Staffing Implications of Software Productivity Models

R. C. Tausworthe
DSN Data Systems Section

This article investigates the attributes of software project staffing and productivity implied by equating the effects of two popular software models in a small neighborhood of a given effort-duration point. The first model, the "communications overhead" model, presupposes that organizational productivity decreases as a function of the project staff size, due to interfacing and intercommunication. The second, the so-called "software equation," relates the product size to effort and duration through a power-law tradeoff formula. The conclusions that may be reached by assuming that both of these describe project behavior, the former as a global phenomenon and the latter as a localized effect in a small neighborhood of a given effort-duration point, are that (1) there is a calculable maximum effective staff level, which, if exceeded, reduces the project production rate, (2) there is a calculable maximum extent to which effort and time may be traded effectively, (3) it becomes ineffective in a practical sense to expend more than an additional 25-50% of resources in order to reduce delivery time, (4) the team production efficiency can be computed directly from the staff level, the slope of the intercommunication loss function, and the ratio of exponents in the software equation, (5) the ratio of staff size to maximum effective staff size is directly related to the ratio of the exponents in the software equation, and therefore to the rate at which effort and duration can be traded in the chosen neighborhood, and (6) the project intercommunication overhead can be determined from the staff level and software equation exponents, and vice versa. Several examples are given to illustrate and validate the results for use in DSN implementation.

I. Introduction

Brooks (Ref. 1), in *The Mythical Man-Month*, proposed a simple model of software project intercommunication to show that, if each task of a large project were required to interface with every other task, then the associated intercommunication overhead would quickly negate the believed advantage of partitioning a large task into subtasks. While not meant to be an accurate portrayal of an actual project, the model effectively illustrated an increasing inefficiency symptomatic of projects too large to be performed by a single individual.

Putnam (Ref. 2), in a 1977 study of software projects undertaken by the US Army Computer Systems Command, discovered a statistical relationship among product *L*ines of code, *W*ork effort, and *T*ime duration for those projects whose best-fit formula was a power-law relationship, now referred to as the "software equation,"

$$L = c_k W^{0.33} T^{1.33}$$

(I have taken the liberty of changing Putnam's notation in order to be consistent with my notation in the remainder of the article.)

One rather startling extrapolation one may make from the software equation is that in order to halve the duration of any one of the projects studied, it would have taken 16 times the resources actually used! I say "extrapolation" because I suspect the software equation is more likely to be applicable incrementally—that is, if one were to require a 5% shortening of the schedule, then a 20% (actually 21.5%) increase in resources would be required.

In this paper, I will generalize both of these models parametrically, and suppose that both do describe the statistical trends of software projects in small neighborhoods about a chosen project situation. By equating the model behaviors in these neighborhoods, we shall be able to see how the parameters of one model relate to the parameters in the other. In addition, we shall discover some rather interesting facts about some actual projects for which published data exists.

II. A Generalized Intercommunication Overhead Model

Let us suppose that a software project is to develop L kilo-*L*ines of executable source language instructions, and that this number remains fixed over all our considerations of effort, duration, staffing, etc. That is, we shall suppose that the product size is invariant over the neighborhood of variability in these parameters—a project utilizing greater effort attempting to shorten the schedule slightly would produce the same program as a smaller effort requiring somewhat more time.

Let us denote by W the *W*ork effort (in person-months) to be expended in the production of the L lines of code, and let the *T*ime duration, in months, be denoted by T. Then the average full-time equivalent *S*taff size S in persons, is

$$S = \frac{W}{T}$$

and the overall team productivity can be defined as the number

$$P = \frac{L}{W} \text{ (kilo-lines/person-month)}$$

Let us further suppose that the average fraction of *t*ime that each staff member spends in intercommunication overhead is dependent on the staff size alone, within a particular organizational structure and technology level, and let this fraction be denoted by $t(S)$:

$$t(S) = \frac{\text{(intercommunication time/mo.)}}{\text{(hours/mo. worked)}}$$

Generally speaking, one intuitively expects $t(S)$ to increase monotonically in S due to the expanding number of potential interfaces that arise as staff is increased. But the *individual average productivity* of the staff, defined as the individual productivity during nonintercommunication periods, P_i, is somewhat greater than P, being related to it by

$$P = P_i\,[1 - t(S)]$$

The relationship between the number of kilo-lines produced, the effort, and the staffing is

$$L = P_i W\,[1 - t(S)]$$

Let us denote by W_0 and T_0 the effort and time, respectively, that would be required by a single unencumbered individual to perform the entire software task (assuming also that it could be done entirely by this individual, no matter how long it took). Then, with respect to the actual W and T, there is the relationship

$$W_0 = \frac{L}{P_i} = W\,[1 - t(S)] = T_0$$

This W_0 represents the least effort that must be expended, and T_0 is the maximum time that will be required. By substituting W/T for S, one obtains an effort-time tradeoff relationship

$$\omega = \frac{1}{[1 - t(\omega/\tau)]}$$

where $\omega = W/W_0$ and $\tau = T/T_0$ are "normalized" effort and duration, respectively.

The rate at which an increase in staffing results in an increase in normalized work effort is then

$$\frac{\partial\omega}{\partial S} = \omega^2\, t'(S) > 0$$

where $t'(\ )$ refers to the derivative of t with respect to S. Because of the monotone character of $t(S)$, an increase in staff leads to an increase in effort.

The overall staff production *R*ate R is the number of kilo-lines of code *per month* produced by the entire team of S persons,

$$R = P_i\, S\,[1 - t(S)]$$

The factor

$$\eta = [1 - t(S)]$$

is then the *team production efficiency*. Note that the normalized task effort is the inverse of the production efficiency,

$$\omega = \frac{1}{\eta}$$

The maximum rate of software production will occur when the derivative of R with respect to S becomes zero, a condition requiring a value S_0 that will satisfy the relationship

$$t'(S_0) = \frac{[1 - t(S_0)]}{S_0}$$

We shall refer to this staffing level as the *maximum effective staff*. Two particular examples of $t(S)$ will serve to illustrate the characteristics of the intercommunication overhead model.

A. Linear Intercommunication Overhead

Let us assume first, as did Brooks, that the overhead is linear in staff,

$$t(S) = t_0(S - 1)$$

That is, there is no overhead for 1 person working alone, but when there are $S - 1$ other people, then each requires an average fraction t_0 of every other individual's time. Under these assumptions, the maximum effective staff level is

$$S_0 = \frac{(1 + t_0)}{(2\,t_0)}$$

This value yields a maximum team production rate of

$$R_{max} = \frac{P_t S_0^2}{(2\,S_0 - 1)}$$

and team efficiency

$$\eta_0 = \frac{(1 + t_0)}{2 = S_0}\,(2\,S_0 - 1) \approx 0.5$$

This perhaps alarming result states that a team producing at its maximum rate is burning up half its effort in intercommunication overhead! The behavior is illustrated in Fig. 1.

The normalized effort-duration tradeoff equation for this model takes the form

$$\tau = \frac{t_0\,\omega^2}{(1 + t_0)\,\omega - 1}$$

which has its minimum value at the maximum-production-rate point,

$$\tau_{min} = \frac{4\,t_0}{(1 + t_0)^2} \approx 4\,t_0$$

at which point the normalized effort is

$$\omega_0 = \frac{2}{(1 + t_0)} < 2$$

Figure 2 shows the characteristic of this tradeoff law at t_0 values of 0.1 and 0.2 for illustrative purposes.

According to this model, it *never* pays to expend more than twice the single-individual effort. Moreover, even though the ω producing the shortest schedule is less than 2, the cost-effective range is much less than this, as shown in the figure. Effort can be traded for schedule time realistically only up to about 1.25 W_0, and a factor of 2 reduction in time can only come about if the individual intercommunication can be kept below about 15% per interface.

B. Exponentially Decaying Intercommunication Overhead

One unsettling aspect of the linear intercommunication overhead model is that, at some staffing level, the production rate goes to zero, and beyond, unrealistically into negative values. Perhaps a more realistic model is one which assumes that $t(S)$ tapers off, never exceeding unity, at a rate proportional to the remaining fraction of time available for intercommunication as staff increases, or

$$t'(S) = t_1\,[1 - t(S)]$$

Then we are led to the form

$$t(S) = 1 - \exp\,[-t_1(S - 1)]$$

The maximum effective staff in this case becomes

$$S_0 = \frac{1}{t_1}$$

and the maximum production rate is

$$R_{max} = P_t\,S \exp\left(-1 + \frac{1}{S}\right) \approx \frac{P_t S}{e}$$

The team effiency at this rate is

$$\eta_0 = \exp\left(-1 + \frac{1}{S}\right) \approx \frac{1}{e}$$

Now this is perhaps even more alarming a revelation than before, because it says that when producing software at the maximum team rate, that team is burning up 63% of its time in intercommunication! The consolation, as shown in Fig. 1, is that the team performance under this assumed model is superior to that of the linear-time team model. More staff can be applied before the maximum effective staff level is reached.

The effort-duration tradeoff equation according to this model is

$$\tau = \frac{t_1 \omega}{[t_1 + \ln(\omega)]}$$

The minimum τ occurs at

$$\omega_0 = \exp(1 - t_1) < e$$

and the minimum value is

$$\tau_{min} = t_1 \exp(1 - t_1) \approx e\, t_1$$

The form of this tradeoff is shown in Fig. 3 for t_1 values of 0.1 and 0.2, for illustrative purposes. Note that the minimum τ is much broader in this model, so that, although the actual minimum occurs when ω is about e in value, the practical effective range for ω is less than about 1.5. That is, it is not cost-effective to expend more than about 1.5 times the single-individual effort W_0 in an attempt to reduce the schedule time. A reduction in schedule by a factor of 2 is possible only when the individual intercommunication factor t_1 can be kept below 0.2.

C. Conclusions from Intercommunication Overhead Models

Both of the examples of intercommunication overhead above bespeak a maximum effective staffing level at which the project is 37-50% efficient. Beyond this point, further staffing is counterproductive. Both examples conclude that the maximum practical extent to which added effort is effective in buying schedule time is limited to about 25-50%. Significant schedule reduction factors are possible only when the intercommunication factors can be kept below 15-20%.

III. Matching the Software Equation Model

Let us generalize the Putnam Software Equation as the form

$$L = c_k W^p T^q$$

and let is define $r = q/p$, the exponent ratio. As in the previous section, L is held constant with respect to effort-duration tradeoff considerations. The value of p is assuredly positive: it generally requires more work at a given L to reduce T. If q is positive, effort can be traded to decrease the schedule time required to deliver a given L. The larger r is, the larger the increase in effort required to shorten the schedule, and the larger the team production inefficiency. If q is zero, then L is a function of W alone, T is determined solely by the staffing level, $T = W/S$, and no additional effort is required to reduce schedule time (in the neighborhood in which the p and $q = 0$ are valid). If q were ever to be negative, then an increase in W would render an increase in T, a situation indicating overmanned projects.

Substitution of $T = W/S$, differentiation with respect to S, and normalization of the software equation produces the result

$$\frac{\partial \omega}{\partial S} = \omega \frac{r}{[S(1+r)]} = \tau \frac{r}{(1+r)}$$

Let us now suppose that both the software equation and the intercommunications overhead model agree at the point (L, W, T). The two models can be equated by suitable choices of the "technology constant" c_k and individual productivity P_i. Then, in addition, let us suppose that the derivatives of effort with respect to staff level for both models also agree at this point. Such can only be attempted when $r > 0$, because the derivative in the intercommunication overhead model is always positive. When this is the case, the two models may be said to agree in the neighborhood of the point (L, W, T).

Thus, by equating the derivatives, we arrive at a relationship between the parameters of the two models:

$$\frac{S\, t'(S)}{[1 - t(S)]} = \frac{r}{1 + r}$$

or

$$\eta = S\, t'(S)\, \frac{r+1}{r}$$

Let us now examine this relationship for the two examples of the interface overhead model:

A. Linear Intercommunication Overhead

Substitution of the linear $t(S)$ form into the neighborhood agreement condition yields

$$S = \left[\frac{2r}{1+2r}\right]\left[\frac{1+t_0}{2t_0}\right] = S_0 \; \frac{r}{r+0.5}$$

This equation states that the staffing level is related to the maximum effective staff point through the software exponent ratio r. At the Putnam value, $r = 4$, the staffing level is 89% of the maximum effective level, and the team efficiency is

$$\eta = 0.55\,(1 + t_0) \approx 55\text{-}65\%$$

$$\omega = \frac{1.8}{1+t_0} \approx 1.5 - 1.8$$

As seen in Fig. 2, projects having this high an ω are at the point that extra effort is very ineffective.

B. Exponentially Decaying Intercommunication Overhead

By substituting the exponential form for $t(S)$ into the neighborhood agreement condition, we find

$$S = \frac{r}{[t_1\,(1+r)]} = S_0 \, \frac{r}{(1+r)}$$

Again, we see that the staffing level is related to the maximum effective staff via the exponent ratio. The Putnam value $r = 4$ produces

$$S = 0.8\,S_0$$

$$\eta = \exp \; \frac{(S-1)}{S_0} = \exp\,[-0.8 + t_1] \approx 45\% - 55\%$$

$$\omega = \frac{1}{\eta} = \exp\,[0.8 - t_1] \approx 1.8 - 2.2$$

Although this example indicates a somewhat more comfortable margin below maximum effective staffing than did the linear model, it nevertheless shows an alarmingly low cost inefficiency.

IV. Examples Using Available Data

Several data sets of project resource statistics published in the literature readily show that Putnam's value of $r = 4$ is not universal. Specifically, Freburger and Basili (Ref. 3) publish data which yield the following 3-parameter best power-law fits:

$$L_0 = 1.24\,W^{0.95}\,T^{-0.094} \qquad (r = -0.1)$$

$$L_1 = 0.22\,W^{0.78}\,T^{0.78} \qquad (r = 1.0)$$

in which L_0 is kilo-lines of delivered code, and L_1 is *developed* delivered code. It is interesting here to note that the former relationship is nearly independent of T, whereas the latter shows a definite beneficial $W - T$ tradeoff characteristic. The negative q in the former relationship indicates that, on a *delivered code basis*, added resources in one of the projects would have extended the schedule! An equivalence between the software equation and the intercommunication overhead model cannot be established when r is zero or negative.

This data set is not the only one to show a negative q: Boehm (Ref. 4), in his *Software Economics* book, has a data base used to calibrate his COCOMO software cost model. A 3-parameter best power-law fit to the adjusted data produces the relationship

$$L = 0.942\,W^{0.675}\,T^{-0.028} \qquad (r = -0.041)$$

and, on the unadjusted data,

$$L = 0.957\,W^{0.646}\,T^{0.0555} \qquad (r = 0.086)$$

Gaffney (Ref. 5), on the other hand, did a 3-parameter best power-law fit of IBM data (Federal Systems Division, Manassas) to arrive at the relationship

$$L = c_k\,W^{0.63}\,T^{0.56} \qquad (r = 0.88)$$

This last value of r aligns more closely with the Freburger-Basili value for developed delivered code.

Figure 4 shows plots of staffing normalized to maximum effective levels as a function of r, for both examples of the intercommunication overhead model. Efficiency curves are shown in Fig. 5.

V. Conclusion

This article has shown that when there is a positive effort-duration tradeoff relationship in a software project, it is possible to estimate the team production efficiency and proximity to maximum effective staffing. These figures can be used to advantage by software managers who must judge the effectiveness of increasing resources in order to shorten schedules. It points out the necessity of keeping accurate records of software project statistics, so that the parameters in the model can be estimated accurately.

Low values of r in an organization are a mark to be proud of, showing efficiency in terms of structuring subtasks for clean interfaces. High (or negative) values of r may be indicative of overall task complexity, volatility of requirements, organizational inefficiency, or any number of other traits that tend to hinder progress. The value of r may thus be treated as a figure of merit–a measurable statistic indicative of the efficiency of a set of projects in performance of assigned tasks.

The ratio S/S_0 is another indicator for management. When low, it indicates that adding resources can potentially help a project in schedule trouble. If closer to unity, it is a warning that adding resources may not help, will not appreciably shorten the schedule, will incur expense at a low return in productivity, and, if applied often in other projects, will thereby contribute to an organizational reputation for expensive software.

References

1. Brooks, F. P., *The Mythical Man-Month*, Addison-Wesley Publishing Co., Reading, Mass., 1975.
2. Putnam, L. H., "Progress in modeling the software life cycle in a phenomenological way to obtain engineering quality estimates and dynamic control of the process," *Second Software Life Cycle Management Workshop*, sponsored by U.S. Army Computer Systems Command and IEEE Computer Society, Atlanta, Ga., Aug. 1978.
3. Freburger, K., and Basili, V. R., "The Software Engineering Laboratory, Relationship Equations," Report TR-764, University of Maryland Computer Science Center, College Park, Md, May 1979.
4. Boehm, B. W., *Software Economics*, Prentice-Hall Publishing Co., Englewood Cliffs, N.J., 1982.
5. Gaffney, J. E., "An Approach to Software Cost and Schedule Estimation," submitted to *Journal of Defense Systems Acquisition Management* (pending).

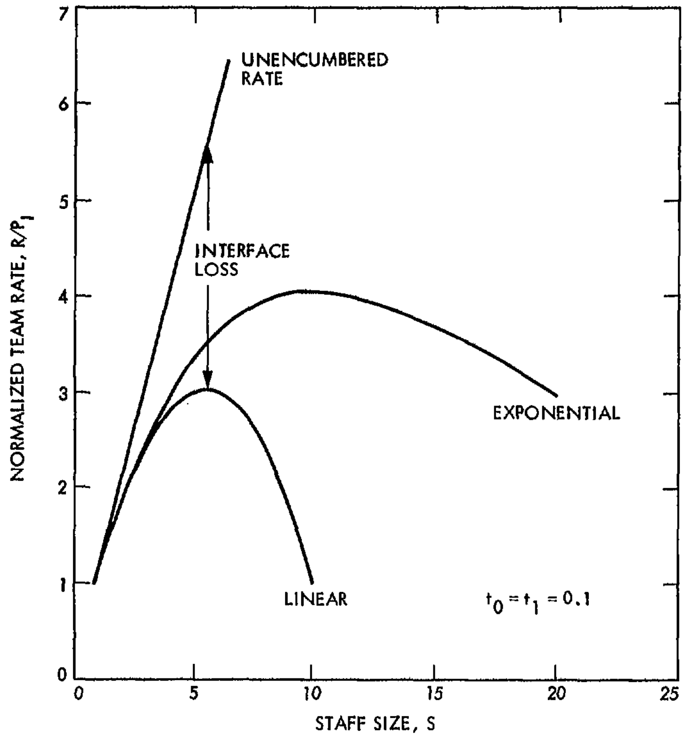

Fig. 1. Software team production rate, intercommunications overhead model

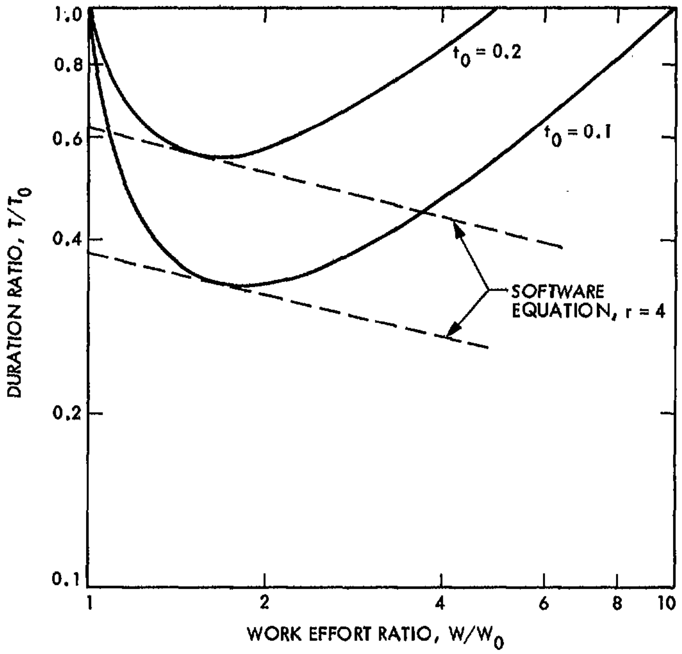

Fig. 2. Time duration vs effort tradeoff curve for linear intercommunication overhead model

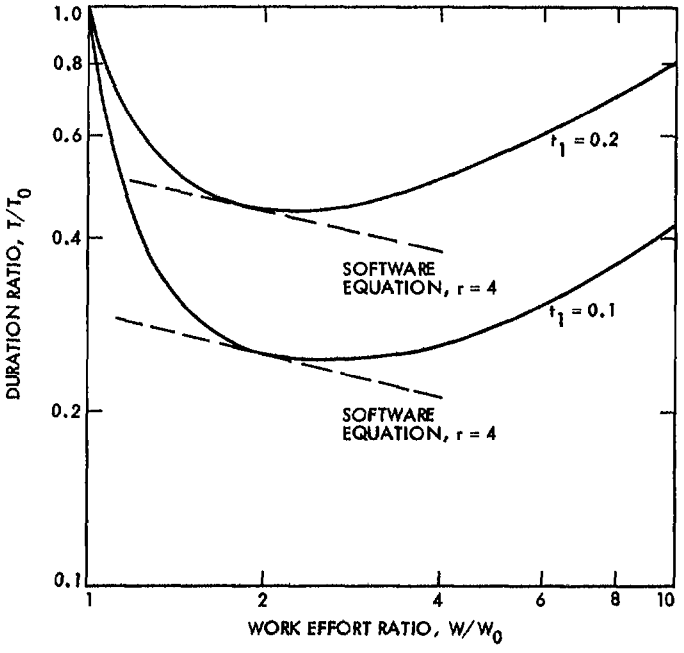

Fig. 3. Time duration vs work effort tradeoff characteristic for the decaying exponential intercommunication overhead model

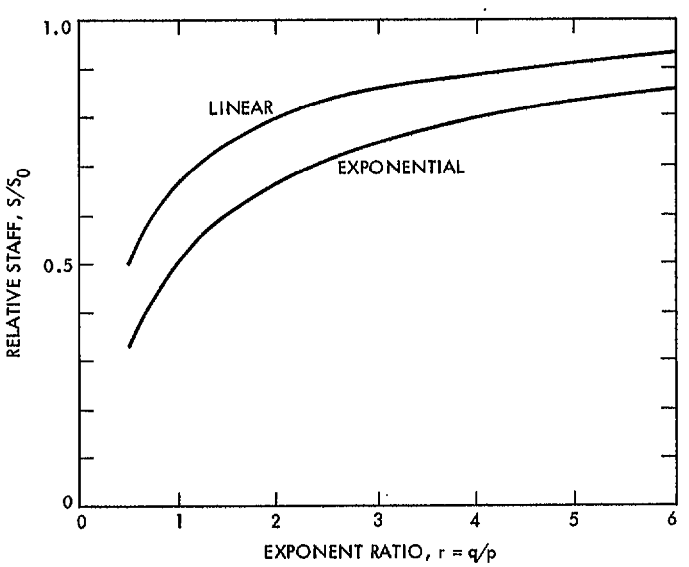

Fig. 4. Normalized staff levels for the intercommunication overhead model

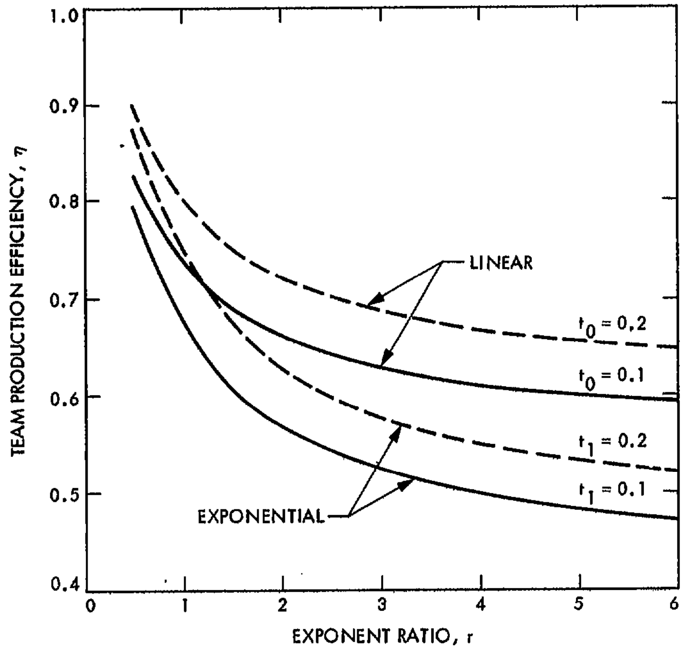

Fig. 5. Team production efficiency of projects as a function of the exponent ratio and intercommunications overhead parameter