FAA Work Breakdown Structure (WBS) • Recommend type of Cost-Estimating Relation (CER) for each	• Prioritize sources of cost needing CER's • Refine, prototype, experiment with top-priority CER's	• Refine, prototype, experiment with top-priority CER's • Address lower-priority CER's as appropriate • Relate to scope of COCOMO II security extensions	• Refine sources of cost, CER's based on usage feedback • Integrate with other models
3. Secure Product Taxonomy (Product Elements)	• Identify, define, scope product elements • Relate to FAA WBS, sources of cost, COCOMO II security extensions scope	• Experimental use, feedback, and refinement	• Monitor evolution	• Monitor evolution
4. COCOMO II Security Extensions	• Refine model form and data definitions	• Refine scope, form, definitions based on results of Tasks 1-3 • Develop Prototype Tool • Collect & analyze expert opinion	• Baseline model definitions • Collect & analyze expert opinion • Collect project data • Experimentally apply to pilot projects, obtain usage feedback	• Collect project data • Develop initially calibrated model; experiment and refine

* The table shows Increment 4 (revised 2004). Funding ended (2006) and this Increment was omitted.

Increment 4

Increment 4 was not funded by the FAA due to changes in priorities. However, Aerospace Corporation has funded research on costing secure space systems.

The most recent detailed report on the new model can be in the workshop portion of CSSE's COCOMO forum or Annual Research Review and Executive forum Links to both events can be found on CSSE's events page, http://csse.usc.edu/events.

COSECMO Overview

COCOMO II estimates the effort to develop a software system based on the size of the software, and on certain product and process characteristics (called *drivers*) that affect the effort to develop the software. For example,

software that needs to be highly reliable generally takes more effort to produce than software of the same size which does not have to be demonstrably as reliable.

COCOMO uses the following formulas to compute effort.

$$Effort(Devel) = Effort(EC) \times 1.18$$

$$Effort(EC) = A \times (Size)^{E} \times \prod EMi$$

$$E = B + 0.01 \times \sum SFj$$

In the first equation, the 1.18 multiplier factors in the 6% for Inception effort and the 12% for transition effort which COCOMO typically uses.

COCOMO uses the following formulas to estimate the cost of and time to (called *schedule*) develop the software based on the estimated effort.

$$Cost = Effort \times Labor\ Rate$$

Where:

Effort(Devel)	Effort in person-months from the start of *inception* to the end of *transition* (*development life-cycle*)
Effort(EC)	Effort in person-months for *elaboration* & *construction* phases
A	Effort coefficient (nominally 2.94; can be calibrated to a specific development organization)
B	Scaling base exponent (nominally 0.91; can be calibrated to a specific development organization)
EM	Environment Multipliers (e.g. Reliability (RELY), Data)
SF	Process Scale Factors (e.g. Process Maturity (PMST))

(See the COCOMO II book for a detailed description of the COCOMO formulas and drivers.)

The COSECMO extension adds the effort to develop the additional software needed to implement the security functions, and the effort to assure that the system is secure. The following formulas compute the effort for the development of a software system whose security is assured (i.e. can be evaluated as reaching a specified assurance level).

$$Cost(Total\ Assured) = Cost(Assured\ Devel) + Cost(Independent\ Assurance)$$

$$Effort(Assured\ Devel) = Effort(Assured\ EC) \times 1.18$$

$$Effort(Assured\ EC) = Effort(EC) + Effort(Internal\ Assurance)$$

$$Effort(Internal\ Assurance) = Effort(EC) * \%Effort(AL)$$

where:

AL	*Assurance Level*
Effort(Assured Devel)	Effort in person-months from the start of inception to the end of transition (*development life-cycle*) for an assured, secure software system.
Effort(Assured EC)	Effort in person-months for elaboration & construction phases for an assured, secure software system.
Effort(Internal Assurance)	The additional developer assurance effort to verify that the system is secure beyond what is required for an ordinary highly reliable system.
Cost (Independent Assurance)	The additional cost to have an independent organization verify that the system is secure beyond what is required for an ordinary highly reliable system. Independent assurance is required by security standards for some levels of assurance. For consumer-oriented (& similar) software systems the developer typically pays to have an independent organization check the developer's work in order to sell the product to a particular market (e.g. the government). For non-consumer oriented (& similar) software systems, the customer typically pays to have an independent organization check the developer's work. In the latter case, the estimated cost in money to the developer is zero, but the schedule is increased by the time it takes to perform the independent analysis.
%Effort(AL)	Percent additional effort for developer assurance to verify that the system is secure beyond what is required for an ordinary -highly reliable system.

COSECMO uses two simplifying assumptions.

1. The labor rate for the developing organization equals the labor rate for the independent-assurance organization.

2. The assurance effort for the developer during elaboration and construction is no less than the assurance effort for the independent organization; the developer, wanting to be sure the software will pass independent evaluation, is likely to do everything the independent-assurance organization would.

If the internal-assurance effort is known, then it defines an upper bound on the estimated independent-assurance effort. If the independent-assurance is known, it defines a lower bound for the internal-assurance effort.

The percent additional effort for assurance (*%Effort(AL)*) depends on the security standard used for evaluation. For the *Common Criteria* v2, what USC-CSSE called *Assurance Level* (*AL*) is called *Evaluation Assurance Level* (*EAL*). (USC-CSSE is working on formulas for other standards.)

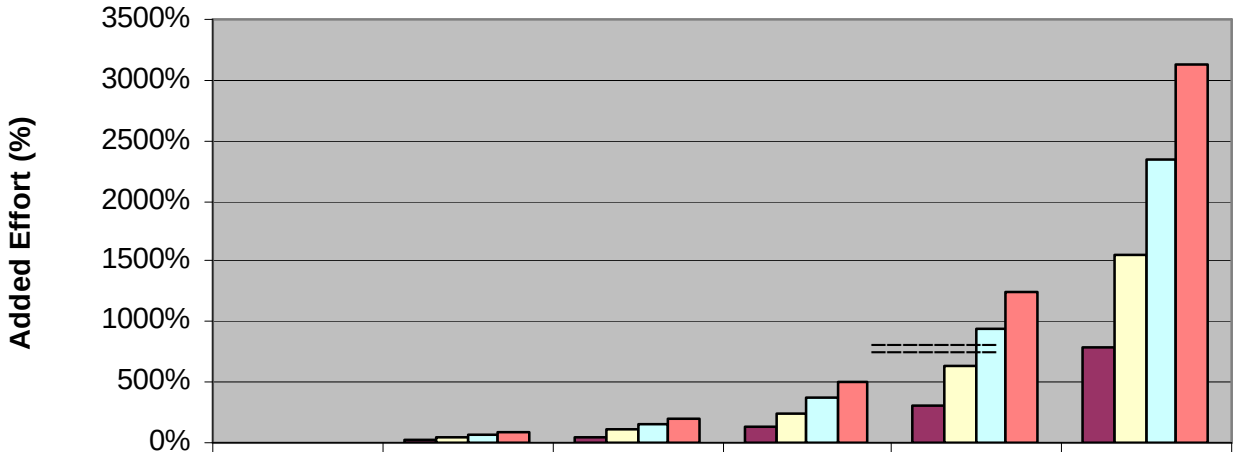
$$\%Effort(EAL) = \%Effort_3 * SECU^{(EAL - 3)} \quad \text{for } EAL >= 3$$
$$= 0 \quad \text{for } EAL < 3$$

where:

SECU	An effort multiplier for a one-level EAL increase (i.e. Effort(EAL n+1) = Effort(EAL n) * SECU) for EAL greater than 3.
%Effort(EAL)	Percent additional effort for developer assurance effort to verify that the system is secure beyond what is required for an ordinary -highly reliable system. Going from EAL 1 to EAL 2 does not appear to require significant additional effort.
$\%Effort_3$	The percent additional effort at EAL 3 (see Figure 1)

Figure 1 shows preliminary %Effort(EAL) for 4 system sizes, the Common Criteria's 7 EAL, and a SECU value of 2.5. These values assume that the base effort was computed with COCOMO II's Required Reliability driver (RELY) set to Very High for assurance levels 3 and above.

Figure 1 also shows how the 7 Common Criteria EAL relate to COCOMO II driver name-values (e.g. High), including 2 new driver values: Super High and Ultra High.



	Nominal EAL 1--2	High EAL 3	Very High EAL 4	Extra High EAL 5	Super High EAL 6	Ultra High EAL 7
5,000	0%	20%	50%	125%	313%	781%
20,000	0%	40%	100%	250%	625%	1563%
100,000	0%	60%	150%	375%	938%	2344%
1,000,000	0%	80%	200%	500%	1250%	3125%

Figure 1: Percent Additional Effort Based on Evaluated Assurance Level (EAL) & Size

Here are two examples that show how to apply the model.

Example 1: the average effort to produce 1,000 source lines of code (1,000 SLOC = 1 KSLOC) of telecommunications software is 4 person-months (PM) [Reifer, CrossTalk, March 2002]. So if a 5 KSLOC portion of a telecom system has an EAL of 4, then the base effort would be 20 PM, the additional assurance effort would be 10 PM, and the total effort would be 30 PM (a 50% increase), as shown in the following calculation.

```
Effort(Internal Assurance)   = 20PM * (20% * 2.5^(4 - 3))
                             = 20PM * (20% * 2.5)
                             = 20PM * 50%
                             = 10PM

Effort(Total)                = 20PM + 10PM = 30PM
```

The 20% in the first line is the EAL 3 value, Figure 1. With Figure 1 at hand, one can start at the third line by taking from the figure the EAL 4 value for 5 KSLOC at Very High assurance.

-Example 2: if the same 5 KSLOC system was developed with a desired security assurance level of EAL 7, then the additional assurance effort would be 156 PM and the total effort 176 PM (a 780% increase), as shown in the following calculation.

$$\begin{aligned} \text{Effort(Internal Assurance)} &= 20\text{PM} * (20\% * 2.5^{(7-3)}) \\ &= 20\text{PM} * (20\% * 2.54) \\ &= 20\text{PM} * (20\% * 39) \\ &= 20\text{PM} * 780\% \\ &= 156\text{PM} \end{aligned}$$

$$\text{Effort(Total)} = 20\text{PM} + 156\text{PM} = 176\text{PM}$$

The numbers in Figure 1 were derived using data from a real-time OS developer who reported that an independent evaluation agent quoted 50-70% of the developer's baseline software development effort to evaluate 5 KSLOC of infrastructure software (i.e. the OS kernel) at EAL 4, and 800-1000% per SLOC for EAL 7, with effort for intermediate levels scaling up proportionally. As noted above, USC-CSSE assumed for this model that the developer is likely to put in a similar amount of effort to evaluate software security prior to submitting the software for independent evaluation, so the developer's added effort to achieve software assurance is roughly equal to the effort to independently evaluate it.

Based on this assumption and the quoted effort for independent evaluation, USC-CSSE calculated for the values for the evaluation effort multiplier (SECU) per EAL and the base percent effort to evaluate an EAL 3 system ($\%Effort_3$). The $\%Effort_3$ values for the other software sizes were determined by questioning a small group of experts. The %Effort for other levels and other sizes were calculated using the formulas shown above.

Conclusions & Directions for Future Research

USC-CSSE has defined a model for costing software-intensive systems that satisfies the expectations and intuition of costing and security experts. USC-CSSE has calibrated this model based on 1 data point. The model seems to produce values that are consistent with costs of projects based on the Orange Book. However, more data are needed to better calibrate and then to validate the model.

USC-CSSE will continue refining, calibrating, and validating the COSECMO model as funding permits and as data become available. USC-CSSE will also explore how the model relates to other cost models in the COCOMO family (e.g. COCOTS - costing of systems that are mainly assembled from Commercial-off-the-Shelf software).

At this stage the model appears useful in testing prototype projects for secure software systems.

References

1. *Directive 8500.1*. 2002, U.S. Department of Defense:

2. *An Introduction to Computer Security: The NIST Handbook*, U.S.D.o. Commerce. 1995, Gaithersburg, MD.

3. Bisignani, M., and T. Reed, "Software Security Costing Issues", *COCOMO Users' Group Meeting*. 1988. Los Angeles: USC Center for Software Engineering.

4. Boehm, B.W., et al., *Software Cost Estimation with COCOMO II*. 2000, Prentice–Hall: Englewood Cliffs, NJ.

5. ISO JTC 1/SC 27, *Evaluation Criteria for IT Security*, in *Part 2: Security functional requirements*. 1999, International Organization for Standardization (ISO):

6. ISO JTC 1/SC 27, *Evaluation Criteria for IT Security*, in *Part 1: Introduction and general model*. 1999, International Organization for Standardization (ISO):

7. ISO JTC 1/SC 27, *Evaluation Criteria for IT Security*, in *Part 3: Security assurance requirements*. 1999, International Organization for Standardization (ISO):

8. National Computer Security Center, *Trusted Computer System Evaluation Criteria*. 1985, National Computer Security Center: Washington, D.C.

9. Reifer, D., *Security: A Rating Concept for COCOMO II*. 2002. Reifer Consultants, Inc.