SOFTWARE LIFE CYCLE DYNAMIC SIMULATION MODEL: THE ORGANIZATIONAL PERFORMANCE SUBMODEL*

3/1/85

N87-29143

p. 25

Robert C. Tausworthe
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109

ABSTRACT

This paper describes the submodel structure of a software life cycle dynamic simulation model. The software process is divided into seven phases, each with product, staff, and funding flows. The model is subdivided into an organizational response submodel, a management submodel, a management influence interface, and a model analyst interface. The paper concentrates on the organizational response model, which simulates the performance characteristics of a software development subject to internal and external influences. These influences emanate from two sources: the model analyst interface, which configures the model to simulate the response of an implementing organization subject to its own internal influences, and the management submodel that exerts external dynamic control over the production process.

The paper provides a complete characterization of the organizational response submodel in the form of parametrized differential equations governing product, staffing, and funding levels. The parameter values and functions are allocated to the two interfaces.

* This work reported in this document was performed at the Jet Propulsion Laboratory of the California Institute of Technology under a contract with the National Aeronautics and Space Administration.

SOFTWARE LIFE CYCLE DYNAMIC SIMULATION MODEL: THE ORGANIZATIONAL PERFORMANCE SUBMODEL

Robert C. Tausworthe
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109

1. INTRODUCTION

In earlier papers [1, 2], the author, in collaboration with Chi Lin and Merle McKenzie, exposed structural design concepts for the construction of a dynamic simulation model of the software life cycle process. These works derived requirements on the form and granularity of the activity breakdown necessary for an accurate simulation. In subsequent works, Don Reifer [3, 4] produced a generic software life cycle work breakdown structure having the required level of detail, and studied the infrastructural dependencies among rates of production, utilization of staff and funding resources, product size characteristics, and situational and environmental factors.

This paper is an extension of these works, describing further structural details of the model, the organization of the overall model into submodels, and the description of one of these submodels in detail.

2. MODEL STRUCTURE

2.1. Core Unit Structure

The works cited above describe the software life cycle process as a dynamic cyclic architecture of project phases, each broken into its constituent unit-task-level activities and flows of products, personnel, funding, and other resources among the activities. Each activity is viewed as having a common structure, referred to as the 'core unit,' shown in Figure 1. The cylindrical 'tank' symbols in the figure refer to quantities, or 'levels,' that may flow within the model. Directed arrows denote paths of flow, and the oblong symbols in the flow paths denote rate controllers. The triangular symbol is a level duplicator, and the pentagonal symbol is a flow duplicator.

2.2. The Software Life Cycle Phase Structure

The software life cycle process treated here has one core unit activity for each of the following seven major phases in the process:

1. *system requirements definition and analysis
2. *system design and hardware/software allocation
3. software requirements analysis

4. software preliminary design
5. software detailed design, implementation, and test
6. *system integration and testing
7. *system maintenance

The four phases marked with asterisks (*) above are not exclusively software-oriented. Modeling of the activities in these phases is limited to the involvement of software personnel.

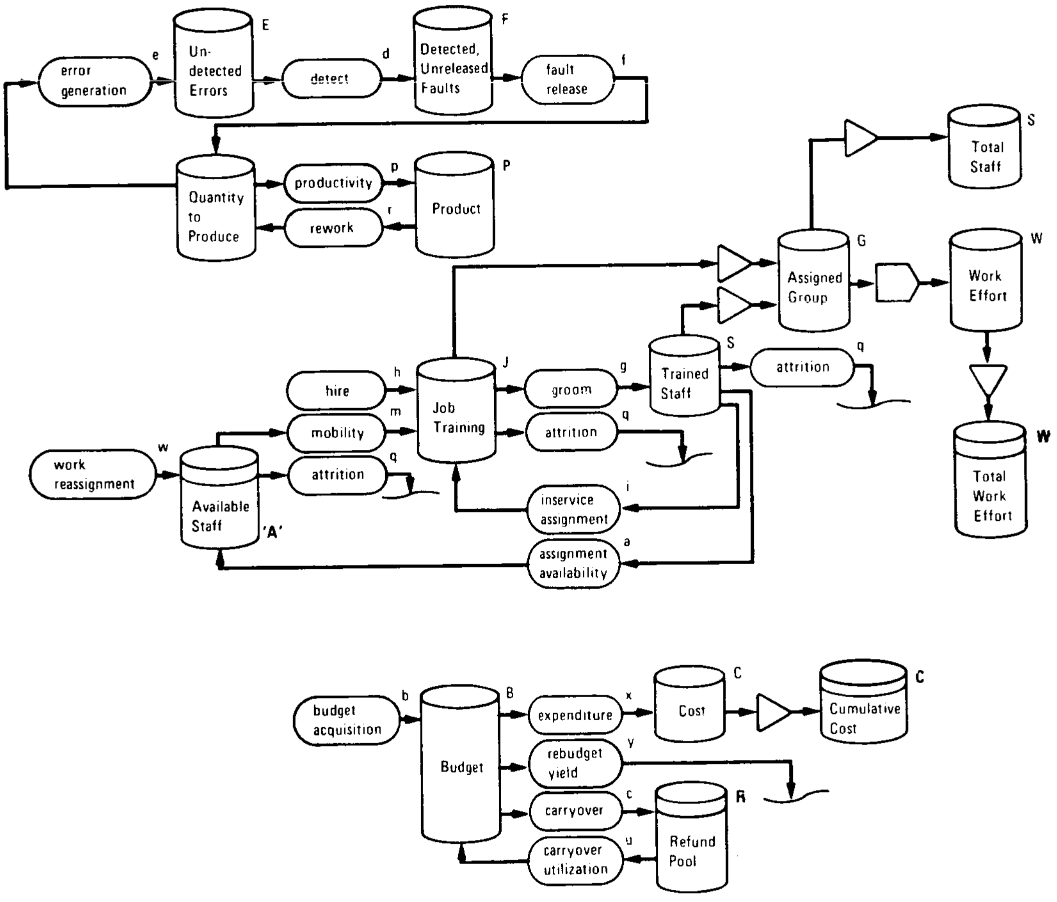

Fig. 1: Software Life Cycle Simulator Core Unit Structure

Each phase starts with a certain unknown volume of product that must be produced as precedent to the succeeding phases. A certain amount of the product will be produced correctly, some will be produced with as-yet-undetected faults in it, and some of the product will contain faults that have been discovered, but not yet repaired. Even portions produced correctly may have to be reworked when requirements change. The rates at which each of these portions of the product are generated (and, for errors, disposed of) are dynamic functions of both inherent and manageable parameters.

The production rate, or rate at which the quantity of product backlog is transformed into the finished product, is dependent, among other things, on the size and characteristics of the applied staff. For each phase, staff may be acquired from in-house resources or from the labor market. Each, upon entering into the new phase, may undergo a period of training (and a longer period of learning), perhaps administered by staff elements already on the job (inservice training), who take time out from their regular duties for this purpose. Staff may also be lost through attrition, or may be reassigned to activities in other phases or to other in-house tasks.

Staffing requires fiscal resources in order to exist. The allocation and acquisition of sufficient budget to sustain the staff is a prime requisite for doing work in a particular phase. On occasion, there may some funding for a phase left over after the phase product is complete that may be made available to another phase. In some cases, funds budgeted to a particular activity may have to be preempted to support another phase, or another project. In the latter case, the funding is lost.

2.3. Submodel Structure

The overall simulator is divided into four parts (Figure 2): an organizational performance submodel, a management submodel, a management influence interface, and a model analyst interface. The organizational performance submodel includes all of the core unit activities, and is described by a set of differential equations governing the levels and flows of products, personnel, and funding. Organizational performance is completely specified by the current state of the levels and the flowrate functions.

The management submodel contains a plan model, a visibility model, and an action model, each appropriately parametrized. Visibility into organizational performance is achieved via access to levels, flowrates, and flowrate parameters. Control is accomplished through manipulation of parameters within the management submodel interface.

The model analyst provides non-management performance parameter values that are inherent to the software process and to the performing organization, as derived from statistical or conceptual data.

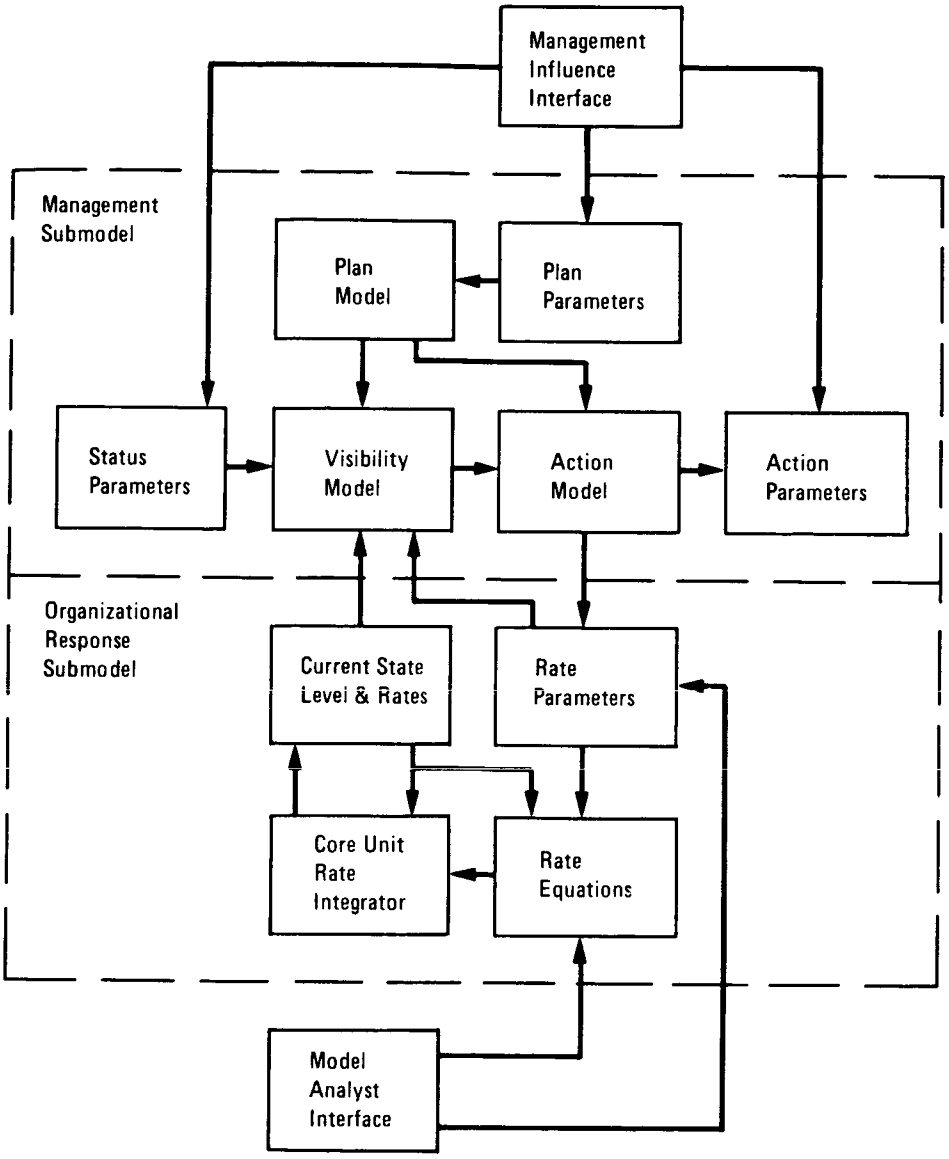

Fig. 2 : Software Life Cycle Simulator Submodel Structure

3. MODEL DESCRIPTION

3.1. Notation

In describing the equations governing the organizational performance model, the following notation is used:

Capital letters denote levels, lower-case letters denote flowrates and flowrate auxiliaries, and Greek characters denote flowrate parameters. Boldface capitals refer to cumulative levels across the entire submodel. Levels, rates, and parameters are subscripted by their phase indexes, 1 through 7, as above. The subscript ϕ refers to the phase under current consideration. When describing the relationships of quantities within a particular phase, this subscript is often suppressed.

Some parameters may have multiple subscripts, the first of which is always the phase number, perhaps suppressed. However, if one quantity in an equation describing the behavior of the model bears a phase subscript, then all parameters in the equation are subscripted by the appropriate phase. The phase number is never suppressed when all quantities in an equation do not refer to the same phase.

All quantities are, or potentially are, functions of time. The time dependency, however, is generally also suppressed for simplification of the formulas.

For each phase core unit, the levels are:

Q = Quantity of product yet to be produced
P = amount of Product produced so far
F = amount of Faulty product so far detected, this phase
E = product Error (fault) content, this phase
J = size of Job training staff pool
S = size of trained Staff
G = Group size, staff in this phase
$\mathbf{S}$ = total project Staff level
$\mathbf{A}$ = size of Available staff pool
W = Work effort, this phase
$\mathbf{W}$ = total Work effort, all phases
B = unspent Budget
C = Cost, so far this phase
$\mathbf{C}$ = total Cost, so far, all phases
$\mathbf{R}$ = carryover Refund pool

The flowrates among these levels are:

p = productivity
r = rework rate
d = fault detection rate
e = error generation rate
f = fault release rate
h = hire rate, labor market
w = worker reassignment rate
m = mobility of available staff
g = grooming (training) rate
i = inservice training assignment rate
q = quit (attrition) rate
a = available staff assignment availability rate
b = budget acquisition rate
x = budget expenditure rate
c = carryover rate
y = yank (budget reduction) rate
u = utilization rate of carryover pool

The function U(x) is the so-called 'unit-step' function

$$U(x) = 1 \quad \text{if } x = 0 \text{ or } x > 0$$
$$= 0 \quad \text{otherwise}$$

3.2. Normalization

All levels and rates defined above are normalized quantities with respect to amount of product, effort, budget, and time duration. In the following discussion, a prime symbol (') will be affixed to unnormalized quantities, and unprimed symbols denote normalized quantities.

Time is normalized such that unit time corresponds to a certain fraction of each simulated project, no matter how long the actual schedule is. It also will be convenient to normalize level values so that unit product, total effort, and total funding levels are used, regardless of project size.

The same time normalization is used for all core units, viz., the planned schedule time to enter phase 7. The product levels and product flow rates within each core unit are normalized individually by the actual (unknown) volume of product for that unit. The remaining levels and rates are normalized by overall planned values. Effort levels and rates are normalized by the planned expected project effort, while funding levels and rates are normalized using the planned project cost.

3.2.1. Parametric Time Normalization

Let t' denote actual (real) time values, and t denote parametric time, related by

$$t' = T\,t$$

where T is the specified time-normalization parameter. Note that when parametric time t = 1, then real time t' = T.

Let Z'(t') be a flow volume, or level, and Z(t), its corresponding normalized equivalent under the relationship

$$Z'(t') = Z_0\,Z(t)$$

with an appropriately defined time-independent parameter Z_0. When normalized such that $Z_{max} = 1$, the value of Z_0 becomes

$$Z_0 = Z'_{max}$$

For a flowrate z'(t'), the corresponding equation is

$$z'(t') = z_0\,z(t)$$

The volume of flow due to z' over a period of time will be finite, because the model deals with finite resources. Let this accumulated value be Z', and the corresponding normalized value be Z. Because of this time normalization factor, the values Z and Z' are related by

$$Z' = T\, z_0\, Z$$

or

$$z_0 = Z'/(Z\, T) = Z_0/T$$

where Z_0 is the level normalization factor. Time normalization, therefore, leads to T factors in the normalizing coefficients of flowrates.

3.2.2. Product Normalization

Let P_0 denote the actual (unknown) final amount of product to be produced in a particular phase. Then the relationship between P' and P is

$$P'(t') = P_0\, P(t)$$

in which P is confined to the interval [0, 1]. The flowrates influencing the product are

$$p'(t') = (P_0/T)\, p(t)$$

and

$$r'(t') = (P_0/T)\, r(t)$$

The model thus concerns itself with unit products in each phase, so that the actual sizes of each unit within the normalized organizational submodel are immaterial. Should the unknown final amount of product P_0 expand to P_1 due to, for example, an increase in requirements, then the submodel would view this as if P_1 were the normalizing factor, whereupon the normalized P would show a decrease by an amount $(P_1 - P_0)/P_1$.

3.2.3. Fault Normalization

Both detected and undetected product fault levels are normalized with respect to the unknown final product size,

$$\begin{aligned} E'(t') &= P_0\, E(t) \\ F'(t') &= P_0\, F(t) \\ d'(t') &= (P_0/T)\, d(t) \\ f'(t') &= (P_0/T)\, f(t) \end{aligned}$$

Therefore, E and F are measures of the relative error content in the product.

3.2.4. Staff Normalization

Let $\mathbf{W}_0$ be the real planned total effort for the project, and let W_0 be the planned effort for a core unit. Then

$$\mathbf{W}_0 = \mathbf{W}_{1,0} + \dots + \mathbf{W}_{7,0}$$

$$\mathbf{W}'(t') = \mathbf{W}_0 \, \mathbf{W}(t)$$

Let $\mathbf{S}_0$ be a staffing normalization value defined by

$$\mathbf{S}_0 = \mathbf{W}_0 \,/\, T$$

i. e., $\mathbf{S}_0$ is the average full-time-equivalent staff engaged in the project. The flowrate normalizations are

$$\begin{aligned}
h'(t') &= (\mathbf{S}_0/T)\ h(t)\\
m'(t') &= (\mathbf{S}_0/T)\ m(t)\\
q'(t') &= (\mathbf{S}_0/T)\ q(t)\\
g'(t') &= (\mathbf{S}_0/T)\ g(t)\\
i'(t') &= (\mathbf{S}_0/T)\ i(t)\\
a'(t') &= (\mathbf{S}_0/T)\ a(t)
\end{aligned}$$

and the levels are

$$\begin{aligned}
J'(t') &= \mathbf{S}_0\ J(t)\\
S'(t') &= \mathbf{S}_0\ S(t)\\
W'(t') &= \mathbf{W}_0\ W(t)
\end{aligned}$$

3.2.5. Budget Normalization

If $\mathbf{C}_0$ represents the real planned total cost of a project, and if C_0 is the cost allocation made to a particular phase, then

$$\mathbf{C}_0 = C_{1,0} + \dots + C_{7,0}$$

$$\mathbf{C}'(T') = \mathbf{C}_0\ \mathbf{C}(t)$$

The normalized cost rates are then

$$\begin{aligned}
b'(t') &= (\mathbf{C}_0/T)\ b(t)\\
x'(t') &= (\mathbf{C}_0/T)\ x(t)\\
y'(t') &= (\mathbf{C}_0/T)\ y(t)\\
c'(t') &= (\mathbf{C}_0/T)\ c(t)\\
u'(t') &= (\mathbf{C}_0/T)\ u(t)
\end{aligned}$$

and the levels are

$$\begin{aligned}
B'(t') &= \mathbf{C}_0\ B(t)\\
C'(t') &= \mathbf{C}_0\ C(t)\\
\mathbf{R}'(t') &= \mathbf{C}_0\ \mathbf{R}(t)
\end{aligned}$$

3.3. Approximation of Time Delays

The modeling of time delays in a process simulation model can be accomplished in one of two ways: all samples of the process may be stored in a queue during the delay time, or the samples may be put through a linear filter whose transfer characteristic approximates the desired delay. The former method requires a queue length equal to the process sampling rate times the delay time, and is appropriate whenever queue storage requirements are not extreme. The latter method suffers from amplitude and phase distortions when the degree of the delay filter is too small, but memory requirements are generally much more modest. Each of the methods may appear in the organizational performance submodel, as appropriate.

The linear filter transfer function of degree n corresponding to a 'maximally flat' unit-delay is given (in Laplace transform notation) by the equation [5]

$$D_n(s) = b_0 / (b_0 + b_1 s + \dots + b_n s^n)$$

where b_k, $k = 0, \dots, n$ represent the coefficients

$$b_k = \frac{(2n - k)!}{2^{n-k}\, k!\, (n - k)!}$$

The filters for n = 2 and n = 3 are, for example,

$$D_2(s) = \frac{3}{3 + 3s + s^2}$$

and

$$D_3(s) = \frac{15}{15 + 15s + 6s^2 + s^3}$$

The response of these filters may be expressed as nth-order linear differential equations. That is,

$$y(t) = D_n(s)\, x(t)$$

translates into the differential equation

$$b_n y^{(n)} + \dots + b_1 \dot{y} + b_0 y = b_0 x$$

For ease in computer solution, this equation is usually rewritten in state-vector form, in which y_k denotes the kth derivative of y(t), as

$$\dot{y}_{n-1} = [b_0x - (b_0y_0 + \ldots + b_{n-1}y_{n-1})] / b_n$$

$$\dot{y}_{n-2} = y_{n-1}$$

$$\ldots$$

$$\dot{y}_0 = y_1$$

In this way, the vector of derivatives, $(\dot{y}_0, \ldots, \dot{y}_{n-1})$ can be determined from the state vector $(y_0, \ldots, y_{n-1})$ at time t, and then numerically integrated to give the state vector values at time $t + \Delta t$.

For delay τ, rather than unit delay, it is merely necessary to replace each b_k above by $b_k' = b_k\tau^k$. For example, the 3<u>rd</u>-order maximally-flat τ-delay filter equations are

$$\dot{y}_0 = y_1$$

$$\dot{y}_1 = y_2$$

$$\dot{y}_2 = (15/\tau^3)\ (x - y_0 - \tau y_1 - 0.4\tau^2 y_2)$$

4. LEVEL EQUATIONS

4.1. Level Equation Form

The normalized equations of flow for each core unit may now be completely specified in terms of flowrates and levels. As above, the overdot in the equations below denotes time-differentiation,

$$\dot{Z} = dZ/dt$$

Since all levels in the model are non-negative quantities, the flow equation for each level necessarily takes the form

$$\dot{Z} = z\ U(Z)$$

so that a negative flow rate never produces a negative level. In the submodel level equations to follow, the step-function factor is omitted for simplicity.

The level equations that follow are mere mathematical restatements of the flow structure depicted in Figure 1.

4.2. Quantity of Product

Each phase deals with a unit product, which, at any particular time, may be composed of an as-yet-unproduced Quantity (Q), an amount having as-yet-undiscovered Errors (E), an amount having discovered, unrepaired Faults (F), and an amount of finished, correct Product (P).

$$\dot{Q} = r + f - p - e$$

$$\dot{P} = p - r$$

$$\dot{E} = e - d$$

$$F = 1 - Q - P - E$$

4.3. Job Training Staff Pool

The job-training level is composed of incoming untrained (new) staff, J_n, and trained staff, J_t, brought in for inservice training. The overall level is described by

$$\dot{J} = h + m + i - g - q_J$$

and the individual untrained and trained levels are

$$\dot{J}_n = h + m - (J_n/J)(g + a_J + q_J)$$

$$\dot{J}_t = i - (J_t/J)(g + a_J + q_J)$$

where a_J is the staff assignment availability (not shown in Figure 1) applicable to the job training pool.

4.4. Trained Staff

The trained Staff, S, consists of personnel dedicated to production and QA activities within the current phase.

$$\dot{S} = g - i - a_S - q_S$$

Note that the staff assignment availability, a, in Figure 1 is, in reality, made up of two parts, a_J and a_S, respectively applicable to the job training pool and the trained staff pool. The total group staff, G, is thus described by

$$\dot{G} = h + m - a - q$$

where $a = a_J + a_S$ and $q = q_J + q_S$.

4.5. Work Effort

The Work effort level, W, is the cumulative group staff time spent so far in the current phase.

$$\dot{W} = J + S = G$$

4.6. Project Work Effort

The project cumulative work effort, **W**, is the current value of total work effort in all phases.

$$\mathbf{W} = W_1 + \ldots + W_7$$

4.7. Available Staff

Available staff, **A**, denotes a pool of personnel resources not currently engaged in the project, but available to do so. Staff completing a given phase are reassigned to the available staff level before being reassigned, as appropriate, to other phases. Staff external to the project may also be added to the pool by work reassignment.

$$\dot{\mathbf{A}} = w + (a_1 - m_1 - q_{1,A}) + \ldots + (a_7 - m_7 - q_{7,A})$$

4.8. Budget

The phase budget, B, is the current value of remaining dollar resources allocated to the current phase.

$$\dot{B} = b + u - x - c - y$$

4.9. Cost

The phase cost, C, is the current value of the phase expenditure.

$$\dot{C} = x$$

4.10. Total Cost

The project total current cost, **C**, is the sum of all phase costs.

$$\mathbf{C} = C_1 + \ldots + C_7$$

4.11. Carryover Refund Pool

The carryover refund pool, $\mathbf{R}$, is the total of all funds given up by some phases and not yet utilized (obligated) to other phases.

$$\mathbf{R} = (c_1 - u_1) + \ldots + (c_7 - u_7)$$

5. RATE EQUATIONS

The formulation of the organizational performance model, as can be seen from the foregoing paragraphs, places the burden of achieving simulation accuracy in proper definition of rate equations and initial values for levels. This section parametrizes the flowrate quantities using simple, intuitive phenomenological models, as follows:

5.1. Productivity

The general form for the productivity, or production rate, equation is

$$p = p_0 \, p_J \, p_c \, p_q \, p_s \, p_1 \cdots p_n$$

where p_0 is a nominal time-independent productivity value for trained staff, p_J is an adjustment factor for staff in training, p_c is a compensation for communication overhead and other effects of overall project staff size, p_q adjusts for effort being used in error-detection and quality assurance (QA), p_s compensates for learning effects in the phase, and the other multipliers p_k are adjustments due to environmental, situational, organizational, experience, and other factors.

Each of the phase products is considered a separate, precedent milestone for doing correct work in the succeeding phases. Work may still be done without having 100% precedent in succeeding phases, but there will be a higher probability of making errors in that work that will later have to be corrected. No total overall product metric is defined.

5.1.1. Job Training Pool Effects

The effects of having a staff group G split between an untrained staff pool (J) and trained staff pool (S) is modeled by the productivity adjustment factor

$$p_J = S + \pi_J J$$

where π_J is a productivity ratio value for staff in the job-training pool (J), including personnel doing the training. It principally reflects the effects of time spent in training activities rather than in production. Learning-curve effects are treated separately, below. The parameter π_J is supplied by the management submodel.

5.1.2. Communications Overhead

One productivity adjustment is due to organizational communications overhead. This overhead is simulated using an overhead model [6] that postulates that the time increment spent in overhead activities is proportional to the staff increment and the non-overhead time remaining. The productivity adjustment factor in this case is given by

$$p_c = \exp[-(S - 1/S_0)\gamma_c]$$

for $S > 1/S_0$, and $p_c = 1$ otherwise.

The parameter γ_c is the communications relative time factor, supplied by the model analyst.

5.1.3. Staffing and Learning Curve Effects

It is generally accepted that productivity of personnel increases due to several kinds of learning, among which are general experience, organizational experience, and specific task familiarity. Each of these productivity effects is commonly described dynamically by a first-order linear differential equation having a 'learning time-constant' parameter.

The organizational response submodel approximates only the task familiarization effects, and relegates the other, longer-term experience adjustments to other p_k factors. Thus, the total staff productivity due to size and state of familiarity is taken to be the form,

$$p_s = \pi_0 + \pi_s \; / \; G$$

where π_s satisfies the equation

$$\tau_s\dot{\pi}_s + \pi_s = (1 - \pi_0)\; G$$

in which τ_s is the learning time-constant, G is the staffing function to which that kind of learning applies, and π_0 represents the untrained-staff/trained-staff productivity ratio. The trained-staff productivity is thus normalized to unity.

The value of p_s is the learning-state productivity adjustment for the staff group G as a function of time. If the staff G were applied all at once, p_s would rise from π_0 asymptotically to 1. However, since the staffing plan may not be a step-function, the differential equation form is used.

The learning time parameter τ_s is a function of the teacher/student ratio, ρ, and is longest when staff members learn on their own (i. e., at $\tau_{s,0}$, when $\rho = 0$).

The variation in learning time may be approximated by a cubic form

$$\tau_s = \tau_{s,0} + \tau_s' \rho (1 - \rho)^2 + \Delta\tau_s \rho^2 (2\rho - 3)$$

This particular form takes on the self-taught time value at $\rho = 0$, is minimum when the teacher/student ratio is unity, being reduced by a (positive) increment $\Delta\tau_s$ at this point, and has (negative) slope τ_s' at the origin. For $\rho > 1$, it was assumed that there would be too many teachers per student, so that the training time would actually take longer than 1-on-1 training. The form is subject to the restriction

$$\Delta\tau_s > - \tau_s' / 2 > 0$$

The parameters π_0, $\tau_{s,0}$, τ_s', and $\Delta\tau_s$ all are supplied by the model analyst interface.

5.1.4. Effect of Quality Assurance

If ξ_q represents the fraction of effort devoted to quality assurance (QA) pursuits (i. e., the 'extent' of QA) to discover errors or otherwise improve the quality of product, there is a corresponding reduction in productivity due to the reduced effort. Nevertheless, the overall correct-product rate may be improved, because faults may be discovered before they propigate into other phases.

The productivity adjustment factor for QA activity is thus of the linear type,

$$p_q = 1 - \xi_q$$

The QA fraction emanates from the management submodel.

5.1.5. Linear Extent Factors

Other productivity adjustment factors may take the linear form exhibited above,

$$\begin{aligned} p_k &= 1 + \pi_k \xi_k \quad \text{for } \pi_k \xi_k > -1 \\ &= 0 \qquad\qquad \text{otherwise} \end{aligned}$$

The extent factors ξ_k range in the intervals [0, 1] or [-1, 1]. In either case, the productivity adjustment factor ranges from its least to most beneficial value as ξ_k varies from the lower to the upper limit. In the former case, π_k is the total swing in productivity adjustment,

$$\pi_k = p_{k(max)} - p_{k(min)}$$

while in the latter case, it is only half this amount.

The π_k parameters are supplied by the model analyst for projects in general, while the ξ_k are supplied by the model user to relate the extent to which each factor under consideration is present in the project to be simulated.

5.1.6. Exponential Extent Factors

Several productivity adjustment factors take the form

$$p_k = (\pi_k)^{\xi}$$

where $\pi_k > 1$ is the maximum beneficial effect of project factor k, and $\xi = \xi_k$ is a value in the interval [-1, +1] that registers the extent to which factor k is present in the current project. When $\xi_k = -1$, there is minimum benefit of factor k, so the value π_k is seen to be the square of the max/min productivity ratio,

$$\pi_k^2 = p_{k(max)} \; / \; p_{k(min)}$$

The π_k parameters are supplied by the model analyst for projects in general, while the ξ_k is supplied by the model user to relate the extent to which the factor under consideration is present in the project to be simulated.

5.2. Error Generation Rate

The rate that undetected errors are introduced into the product of a given phase is assumed to be proportional to the rate at which the product is being produced. That is, all other things being equal, the error content per unit of product would be the same. Also, the error content is assumed to decrease as the staff comes up on the learning curve, by an amount proportionate to the staff's increase in productivity. Additionally, the error rate depends on the amount of products in precedent phases not yet produced, or produced in error. For example, if the software requirements generated as the product of phase 3 are incomplete or in error, yet phase 5 insists on doing implementation, then there will be a higher likelihood of work being erroneously done in phase 5. Parametrically, the error generation rate takes the form

$$e = (p \; / \; p_s) \; [\varepsilon_0 + (Q_1 + F_1)\varepsilon_1 + \dots + (Q_{\emptyset-1} + F_{\emptyset-1})\varepsilon_{\emptyset-1}$$
$$+ E_1\eta_1 + \dots + E_7\eta_7]$$

The quantity ε_0 represents the relative volume of errors that would be introduced in the current phase, even if all precedent work were completed. Each ε_k and η_k reflects an increase in error generation rate in the current phase due to incompleted products $(Q_k + F_k)$ and errors (E_k) of precedent phases. The contribution to error generation due to incompleted products is termed 'speculative error'. The error generation due to precedent errors is 'compounded error'.

This model of error creation presumes that the magnitudes of the product levels and detected fault levels of precedent phases are immediately transmitted to the current phase. While this may not be generally true, preliminary results are often made available at regular intervals. If the project behavior is sensitive to this assumption, each of the Q_k, F_k, and E_k in the error rate equation above may be delayed by a parametric amount, e. g.,

$$Q_k = Q_k(t - \tau_k)$$

The values ε_k and η_k are supplied to the organizational performance submodel by the model analyst, and τ_k comes from the management submodel.

5.3. Fault Detection Rate

The rate at which faults are detected is assumed to be a function of the productivity of the effort devoted to finding errors (QA), the number of errors yet undetected, and the detectability of those errors. The following linear form is postulated:

$$d_\emptyset = E_\emptyset \{p_\emptyset \delta_{\emptyset,\emptyset} [\xi_{\emptyset,q} / (1-\xi_{\emptyset,q})] + \\ \ldots + p_7 \delta_{\emptyset,7} [\xi_{7,q} / (1-\xi_{7,q})]\}$$

The δ_k are related to the ease with which a (later) phase k detects an error created in the current phase, and are supplied to the organizational performance submodel by the model analyst.

5.4. Fault Release Rate

The rate at which work detected to be faulty is released back to the product backlog queue depends heavily on management policy and decision. In some instances, work is immediately released to be corrected. In other cases, work may be held for correction in a later software version update. Therefore, the release function, f, is supplied to the organizational performance submodel via the management submodel.

5.5. Rework Rate

Rework here is the process of returning portions of a phase product back to the product backlog queue. Such action is taken when improvements to a phase's products are in order, or when requirements change and a revision is necessary. Both of these situations are management driven, and thus the rework rate function, r, is supplied via the management submodel interface.

5.6. Outside Hire Rate

The hire rate from the external labor pool, h, will also be designated as a function supplied by the management submodel, since the strategy governing hires is a management prerogative.

5.7. Work Reassignment Rate

The rate at which personnel from outside the project are made available for use in the project is controlled by management. Hence, the work reassignment rate, w, emanates from the management submodel.

5.8. Staff Mobility Rate

As with outside hire rate, the mobility strategy, m, or use of available staff, is provided within the management interface.

5.9. Staff Attrition Rate

Staff attrition rates within the job-training pool, the trained staff pool, and available staff pool are assumed to all be the same proportion of the staff levels involved:

$$q_J = q_0 J$$

$$q_S = q_0 S$$

$$q_A = q_0 A$$

The attrition coefficients, q_0, are supplied by the management submodel.

5.10. Staff Inservice Training

The assignment of trained staff to perform inservice training is a management action, whose purpose is to shorten the training period for incoming untrained staff. The assignment is assumed to depend on the number to be trained, and the number available to train them. An inservice assignment rate of

$$i = (\rho J_n - J_t) / \tau_i$$

will bring (asymptotically) a trained-to-untrained personnel ratio (i.e., a teacher/student ratio) of $\rho = J_t / J_n$ into the training pool. The time-constant τ_i reflects the time required to break trained staff free and bring them into the training activity. Both the teacher/student ratio, ρ, and the time constant τ_i are defined within the management submodel.

5.11. Staff Grooming Rate

Personnel receiving job-training from others are assumed to spend a fixed, finite time at their studies, or in classrooms. Thereafter, they commence activities as trained staff. If τ_g represents the nominal time spent in such activities, then the 'grooming' rate is

$$g(t) = h(t - \tau_g) + m(t - \tau_g) + i(t - \tau_g) - q_0 J(t - \tau_g)$$

That is, the transfer of personnel from training activities into productive status is modeled as a mere time delay imposed on the inflow into the training pool. The training time parameter τ_g is defined by the management submodel.

5.12. Staff Assignment Availability

Staff assignment is a management prerogative, so the assignment availability functions, aJ and a_S, are provided by the management submodel.

5.13. Budget Acquisition Rate

Funding acquisition, distribution among phases, and time of activity initiation are management functions, so the budget acquisition rate, b, controlling these characteristics, is provided by the management submodel.

5.14. Budget Expenditure Rate

Budget expenditures in a given phase are primarily due to two factors: personnel and product costs.

$$x = \omega_G G + \omega_p (p + e)$$

The parameter ω_G is the average burdened wage of the group staff, and ω_p is the non-personnel-related production cost per unit product. The latter term includes costs to create errors in the product, as well as to generate correct product. In all phases, ω_p includes documentation. During the implementation and testing phases, it also includes computer utilization and other support costs. The management submodel supplies both ω_G and ω_p.

5.15. Budget Reduction Rate

Projects may, at times, increase or decrease a task's funding allotment. A decrease that is lost to the project is termed the 'budget reduction,' or 'yank' rate. The amounts of reductions and conditions for initiating such events are not part of the organizational response, although the effects within the project caused by such reductions are. The yank function, y, is, therefore, supplied by the management submodel.

5.16. Budget Carryover Rate

Budget carryover, or the release of funding from one phase for use elsewhere within the project, is a management prerogative. The carryover function, c, therefore emanates from the management submodel.

5.17. Budget Carryover Utilization Rate

Reutilization of carryover (and contingency) funds is a management control, and, therefore, the reutilization strategy, u, is supplied by the management submodel.