Approaches to Calculating Systems Engineering Schedule in Parametric Cost Models

John E. Gaffney[1], Ricardo Valerdi[2] and Michael A. Ross[3]

[1] Lockheed Martin, USA, j.gaffney@lmco.com
[2] Massachusetts Institute of Technology, USA, rvalerdi@mit.edu
[3] r2Estimating, USA, mike.ross@r2estimating.com

Abstract
This paper provides two similar approaches for calculating project schedule from a systems engineering perspective. To illustrate the genesis of each approach, we provide two analogies; one from economics and one from physics. These are complemented with mathematical derivations that provide quantitative comparisons of the tradeoff between the duration of a project and the optimal effort. Connections are made to Books' Law and Parkinson's Law to validate the theoretical discussion with the pragmatic observations from the software engineering literature.

Keywords – cost estimation, schedule estimation, COSYSMO, COCOMO II.

1 Introduction

An issue of considerable importance to proposal managers, program managers, technical planners and to software engineering and systems engineering managers is how schedule compression or stretch-out affects engineering costs or overall project costs. Schedule compression (or stretch-out) is defined as the amount or percentage of reduction (increase) of a project or software or systems engineering schedule with respect to some ideal or nominal value of cost or productivity.

Understanding the relationship between cost (and productivity) and schedule (duration) can help us to answer such questions as:

What is the optimum duration to perform this task (e.g., development of a software system)?

Can schedule (duration) and cost (effort) be traded off; if so, what is the tradeoff?

Here, optimum might mean with respect to minimizing cost. More generally, it might mean with respect to some utility or value function of cost and duration, in which the utilities of the cost and schedule (duration) values are stated, indicating the relative importance of schedule (project duration) and cost. In the extreme, a project might be either cost driven or schedule driven.

We consider two approaches for calculating systems engineering schedule and discuss the assumptions of each. Both approaches use a parametric or top-down mathematical model to represent relationships among size (S), cost or effort (K), and schedule or duration (T) and have the following shared assumptions:

- Cost or effort (K) is a function of the size of the system (S)
- Schedule or duration (T) can be calculated by the cost or effort (K) produced by a parametric model
- Schedule or duration (T) reduction will result in a cost or effort (K) increase
- Organizations have process efficiencies that determine their shortest possible schedule or duration (T) of a project

The main difference between the two approaches is in the way they determine process efficiencies on projects. The first approach leverages the idea of the Cobb-Douglas production function that enables tradeoffs between effort and schedule. The second method uses an energy to work analogy to define a probabilistic content production relationship that enables tradeoffs between effort and duration with attendant confidence probability (cost and schedule risk) and then defines the notion of management stress to constrain and limit the content production relationship in terms of nominal (natural) management stress, minimum duration (Brook's Law), and minimum effort (Parkinson's Law). We provide derivations for both approaches and discuss the underlying assumptions that drive them. We conclude by summarizing how these two approaches can be reconciled to improve the COSYSMO systems engineering cost estimation model.

2 Why Estimate Schedule?

2.1 Background

An issue of considerable importance to proposal managers, program managers, technical planners, and to software engineering and systems engineering managers is how schedule compression or stretch-out affects engineering costs and overall project costs. Schedule compression (or stretch-out) can be defined as the amount or percentage of reduction (increase) of a project or software or systems engineering schedule with respect to some ideal or nominal value as related to cost or productivity. One difficulty is identifying whether there was compression, stretch-out, or a *normal* situation in any particular project instance. Another problem is knowing what is dependent and what is independent, or do we only know associations? This question relates to the fact that correlation does not mean causality. This paper does not deal with such philosophical

issues. Rather, it defines various relationships (equations) amongst size S, effort K, and schedule (duration) or T. It describes how the values of the parameters for such relationships can be obtained and how these relationships can be used to answer various questions of considerable practical value to estimators, and to managers and technical personnel (e.g., software engineers and systems engineers). However, the paper does consider the very important issue of practical limits on project performance, i.e., the *physics* of project execution (and the included or subsidiary software engineering and systems engineering tasks). This is in recognition, for example, that there is a limit to which project tasks can be subdivided enabling parallelism of project tasks to enable project schedules to minimized. This *physical* phenomenon is partially due to the increasing overhead due to increased amounts of communication that grows in proportion to n^2, where n is the number of units/persons that must communicate with each other.

2.2 Research Questions

A fairly safe assumption regarding any project is that every stakeholder has some degree of interest in how long a project will take to complete. Many business and technical decisions are based on some *estimate* or *best guess* about the duration and schedule of a project. Estimators and others are interested in being able to obtain answers to such questions as:

What is the expected percent change in effort for a stated percent change in schedule (duration)?

What is the expected change in effort (e.g., increase/decrease in person hours/months) that would correspond to a given change in schedule (e.g., increase/decrease in months/weeks)?

Depending upon the relationship between schedule and effort, there can be a positive or negative relationship between effort and schedule. Schedule compression (or stretch-out) can be defined as the amount or percentage of reduction (increase) of a project or software or systems engineering schedule with respect to some ideal or nominal value as related to cost or productivity. An issue of considerable importance to proposal managers, program managers, technical planners and to software engineering and systems engineering managers is how schedule compression or stretch-out affects engineering costs or overall project costs. Now, we provide a method to provide answers to questions, such as the two given above that relate to the (likely) affect of a change in schedule or effort. Knowing a relationship between cost (and productivity) and schedule (duration) can help us to answer such questions as:

(1) *"What is the optimum duration to perform this task (e.g., development of a software system)?"* Here, optimum might mean with respect to minimizing cost. More generally, it might mean with respect to some utility or value function of cost and duration, in which the relative importance or utilities of the cost and schedule (duration) values are stated. In the extreme, a project might be cost driven or schedule driven.

"Can schedule (duration) and cost (effort) be traded off; if so, what is the tradeoff?"

Often, the schedule for performing the systems or software engineering work on a project will be imposed upon those who will perform the systems or software engineering tasks. However, it may also be of interest and value to the person performing the estimate as well as the proposal, program or technical managers whom he is supporting, to determine what schedule or duration would be expected to correspond to the effort estimate produced by COSYSMO [1], COCOMO [2,3] or other resource (labor) estimation models. This value of schedule would be based on past project experience and might be smaller than, greater than, or equal to the schedule value that might be imposed upon those performing the systems engineering tasks. COSYSMO, COCOMO and the other members of the COCOMO *family* of estimation models estimate effort as their primary outputs. Consequently, a major topic of this paper is the estimation of the schedule that would correspond to the estimated effort and the possibility that exists in some cases for trade-offs between schedule and effort.

3 Defining Effort, Duration and Size

The following definitions are extracted from [4] and edited to suit the system engineering context of this paper. We first conceptually define staffing to be some function of elapsed calendar time t, $\boldsymbol{f_P}(t)$, that describes, for a particular instance of some task or collection of tasks, the application of people over time within the task's time interval. We define this time interval in absolute terms as $\left[\mathrm{T}_{\text{start}}, \mathrm{T}_{\text{finish}}\right]$ where $\mathrm{T}_{\text{start}}$ and $\mathrm{T}_{\text{finish}}$ represent the start and finish dates, respectively, of the task. In the interest of generalization, we prefer to use a $\mathrm{T}_{\text{start}}$-relative frame to describe this interval; therefore, $\mathrm{T}_{\text{start}}$ relative to $\mathrm{T}_{\text{start}}$ is $\mathrm{T}_{\text{start}} - \mathrm{T}_{\text{start}} = 0$ and $\mathrm{T}_{\text{finish}}$ relative to $\mathrm{T}_{\text{start}}$ is $\mathrm{T}_{\text{finish}} - \mathrm{T}_{\text{start}}$, the value of which we will represent as T. The resulting $\mathrm{T}_{\text{start}}$-relative task interval is $\left[0, T\right]$. Note that the value of T represents not only the $\mathrm{T}_{\text{start}}$-relative point in time where the task finishes, it also represents the duration (elapsed calendar time) of the task interval.

3.1 Effort

With our conceptual definition of a staffing function $\boldsymbol{f_P}$, we now define the concept of effort to be some function of elapsed calendar time t, $\boldsymbol{f_E}(t)$, that describes, for a particular task, the accumulated result of people laboring to do work over elapsed time t:

$$f_E \equiv \int f_P dt \quad \textbf{(1)}$$

Using Equation **(1)** as our definition of an effort function with respect to its associated staffing function we can now define an instantaneous staffing function with respect to its associated effort function by differentiating Equation **(1)** with respect to time t and then solving for f_P:

$$\frac{df_E}{dt} = \frac{d}{dt}\int f_P dt$$
$$\therefore f_P = \frac{df_E}{dt} \quad \textbf{(2)}$$

3.2 Task Effort

We have already defined $[0,T]$ to be the T_{start}-relative task time interval where T represents process duration. We now define task effort K to be the change in effort within this task time interval.

$$K \equiv \int_0^T f_P dt = f_E(T) - f_E(0) = f_E(T) \quad \textbf{(3)}$$

where $f_E(0) \equiv 0$; i.e., no task effort, by definition, can be spent before the task starts.

3.3 Effective Size (Content)

We describe system development processes as transforming one abstraction (the desire or the requirements) to another abstraction (the system product). Each and every abstraction, be it expressed in a natural language, as structured text, or even as graphic constructs; consists of primitive elements that we refer to as ***size units***. Examples of commonly-used size units in system and software engineering include operational scenarios, interfaces, algorithms, use cases, Source Lines of Code (SLOC), function points, objects, methods, classes, and web pages; basically something that can be *consistently* counted and reasonably represents the work that must be done. We choose here to define the notion of ***effective size*** S of a particular abstraction to be the number (count) of size units in the abstraction that are considered to be directly related to the resources (labor and time) necessary to develop said system; this includes developing new content plus selecting, understanding, incorporating, changing, and/or verifying any included legacy content[1].

[1] Examples of legacy content include Commercial Off-The Shelf (COTS) content, reused content, and content from a previous build, increment, or release.

4 Genesis of the Two Approaches

4.1 Economics Approach – Cobb-Douglas Form – COSYSMO-R

The basic approach considered here is to use a parametric or top-down mathematical model to represent relationships among size S, effort (cost) K, and duration (schedule) T. Note that S could be equivalent new source statements in the case of software, e.g. in the COCOMO tool, or equivalent new requirements in the case of systems engineering estimation, e.g., in the COSYSMO tool. So, what is a model (mathematical relationship) that might be used to help to answer the above questions? A generalization of the Cobb-Douglas production function is of the form

$$O = A\prod_{i=1}^{n} F_i^{Q_i} \quad \textbf{(4)}$$

where

O ≡ Output of the production process

A ≡ Total productivity factor (and potentially a scalar calibration constant)

$F_i^{Q_i}$ ≡ Factor of production

If we assume size to be the *output of the production process* and assume each of effort and duration to be *factors of production*, then we can notionally (ignoring, for the moment, the exponents of the production factors) state

$$Size = Productivity\ Factor \times Effort \times Duration \quad \textbf{(5)}$$

4.2 Physics Analogy Approach – Energy to Work Form – r2SEF

A branch of the software estimating world has been using an integrated effort and duration parametric estimating approach originally proposed by Putnam [5] as the SLIM model, improved upon by Jensen [6] as the Seer model, and extended by [4] as the r2 Software Estimating Framework (r2SEF). The approach is based on an analogy to the physics of directed energy into a process yielding work (a product) out of that process. Physics defines the notion of power as being the rate at which work is done; i.e., $Power \equiv Work/Time$. If we assume power is analogous to the cumulative effect of people laboring on a project (effort) and that work is analogous to produced system content (size), then we can notionally (ignoring, for the moment, possible nonlinearities) state

$$Effort \propto \frac{Size}{Time}$$
$$\therefore Size = Efficiency \times Effort \times Time \quad \textbf{(6)}$$

where

$Efficiency$ ≡ Constant of proportionality (and potentially a scalar calibration constant)

In this relationship the product of effort and time represents the energy applied to the process and size represents the work produced by the process. The symbol $\propto$ indicates that the two sides are proportional [4] has extended this idea to include the notion that defects are the undesired byproducts of the engineering development process; i.e., defects can be viewed as work (albeit undesirable) produced by the process.

5 Two Approaches – Derivation

5.1 Economics Approach – Cobb-Douglas Form – COSYSMO-R

Production functions such as the Cobb-Douglas Production Function were developed for use in economics analyses. This functional form relates an output, such as number of automobiles produced annually to number of labor hours and amount of invested capital. From this form can be derived a *productivity*, actually a *unit cost*, labor hours per automobile. Notice the indicated possible tradeoff between capital (appropriately applied in terms of training and technology) and labor; more capital implies less labor to obtain the same output.
Two additional, related examples are:

Inputs: invested capital, labor; output=$ profit. From this form can be derived a *productivity*, *return-on-invested capital* relationship and

Inputs: invested capital, labor; output=$ sales. From this form can be derived a *productivity*, *sales per employee* relationship.

Referring back to Equation **(4)** if all of the exponents Q_i are positive then the factors of production can be traded off, e.g., more of factor F_1 can make up for less of factor F_2 to yield the same product. For the situation addressed in this paper, we employ the following instance of the Cobb-Douglas production function:

$$S = AK^{p}T^{q} \quad \textbf{(7)}$$

This equation represents a production function to produce an output, of S, based on the factors of production, K (labor, person hours or person months), and T (time, duration, schedule, in weeks or months). S could be thousands of source statements in the case of software estimation such as with the COCOMO model or the number of equivalent requirements in the case of systems engineering estimation such as with the COSYSMO model. The factor A is a constant that captures the effects of other factors such as those dealing with the domain, the process (e.g., software development and testing, systems engineering).Thus, the constant A represents other factors of production not explicitly stated. It can also be viewed as equivalent to the product of the cost drivers in the COCOMO and COSYSMO (labor) resource estimation models. Note that Equation **(7)** is of the same form that is used in the SLIM software development resource estimation tool. The parameter r, where $r \equiv q/p$, is an important parameter as it characterizes the relationship between K and T, effort and schedule, and whether they can be traded off, and if so, the degree to which they can.
Three examples of the equation for S are now given that were developed from actual data that depend on the values of p and q. They cover three important alternative situations:

Case 1: If $p > 0$ and $q > 0$ then increasing values of K associate with decreasing values of T and vice versa; i.e., K and T can be traded off.
$S = A_1 K^{0.6288} T^{0.5555}$; $r = q/p = 0.8834$

Case 2: Low values of r (q/p), especially low q, mean little change in K, for a change in T, for a given value of S; effort is relatively insensitive to schedule and hence to a change to it. Only a very moderate degree of tradeoff between K and T is possible.
$S = A_2 K^{0.929} T^{0.079}$; $r = q/p = 0.0850$

Case 3: If $p < 0$ or $q < 0$ then increasing values of K associate with increasing values of T and vice versa; i.e., K and T cannot be traded off.

Now, we describe how to derive a specific instance of the relationship form shown in Equation **(7)**. It can be used to answer questions, such as the two above, which relate changes in effort as a function of changes in schedule. Suppose we want to determine the effect on K, effort, for a change in schedule from T_1 to T_2, and where S remains constant, $S_1 = S_2$, and the other conditions, as represented by A remain constant, $A_1 = A_2$. Now, we write the equation for S_2/S_1, getting

$$(1/1) = (1/1)(K_2/K_1)^{p}(T_2/T_1)^{q}$$

$$(K_2/K_1)^{p} = \frac{1}{(T_2/T_1)^{q}} \quad \textbf{(8)}$$

$$\therefore (K_2/K_1) = \frac{1}{(T_2/T_1)^{r}}$$

Suppose we want to answer the first earlier-posed question, *What is the expected percent change in effort for a stated percent change in schedule (duration)?* That is, we want to know the percent change in K corresponding to a percent change in T, then we use the relationship:

$$\Delta\%K = \frac{1}{(\Delta\%T)^{r}} \quad \textbf{(9)}$$

where

$\Delta\%K$ ≡ change from K_1 to K_2; $(K_2/K_1)\%$ and

$\Delta\%T$ ≡ change from T_1 to T_2; $(T_2/T_1)\%$

Plots of Equation **(9)** for the three *cases* given above are shown in **Figure 1** where:

Case 1: $\Delta\%K = 1/(\Delta\%T)^{0.8834}$

Case 2: $\Delta\%K = 1/(\Delta\%T)^{0.0850}$

Case 3: $\Delta\%K = 1/(\Delta\%T)^{-0.6827}$

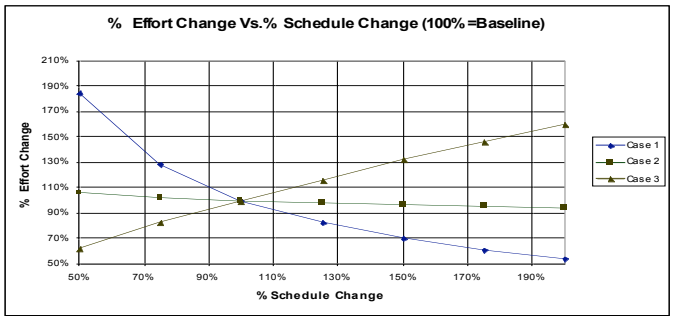

Figure 1 - *Three Cases of Effort versus Schedule*

A more general form for a software engineering, systems engineering, overall project function is shown in **Figure 2**. It combines parts of Cases 1, 2, and 3 to illustrate a composite behavior that might be found for one domain, organization, and/or, function over a range of schedule, effort and size values.

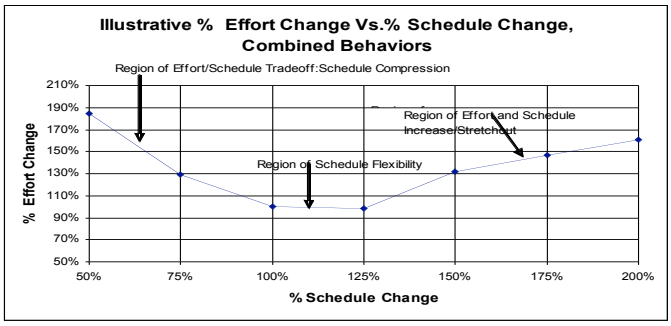

Figure 2 - *Effort versus Schedule; Combining Cases*

Equation **(8)** can also be used to answer the second earlier-posed question, *"What is the expected change in effort (e.g., increase/decrease in person hours/months) that would correspond to a given change in schedule (e.g., increase/decrease in months/weeks)?"* We rewrite this equation as

$$K_2 = \frac{K_1}{(T_2/T_1)^r}$$

$$\therefore K_2 = \frac{K_1}{(\Delta\%T)^r} \qquad \textbf{(10)}$$

K_1 could be a resource (labor) estimate produced by the COSYSMO or COCOMO model for some value of size S_1. Then, for a change in schedule from a baseline of $\Delta\%$ from a baseline value, the effort would go from K_1 to K_2. The estimation of the baseline schedule value is described in the next section.

5.2 Estimating Schedule

Now, we consider some other questions having to do with estimated effort and schedule and how they can be answered. As stated earlier, we focus on the use of estimating models such as COSYSMO and COCOMO that produce resource (labor) estimates as their primary outputs. The first question is:

> *"What is the value of T that corresponds to the value of E that was obtained using a model such as COSYSMO or COCOMO?"*

We want the value of T, call it T_1, that is the nominal or *natural* value of the schedule, that corresponds to the model-yielded value of K, call it K_1. The nominal or *natural* schedule value is based on past project experience for the relevant domain; captured and used as part of the tool calibration process. This value is *not* identified as being a compressed or a stretched-out schedule *unless* the project data used to develop the calibration was so identified. Rather, it is the schedule, T_1, that would be expected to correspond to an effort, K_1, to for a project size, S_1, based on past project experience. A subsidiary question is:

> *"Does the estimator accept this value of T, T_1; i.e., does it meet project criteria, or is it too large (long) or too small (short)?"*

Suppose that the answer is that the estimator does not accept this value, but rather wants a different value of T, call it T_2, say that is imposed by the program manager. Then, the next question is:

> *"What is the value of K, call it K_2, that corresponds to the desired value of schedule, T_2, e.g., one imposed upon the systems engineering job by the proposal manager or by the program manager?"*

We have already covered how this question could be answered in the preceding discussion. Now, we show how to determine the value of the schedule, T_1, that corresponds to an amount of labor, K_1, estimated by a model such as COSYSMO or COCOMO. We use an equation of the form

$$T = BS^v K^z \qquad \textbf{(11)}$$

where T, S, and K are corresponding values of schedule, size and effort (labor) as before. Thus, we obtain the value, T_1, using the relationship

$$T_1 = BS_1^{v} K_1^{z} \quad \textbf{(12)}$$

Note that one could use an equation of the form[2]

$$T_1 = CK_1^{d} \quad \textbf{(13)}$$

to obtain an estimate of T_1; however, this form is less accurate than the form of Equation **(12)**.

5.3 Physics Analogy Approach – Energy to Work Form – r2SEF

There are two hypotheses Suggested by historical data:

H1: ***Content (system size) is the desired product of labor and time***—The amount of size produced is directly related to the resource *amount* (labor and time) applied; i.e., the *product* of labor and time [4].

H2: ***Defects are the unwanted byproduct of labor and time***—The number of defects produced is directly related to resource *intensity* (labor and time); i.e., the *ratio* of labor and time [4].

In the effort-duration tradeoff relationship, c***ontent is made by people laboring to do work over some period of time; the result being neither free, instant, nor perfect.*** Effort K and duration T trend upward (and in most cases non-linearly) as functions of increasing size [4]. Our first empirically-suggested hypothesis states that ***content (system size) is the desired product of labor and time***—the amount of size produced is directly related to the resource amount (labor and time) applied; i.e., the product of labor and time. All three of these variables are uncertain and should therefore be treated as random variables[3]. We therefore propose the following generalized relationship:

$$f_K(\boldsymbol{K}) f_T(\boldsymbol{T}) \propto f_S(\boldsymbol{S}) \;\; \therefore f_K(\boldsymbol{K}) f_T(\boldsymbol{T}) = b f_S(\boldsymbol{S}) \quad \textbf{(14)}$$

where b represents the constant of proportionality.

Performing regression analysis on data from past projects suggests that both effort and duration are reasonably correlated with size and that these correlations can be generally and reasonably modeled by power functions [4] described as

$$f_K(x) \equiv x^{a_K}, f_T(x) \equiv x^{a_T}, \text{ and } f_S(x) \equiv x^{a_S} \quad \textbf{(15)}$$

Performing the algebraic manipulation described by Ross (2008) yields the following content production relationship:

$$\left[\boldsymbol{K}^{\alpha_K} \boldsymbol{T}^{\alpha_T} = \frac{\boldsymbol{S}}{\boldsymbol{ç}} \right]_{\text{<data set name>}} \quad \textbf{(16)}$$

Where:

α_K ≡ Exponent of effort; characterizes nonlinearity between effort and size[4]

α_T ≡ Exponent of duration; characterizes nonlinearity between duration and size[5]

$\boldsymbol{ç}$ ≡ Efficiency expressed as a random variable; characterizes the net effect of environmental (people, process, and product) factors that positively or negatively influence productivity

and where the square bracket symbols with a postfix subscript mean "within the context of the data set named **< data set name >** ".

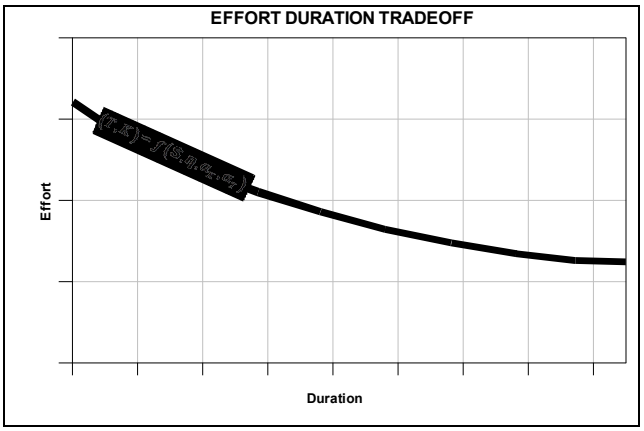

Figure 3 - *Example Tradeoff Curve (Content Productivity)*

Ross (2007) demonstrates how the content production relationship shown as Equation **(16)** can be instantiated with COCOMO II behavior and variables as

$$\left[\boldsymbol{K}^{(1-A)/B} \boldsymbol{T}^{1/B} = \left(\frac{\boldsymbol{S}}{1000} \right) \left(C_{K\,\text{nom}} C_{T\,\text{nom}} \prod_{i=1}^{n} (\mathbf{EM}_i) \right)^{(1/B)} \right]_{\text{COCOMO II}} \quad (17)$$

where

$C_{K\,\text{nom}}$ and $C_{T\,\text{nom}}$ are COCOMO II calibration constants,

$\mathbf{EM}_i$ ≡ the 6-element (COCOMO II Early Design) or 16-element (COCOMO II Post Architecture) vector (one-dimensional array) of COCOMO II effort multipliers, each expressed as a random variable which is triangularly-distributed according to a distribution

[2] This is the form of the duration estimating relationship currently used by COCOMO.

[3] Note that all random variables (i.e., variables that take on values described by some distribution) are formatted in ***Arial Bold Italic*** font.

[4] Note the *different* variables *a subscript K* in Equation **(15)** and *alpha subscript K* in Equation **(16)**.

[5] Note the *different* variables *a subscript T* in Equation **(15)** and *alpha subscript T* in Equation **(16)**.

parameter vector $\langle Low, Mode, High \rangle$,

B ≡ $\text{median}(\boldsymbol{B}) = \text{median}\left(B_{\min} + 0.01\sum_{i=1}^{5} \mathbf{SD}_i \right)$ where $B_{\min}$ is a COCOMO II calibration constant,

$\mathbf{SD_i}$ ≡ the 5-element vector (one-dimensional array) of COCOMO II scale drivers, each expressed as a random variable which is triangularly-distributed according to a distribution parameter vector $\langle Low, Mode, High \rangle$, and

A ≡ $\lambda + 0.2(B - B_{\min})$ where λ is a COCOMO II calibration constant.

5.4 Management Stress – Describing a Particular Effort-Duration Solution

The notion of management stress was suggested by [6] and described as the inherent equilibrium between effort and duration for *software* development processes, this equilibrium being independent of effective software size and efficiency and being constrained by a Rayleigh-shape staffing assumption unique to [5] and [6].

[4] chose to redefine this notion of management stress by eliminating the Rayleigh-shape staffing assumption constraint and by more-generally postulating that duration $\boldsymbol{T}$ is proportional to some function $\boldsymbol{f}$ of effort $\boldsymbol{K}$. In other words, for all task or project instances in a particular data set, ignoring the variety of sizes and efficiencies, as the effort increases, the duration tends to increase and *vice versa*. Stated mathematically

$$T \propto \boldsymbol{f}(K) \quad \therefore T = b\boldsymbol{f}(K) \qquad \textbf{(18)}$$

where b represents the constant of proportionality.

Performing the regression analysis and algebraic manipulation described by [4] yields:

$$\left[M = \frac{\boldsymbol{K}}{\boldsymbol{T}^{\gamma}} \right]_{\text{<data set name>}} \quad \left[\boldsymbol{K} = M\boldsymbol{T}^{\gamma} \right]_{\text{<data set name>}} \quad \left[\boldsymbol{T} = \left(\frac{\boldsymbol{K}}{M} \right)^{1/\gamma} \right]_{\text{<data set name>}} \qquad \textbf{(19)}$$

where

M ≡ Management stress (resource intensity, team communication complexity); higher values indicate a relatively larger effort over a relatively shorter period of time.

γ ≡ Economy or diseconomy associated with higher task or project durations.

A particular effort-duration solution for a given size $\boldsymbol{S}$ and efficiency $\boldsymbol{ç}$ can now be described by substituting first the solved for duration form of Equation **(19)** into Equation **(16)** and second the solved for effort form of Equation **(19)** into Equation **(16)** to yield the following two equations:

$$\left[\boldsymbol{K} = \left(M^{\alpha_T} \left(\frac{\boldsymbol{S}}{\boldsymbol{ç}} \right)^{\gamma} \right)^{1/(\gamma\alpha_K + \alpha_T)} \right]_{\text{<data set name>}} \qquad \textbf{(20)}$$

and

$$\left[\boldsymbol{T} = \left(\left(\frac{1}{M^{\alpha_K}} \right) \left(\frac{\boldsymbol{S}}{\boldsymbol{ç}} \right) \right)^{1/(\gamma\alpha_K + \alpha_T)} \right]_{\text{<data set name>}} \qquad \textbf{(21)}$$

and can, by taking the median of the convolved size/efficiency ratio $\boldsymbol{S}/\boldsymbol{ç}$ [6], be graphically illustrated as the intersection of Equation **(16)** and Equation **(19)** as shown in **Figure** 4.

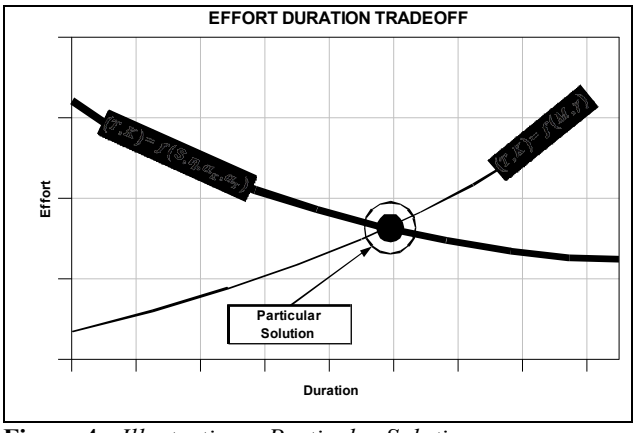

Figure 4 - *Illustrating a Particular Solution*

[7] demonstrates how the management stress relationship shown as Equation **(19)** can be instantiated with COCOMO II behavior and variables as

$$\left[M = \frac{\boldsymbol{K}}{\boldsymbol{T}^{1/A}} \right]_{\text{COCOMO II}} \quad \text{or} \quad \left[\boldsymbol{K} = M\boldsymbol{T}^{1/A} \right]_{\text{COCOMO II}} \quad \text{or} \quad \left[\boldsymbol{T} = \left(\frac{\boldsymbol{K}}{M} \right)^{A} \right]_{\text{COCOMO II}} \qquad \textbf{(22)}$$

where

$$M_{\text{nom}} = \frac{1}{C_{T\,\text{nom}}^{(1/A)}} \qquad \textbf{(23)}$$

A particular COCOMO II effort-duration solution for a given size $\boldsymbol{S}$ and set of effort multipliers and scale drivers (environment parameters) can now be described by the following two equations:

[6] Note that by taking the median of the convolved size/efficiency ratio, the resulting values for effort and duration are each 50% probability solutions.

$$\left[K = M^{A} C_{K\,\mathrm{nom}} C_{T\,\mathrm{nom}} \left(\frac{S}{1000} \right)^{B} \prod_{i=1}^{n} (\mathrm{EM}_i) \right]_{\mathrm{COCOMO\,II}} \quad (24)$$

and

$$\left[T = \left(M^{A-1} C_{K\,\mathrm{nom}} C_{T\,\mathrm{nom}} \left(\frac{S}{1000} \right)^{B} \prod_{i=1}^{n} (\mathrm{EM}_i) \right)^{A} \right]_{\mathrm{COCOMO\,II}} \quad (25)$$

5.5 Minimum Duration Limit – Brooks' Law

It has been well documented that a*dding manpower to a late software project makes it later* [8]. Each and every instance of a software development process (and by extension, each and every instance of a system development process), by its nature (divisibility or potential for concurrency), can effectively handle only so much management stress (only so many people); therefore, there exists, for each and every instance of the process, some minimum achievable process duration [4]. [5,6] and others have analyzed historical project data and concluded that this limit can be defined in terms of a project's maximum achievable management stress.

Mathematically, the $M_{\max}$ limit can be expressed in the extreme as

$$\left[M_{\max} \geq \max(\mathbf{M_i}) \right]_{\text{<data set name>}} \quad (26)$$

where

$\mathbf{M_i}$ = the vector of M values from the projects in data set **< data set name >**

Since this notion of the limit assumes all members of the data set to be within the realm of possibility for the next project (which may or may not be realistic), we suggest a conservatively-practical value for $M_{\max}$ in estimation situations to be the one-sigma percentage standard error of estimate %SE (aka standard multiplicative error) between the mean management stress $\bar{M}$ of the data set and the actual management stress M values of its members:

$$\left[M_{\max} \geq \left(1 + \%\mathrm{SE}(\mathbf{M_i})\right)^{+1.0} \bar{M} \right]_{\text{<data set name>}} \quad (27)$$

and where

$$\%\mathrm{SE}(\mathbf{M_i}) = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} \left(\frac{\bar{M} - \mathbf{M_i}}{\bar{M}} \right)^{2}} \quad (28)$$

5.6 Minimum Effort Limit – Parkinson's Law

Work expands so as to fill the time available for its completion [9]. Theoretically, a specific instance of an engineering development process is not limited by some maximum process duration. Rare is the engineer who complains about having too much time to develop something. However, we argue that there exists, for each and every development process instance, some duration that yields maximum productivity; i.e., some duration that represents the most efficient combination of problem decomposition and corresponding use of labor [4].

For each instance of a system development process, we submit that maximum productivity occurs at some point of minimum-practical management stress. This point of minimum practical management stress $M_{\min}$ defines the optimum use of people over time, and represents a practical limit to the benefit of schedule relaxation.

Mathematically, the $M_{\min}$ limit can be expressed in the extreme as

$$\left[M_{\min} \leq \min(\mathbf{M_i}) \right]_{\text{<data set name>}} \quad (29)$$

As with $M_{\max}$, we suggest a conservatively-practical value for $M_{\min}$ in estimation situations to be based on the percentage standard error of estimate %SE between the mean management stress $\bar{M}$ of the data set and the actual management stress M values of its members

$$\left[M_{\min} \leq \left(1 + \%\mathrm{SE}(\mathbf{M_i})\right)^{-1.0} \bar{M} \right]_{\text{<data set name>}} \quad (30)$$

The median case of the complete system of equations described thus far is shown in **Figure** 5.

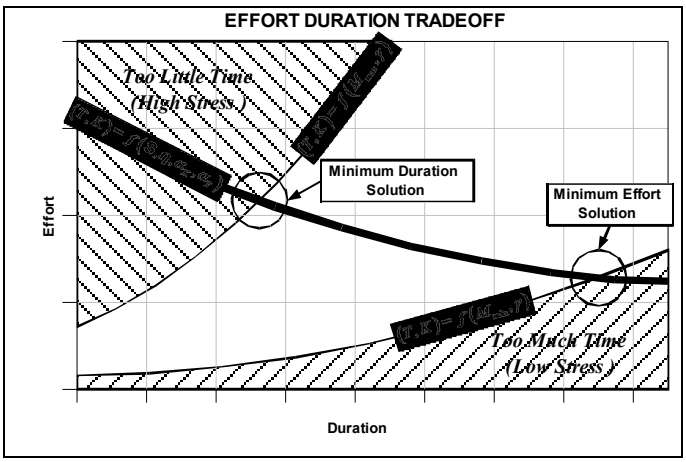

Figure 5 - *Minimum Duration and Minimum Effort Limits*

6 Observations and Next Steps

The two approaches for estimating project schedule as a function of cost provide useful frameworks for analysis. The most significant observation is that, despite the drastically different origins of the ideas, there is a convergence in the approaches for performing tradeoffs between project duration and effort. A logical next step is to integrate these approaches into the COSYSMO model to allow users to estimate systems engineering effort *and* schedule. This capability will provide a more complete assessment of the systems engineering project which will result in lower risk [10] and higher probability of success. This work is being done in context of the larger objective of

improving cost estimation by enhancing existing methods for capturing the economic impact of reuse [11] and harmonizing systems and software cost estimation [12].

7 References

[1] Valerdi, R. (2008), *The Constructive Systems Engineering Cost Model (COSYSMO): Quantifying the Costs of Systems Engineering Effort in Complex Systems*, VDM Verlag.

[2] Boehm, B.W. (1981), *Software Engineering Economics*. Englewood Cliffs: Prentice-Hall, Inc.

[3]Boehm, B.W., et al. (2000), *Software Cost Estimation with COCOMO II*. Upper Saddle River: Prentice-Hall, Inc.

[4] Ross, M. A. (2008), "Next Generation Software Estimating Framework: 25 Years and Thousands of Projects Later", *Journal of Cost Analysis and Parametrics,* 1(2), 7-30.

[5] Putnam, L.H. (1980), *Software Cost Estimating and Life-Cycle Control: Getting the Software Numbers*. New York City : IEEE Computer Society.

[6]Jensen, R.W. (1983), "An Improved Macrolevel Software Development Resource Estimation Model", Proceedings of the Fifth International Society of Parametric Analysts Conference. St. Louis, Missouri, USA, pp. 88-92.

[7] Ross, M. A. (2007), "Instantiating the r2 Software Estimating Framework for COCOMO", Procedings, 22nd International Annual Forum on COCOMO and Systems/Software Cost Modeling. Los Angeles, CA, USA.

[8]Brooks, F.P. (1995), *The Mythical Man-Month: Essays on Software Engineering*. Anniversary. Reading : Addison-Wesley Publishing Company.

[9] Parkinson, C.N. (1958), *Parkinson's Law: The Pursuit of Progress*. London : John Murray.

[10] Valerdi, R., Gaffney, J.E. (2007), "Reducing Risk and Uncertainty in COSYSMO Size and Cost Drivers: Some Techniques for Enhancing Accuracy", 5th Conference on Systems Engineering Research, Hoboken, NJ.

[11] Valerdi, R., Gaffney, J.E., Roedler, G.J., Rieff, J. (2006), "Extensions of COSYSMO to Represent Reuse", 21st Forum on COCOMO and Software Cost Modeling, Herndon, VA.

[12] Wang, G., Valerdi, R., Roedler, G.J., Ankrum, A., Gaffney, J.E. (2009), "Harmonizing Systems and Software Estimation", to appear in the 19th INCOSE Symposium, Singapore.