A Discrete, Deterministic Model for Understanding Software Project Development Contingency Profiles

Bruce R. Barkstrom
15 Josie Lane
Asheville, NC 28804, USA

and

Paula L. Sidell
119 Sandpiper Lane
League City, TX, USA

ABSTRACT

This paper describes a simple, discrete deterministic model for the allocation of project contingency as a function of project phase. When a project starts, there is always some uncertainty regarding the resources it will require. As the project proceeds, the participants discover this uncertainty and remove it through the expenditure of resources. In addition, the project may encounter unexpected changes in its environment that force the project to change. The model assumes that the project will eventually discover a (relatively small) number of contingencies, each of which requires the same workforce and duration to resolve. Because the time of appearance and resolution time for each contingency is independent of other contingencies, there may be several overlapping contingencies the project will work on at the same time. Because the model assumes that contingencies suddenly add and subtract staff, managing staff becomes much more complex than is the case with a more uniform staffing plan. In particular, if there are appreciable hiring and firing costs, a project may find it desirable to simply retain staff that can be assigned to contingencies, even though there may be periods with no contingency work.

Keywords: Project Contingency, Management Style

1. INTRODUCTION

Projects everywhere feel the effects of the stringency with which managers must adhere to initially negotiated spending plans and schedules. If a project were never to experience unexpected changes, this could be a reasonable policy. However, projects rarely have enough initial knowledge to be able to provide a deterministic budget and schedule. For software projects, the number of source lines of code is usually not known exactly. The Cost Estimation Relations (CERs) used to estimate costs have their own uncertainty. In addition, the project's environment is subject to perturbations in its plans because that environment evolves. Sometimes, the project's funders or users change their minds about requirements. Technology the project planners assumed would be available isn't or completely new technology is so much better that a project is forced to accept it.

In this paper, we explore a rather simple model for the project's contingency budget (or "management reserve"). The model assumes that contingencies appear suddenly with an average time interval. Each contingency requires a fixed workforce [person-months] to resolve. Thus, the project assigns staff to work on a contingency until it is resolved. While real contingencies appear at non-deterministic time intervals and require variable staff levels and completion times to resolve, this simple model provides a useful start at thinking about how to manage these unexpected changes. After describing out simple model, we provide some examples of the phenomena using numerical calculations. We are particularly interested in how to estimate the work required to monitor and control contingency expenditures. In the final section of the paper, we suggest some strategies for dealing with contingencies.

2. THE MODEL

Initial Sources of Project Uncertainty

In this paper, we concentrate on software development projects. Cost and schedule estimation for such projects usually uses an estimate of the number of "source lines of code" (SLOC) as the fundamental parameters entering the Cost Estimation Relationships (CERs) [1, 2]. There has been some dissatisfaction with this basic parameterization [3, 4], but for our purposes, SLOC is a satisfactory basis.

One source of uncertainty in the estimated cost and schedule lies in the input values for SLOC. For this paper, we assume that the project can provide a lower bound, $SLOC_L$, and upper bound, $SLOC_U$.

The second source of uncertainty lies in the CERs that provide a recipe for calculating the required project workforce, *MMDEV*, in [person-months], and development time, *TDEV* in [months]. [1] and [2] suggest

$$MMDEV = A\,(SLOC)^B;\;\; TDEV = C\,(MMDEV)^D$$

where *A*, *B*, *C*, and *D* are parameters derived by taking a large database of projects with recorded values of *SLOC*, *MMDEV*, and *TDEV* and then running statistical regressions between the independent variables and the dependent ones. Naturally, the data do not fall exactly on the regression surface. This means that the CER has uncertainty above and beyond the uncertainty in the project's input.

Although we might be tempted to assume a Gaussian probability distribution for SLOC, it is difficult to find the calculated standard deviation for the CERs. Accordingly, we

use Boehm's COCOMO II [2] regression coefficients and estimate the maximum uncertainty by using the twelve regression coefficients for effort quoted in Table 6-6 on page 86 of [1] and the seven regression coefficients for development time in Table 6-7 on page 89 of that reference.

In the numerical examples that illustrate the model's calculations, we take $SLOC_L$ = 80,000 and $SLOC_U$ = 100,000. The resulting minimum *MMDEV* is about 124 person-months and the minimum *TDEV* is about 11 months. The equivalent maximum *MMDEV* is 904 person-months and the maximum *TDEV* is about 35 months. We could use the range in these values as representing an "objective" estimate of the maximum uncertainty in project workforce and development schedule.

This range of workforce and development time is rather large compared with the uncertainty most project managers would be willing to quote. Accordingly, model users should provide their own estimate of the appropriate range of workforce contingency, $\Delta MMDEV$. This quantity represents the initial uncertainty in the planned work at the start of the project. It does not include changes in requirements or in the project environment. For our numerical examples, we take $\Delta MMDEV$ = 100 Person-Months, representing a development workforce uncertainty of about 30%.

In addition to providing an estimate of uncertainty, we use the CERs to estimate the *basement workforce*, $MMDEV_B$ and *basement development time*, $TDEV_B$. These are the minimum workforce and minimum time for the project. We regard these as values such that acceptance of a lower workforce or shorter development time guarantees project failure. Based on these two values, the model gives us an estimate of the appropriate staffing level as

$$S_B = MMDEV_B / TDEV_B$$

For the numerical values we have presented, the basement staff level, S_B, for the standard example is about 12.

Contingency Discoveries From Initial Uncertainties

It would be an unusual project in which the uncertainty associated with the initial estimates of workforce and schedule would be uniformly dispersed over the entire development duration. Most projects with which the authors are familiar experience the work required to remove this initial uncertainty as coming in discrete contingencies. Perhaps the most visible sign of this discrete character comes when the project feels compelled to form a "tiger team" to deal with a problem. Regardless of contingency size, "action items" that arise from recognition of an initial uncertainty should be labeled so that they can be resolved or worked off.

To keep the model as simple as possible, we assume that the project will encounter a finite number, N_0, of discrete contingencies. We assume that each contingency will require ΔW of effort to resolve. This means that the expected number of contingencies will be

$$N_0 = \Delta MMDEV / \Delta W$$

For our standard numerical examples, we assume that ΔW = 20 Person-Months. This means that the example project will encounter 5 contingencies over its development period.

Contingencies are not discovered on schedule. Rather, they arise as the project exerts development effort. If we were dealing with a continuum of contingencies, we might use a differential model in which the number of old contingencies, ΔN, discovered in a time period Δt is proportional to the number of undiscovered contingencies remaining at time t, $N(t)$, times a rate that is proportional to the staffing level for the basement level of effort and a "contingency discovery efficiency", r. In the form of an equation

$$\Delta N = - r S_B N(t) \Delta t$$

When the time interval is taken to the limit of 0, the solution to the resulting differential equation is

$$N(t) = N_0 \exp(- r S_B t)$$

In other words, if we assume that the initial uncertainty is removed when it appears, the number of remaining contingencies declines exponentially with time at a rate determined by the time constant $\tau_D = 1/(r S_B)$. We might call τ_D the time scale for contingency discovery. In our numerical examples, we will take r = 0.1 Contingencies per Person-Month. With a basement staff level of 11.9 persons, τ_D = 2.73 Months.

This analytic result suggests that we can provide a "deterministic" start time for N'th contingency:

$$t_N = \tau_D \ln[N / (N_0 + 1 - N)]; \qquad N = 1, 2, \ldots N_0$$

which keeps the basic exponential form of the continuous approximation, but avoids division by zero for the final contingency.

When there are only a "smallish" number of "largish" discrete contingencies, the continuous model is inappropriate. However, we still expect the project to discover fewer contingencies per unit time interval as time goes on. Basically, unexpected items become harder to find. We can approximate this effect in a completely deterministic model by allowing the time between contingency discoveries to expand by a constant factor. For numerical examples, we take this factor to be 1.2. This means that contingency discoveries in our basic numerical example will occur at 2.7 months, 6.0 months, 9.9 months, 14.6 months, and 20.3 months.

Treatment of Contingency Resolution Work Off

We have already identified the workforce required to work off a contingency, ΔW. In order to complete the model, we need to specify an expected duration, τ_C, of the work on each contingency. The staff assigned to each contingency will clearly be

$$S_C = \Delta W / \tau_C$$

For the numerical examples, we take τ_C = 6 months (or about 180 days). With ΔW = 20 Person-Months, we find S_C = 3.4 persons.

Summary of the Contingency Workforce Model

The model we have described is quite simple. The user first finds the expected project workforce and development time. Then, he or she estimates the uncertainty in the workforce.

Next, the user inputs the workforce required to work off each contingency. This input allows the model to calculate the likely number of contingencies that will be found as a result of the initial uncertainty. It also provides the staffing level for the "basement" level of effort over the project duration.

After these inputs, the user needs to input the contingency discovery efficiency and the expected workforce required to work off each contingency. Based on this input, as well as the factor by which the time interval between contingency discoveries expands and the average duration for working off each contingency, the model provides the contingency staffing level. At the end of the project, all of the initial contingencies should be worked off, ensuring that the model's apportionment of contingencies conserves the estimated total as input.

3. EXAMPLE NUMERICAL RESULTS

Introducing the Numerical Results

There are two primary kinds of results we can create from this type of simulation:

1) Time histories of project staffing for the planned work and for the contingency work
2) Time histories of project expended workforce for the planned work and for the contingencies

In the subsections that follow, we use the previous numerical values to create plots of each kind of time history. The most immediate phenomenon is the discrete jumps in staffing as the project discovers and resolves contingencies. If the manager could hire and fire staff instantaneously and without cost, the actual project staffing could follow these jagged time histories. However, in the real world, hiring and firing are expensive. In addition, staff need training after being hired. We will not treat these complications here, but will consider a simple metric for evaluating the efficiency of managing contingency work.

Time History of Project Staffing

Figure 1 shows the staffing profile for both the planned work and the initial contingenciesas well as a Gantt chart of the planned work and the five contingencies. Each major "bump" in the contingency staffing occurs when work starts to resolve a particular contingency. Each contingency's completion produces a "cliff" as the staffing reduces when the contingency is resolved. In the Gantt chart, the planned work appears as the green bar; the five contingencies as red rectangles. The increasing interval between internal contingencies is visible, in accord with the logarithmic model above.

Because the discovery times and the completion times are incommensurate with each other, the project clearly is not able to keep a constant staffing level fully occupied. To put it another way, there will be times when a staff hired to deal with contingencies has no work to do. This contingency-induced staffing variability creates a "managerial challenge." If the project manager cannot offload staff handling contingencies when they are not dealing with this unplanned work, there are two dangers. On the one hand, the manager may need so many staff that he or she exceeds the budget and is fired. On the other hand, if he or she is able to arrange the contingency budget so there is enough staff to cover all the contingencies, that staff will not be "completely efficient."

Time History of Expended Workforce

Based on the staffing profiles for the planned work and for the

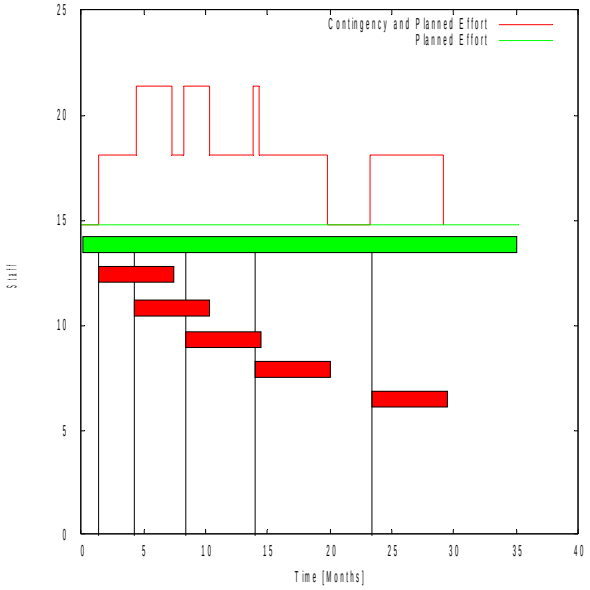

Figure 1. Staffing Profile and Corresponding Gantt Chart for Planned Work and Internally Generated Contingencies.

contingencies, it is straightforward to calculate the expended workforce as a function of time, since that is the integral of staffing over time. Figure 2 shows the expended effort for the planned work (lowest line) and for the contingencies (line with slope breaks).

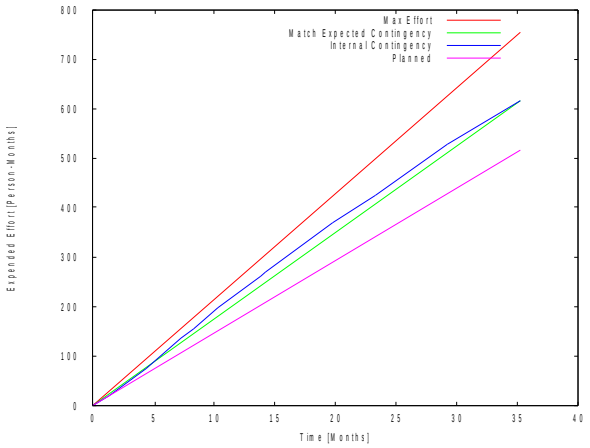

Figure 2. Expended Workforce for Planned Work (lowest line), for Internally Generated Contingencies (line with slope changes), and for a lower and upper bound on workforce.

In addition to the calculated values for the time history of expended effort, fig. 2 shows an upper bound for the time history of expended effort assuming the project hires about 18.7 people at the project start and keeps them until theproject ends. The total budget would need to accommodate about 657 person-months covering the cost of contingencies due to initial uncertainty. The project manager would need to cover 238 Person-Months, although the total workforce for contingency is only 100 Person-Months. This means that the budget including both planned work and contingencies would have to be about 58% larger than one with just planned work. On the other

hand, if the project manager could create a budget with exactly the correct coverage in this scenario, he would only need to cover about 520 Person-Months, only 24% more than the 420 Person-Months in the planned budget.

The green line in fig. 2 shows expended workforce assuming a constant contingency staffing. The manager could hire about 2.8 persons to handle contingencies and keep them on the project from start to finish. As fig. 2 shows, during the first couple of months there are no contingencies. After six months, the contingency load has increased enough so that the project needs more staff to resolve the unexpected work. As a result, the contingency workforce starts out costing more than the straight line budget. By design, this budget would just cover the contingency expenditure by the end of the project, but the timing of the budget is off. There is more need for contingency staff in the middle part of the project. Usually this additional attention also increases the work the project staff must do, leading to the "regenerative schedule disaster" described in [5].

4. INCORPORATING EXTERNALLY CREATED CONTINGENCIES

Why External Sources Create Contingencies

There is no doubt that projects experience changes in their environment that force them to work on activities that were not included in the original, planned work. In the early 1990's, the NASA Earth Observing System Data and Information System (EOSDIS) project planning assumed that users would need a Motif graphical user interface to access and use NASA's Earth science data. When the Internet era began in 1994, the project needed to work on the early version of what is now the familiar Web environment. The development of Web 2.0, the adoption of Geographic Information Systems with Web Mapping Services, and the development of high-powered Semantic Web software provide additional examples of changes that arise external to a software development project. Project funders and other stakeholders may also force a project to do unplanned work. For example, government projects may have new security or reporting requirements forced upon them by agency managers.

Further evidence of the force of such changes appears in the maintenance costs of software projects. In [1, p. 71], such changes are characterized by the *Annual Change Traffic* (ACT), which is "the fraction of the software product's source instructions which undergo change during a typical year, either through addition or modification." Typical values for ACT lie between 8% to 15%. This kind of change also underlies the customer cost of software product maintenance. [6] notes that these costs are currently running between 17% and 25% for such large scale products as enterprise databases or business management software.

Formulating a Model for Externally Created Contingencies

If we were dealing with a continuous stream of project contingencies, we could modify the original equation for discovering contingencies to incorporate a source term

$$\Delta N = - r S_B N(t) \Delta t + Source(t) \Delta t$$

The question then is "what is an appropriate model for the time dependence of externally caused contingencies?"

If there were a known change in the external environment that arose when the project was planning, that would already be taken into account in the project planning process. A simple hypothesis is that the number of contingencies arising from external causes should be 0 when the project starts. At the end of the project's development phase, the rate at which contingencies appear should settle in to a relatively constant value. To be consistent with the ACT model, the number of changes would be proportional to the total development effort. The simplest assumption that satisfies these two end conditions is a source term that is proportional to the project's expended effort. For the simple "level of effort" model we have used in sections 2 and 3, this means that the contingency source term expands linearly with time. When a project starts, it has no product to change. The further along it goes, the more produced items it has that might need to be changed.

Quantitatively,

$$Source(t) = r_{Ext} S_B t$$

This quantification means that the number of externally forced contingencies will increase quadratically with time. In other words, we expect

$$N_E(t) = \tfrac{1}{2} r_{Ext} S_B t^2$$

where $N_E(t)$ is the number of externally forced contingencies disclosed by time t. r_{Ext} is the *external contingency generation rate*.

While it would be more realistic to create a statistical model for the appearance of these externally-driven contingencies, this paper uses a "deterministic" one. At the end of planned project development, when $t = TDEV_B$, the number of these contingencies will be $N_{End} = N_E(TDEV_B)$. It follows that

$$t_N = TDEV_B \sqrt{(N / N_{End})}$$

where t_N is the time at which the N ' th external contingency appears. This function "end loads" the appearance of externally-driven contingencies. Thus, if we want the appearance time for the contingency that is halfway through the full number, that will be at about 70% of the planned development duration, rather than 50%.

We also assume that resolving externally created contingencies requires as much workforce as it does to resolve the contingencies created by the initial uncertainty in planning.

A Numerical Example with External Contingencies

Figure 3 shows the planned and contingency staffing levels. The figure includes both the contingencies generated by the initial uncertainties (as the green line) and those generated by the externally generated ones (as the red one). In this example, the work on the final contingency starts before the end of the planned development work, but continues for about two months after the nominal completion. The original contingencies that arise from the initial uncertainties in the project's plan appear in green. The contingencies that arise from the externally-generated contingencies appear in green. The original, planned staffing appears in blue.

Because the externally-generated contingencies appear based on the amount of completed work, the square root dependence "bunches" the work required to resolve them nearer to the end

of the project.

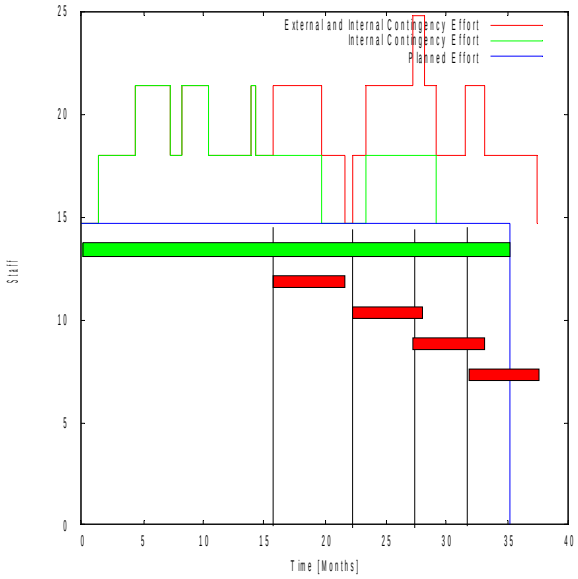

Figure 3. Staffing Levels and Corresponding Gantt Chart for Planned Activities and External Contingencies.

Limiting Form of Contingency Staffing

The algorithm for computing the contingency staffing effort works even if we let the number of internal or external contingencies become very large. Figure 4 shows the planned work, the additional staff required to handle 500 internal contingencies, 400 external contingencies, and the combined contingency work.

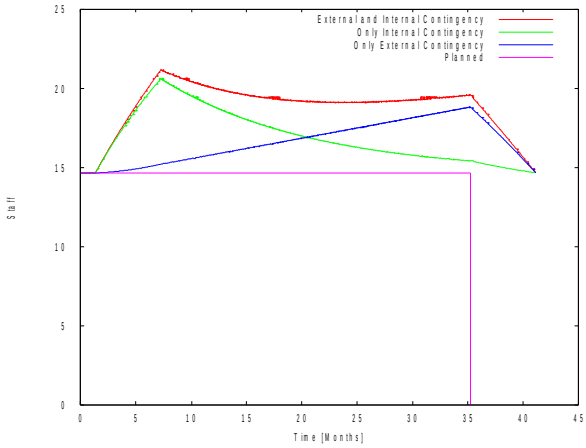

Figure 4. Limiting Form of Staffing Levels for Planned Work and Contingencies.

When the number of contingencies becomes large, we can keep the total contingency effort finite by decreasing the workforce required to close each contingency even the time to close each contingency remains constant. The contingencies generated by the initial project uncertainty increase as $1 - \exp(- r S_B t)$. This factor produces the rise on the left side of the green curve in fig. 4. Thus, the initial discovery of contingencies accumulates work to be done. As the contingencies are worked off, the work in progress declines exponentially with a time constant τ_C. In other words, the internal contingencies act like a queue that is filled by discovery of problems and emptied by the work that resolves them.

On the other hand, externally generated contingencies appear as problems are discovered in the planned work that is already completed. By assumption, the completed work is the result of a "level of effort" expenditure, which leads to a linear increase in the backlog of unresolved problems. As these problems are resolved, they are removed from the external stack. Where work starts before the end of the planned work, but is not finished until after it was supposed to be done, the backlog declines linearly with time after the planned end of development. In normal circumstances, this work would be regarded as development "maintenance."

5. IMPLICATIONS AND RECOMMENDATIONS

Summary of the Hypothesis of This Work

In this paper, we assume that uncertainty is always present in the budget and schedule of a project. There are two different sources of uncertainty:

1) Uncertainty from the initial assumptions in the project's schedule and budget
2) Uncertainty from changes in the project's environment as the work progresses

For both sources, the project must find staff to resolve the discovered contingencies. In reality, the required workforce, duration, and staffing are stochastic; they are not predictable in advance. In this paper, we simplify the stochastic process of discovering contingencies by a deterministic algorithm that bunches the discoveries of contingencies arising from initial uncertainty near the beginning of the project, while those arising from changes in the project's environment are bunched near the end.

Contingency Management Strategies

In the case of initial uncertainty, one reduction strategy is to try to remove development novelty. This strategy is at the root of such approaches as the Capability Maturity Model [7]. This kind of approach systematizes software development processes, emphasizing development of a reliable statistical basis for making estimations of the cost and schedule of projects. In project accounting, we would include Earned Value Analysis (EVA) [8] as a tool designed to reduce project uncertainty arising from initial assumptions. EVA emphasizes identifying concrete measures of progress, as well as careful monitoring of the project's progress against these measures.

Where uncertainty cannot be avoided, experts recommend moving to risk management. In simple forms, this strategy attempts to categorize risks, i.e. sources of uncertainty, based on the kind of threat to the schedule or budget, the probability of the threat occurring, and the probable cost of an increase in budget or delay [9]. In more complex forms, this approach leads to probabilistic risk analysis.

From the perspective of the model presented in this paper, a key strategic decision for the project manager lies in matching the development lifecycle to the ratio between the total budget uncertainty and the budget for planned work. If that ratio is low, the project has only a small amount of uncertainty – and it might adopt a waterfall lifecycle. This approach is likely to work best on projects that contain almost no novelty – and for which the project organization already has a substantial track record. This approach emphasizes having the project spend a significant amount of effort on ensuring clear requirements that

will not be changed after they are developed.

Projects with a moderate amount of novelty (and those undertaken within a stable IT environment) might be better served by a spiral development approach [10]. This strategy seeks to reduce uncertainty bystarting with the portions of the development that have the largest risk.

When the initial project analysis suggests that the contingency effort is likely to be large in comparison with the planned workforce, the distinction between plannable work and contingency work probably doesn't make sense. In this case, the project manager should probably adopt an agile lifecycle [11]. In essence, this lifecycle emphasizes selection of project activities to gain experience with the real problem. From the perspective of this paper, agile methods concentrate on "buying down" the uncertainty as rapidly as possible.

In this paper's model, there are two parameters that a manager may influence: the diligence of initial uncertainty search effort (as reflected in the mean time between discoveries, τ_D) and the speed with which contingencies are resolved (as reflected in τ_C). If the manager succeeds in reducing τ_D, then the initial uncertainties will be discovered more rapidly. If the manager succeeds in reducing τ_C , then the pool of active contingency work will be reduced. Rapid contingency resolution would appear to be the more effective strategy, although that needs to be confirmed with more extensive data than we have been able to find.

We also note that this paper does not allow the manager to choose between reducing the planned work in return for dealing with newly discovered contingencies. In practice, such choices are an intrinsic part of the manager's job. Our work in this paper also does not deal with the overhead arising from project control mechanisms. It should be clear that if a project's funders introduce more stringent controls than those originally planned, then the overhead of that new work acts as a contingency created by a change in the environment. The cost of this increased effort will have to be paid for – whether by slowing the planned work, increasing the contingency reserve, reducing the scope of the project, or by incurring the cost of project termination.

Implications of the "Lumpy" Nature of Contingency Work

It should be clear that project managers must deal with the fact that only very large organizations can suddenly change personnel assignments. If an organization is large enough to have individuals with pecialized skills, it may also be able to assign these individuals to solve problems without a "hiring and firing" penalty or significant training. Smaller organizations may not be able to afford to keep stables of specialists.

From a strategic point of view, a project manager can adopt two approaches.

1) First, he or she could budget for a pool of resources that are available on demand to deal with contingencies. The manager must now accept the fact that some portion of the contingency pool will not be working on contingency resolution all the time. This pool introduces an intrinsic overhead into the project budget.
2) Second, he or she might hire and fire workers as contingencies come and go. The intent is to reduce the idle-time overhead, although this strategy may suffer as well as the costs of hiring and firing plus unplanned training.

Regardless of the strategy the project manager adopts, it is clear that the stochastic nature of contingency resolution creates an "uncertainty premium" that does not exist for projects that have no uncertainty. In a very real sense, the project must pay to reduce uncertainty.

On Monitoring and Managing Contingency Responses

This paper suggests that project managers could improve their understanding of their project's progress by keeping a history of contingencies and their resolution tasks:

1) The project manager should categorize each contingency as arising either from initial uncertainty or from changes in the project's environment.
2) The project manager should keep a record of the workers assigned to resolve the contingency and the time they devote to this resolution. This record might be a list of action items that are reviewed at staff meetings.
3) On a routine basis, the project manager should create a statistical summary of planned and contingency work to track whether the project is following the initial assumptions regarding contingencies or is deviating significantly from them.

4. REFERENCES

[1] B. W. Boehm, **Software Engineering Economics**, Upper Saddle River: Prentice-Hall PTR. Pub. 1981.

[2] B. W. Boehm, C. Abts, A. W. Brown, S. Chulani, B. K. Clark, E. Horowitz, R. Madachy, D. Reifer, B. Steece, **Software Cost Estimation with COCOMO II**, Upper Saddle River, NJ: Prentice Hall PTR, Pub. 2000.

[3] T. C. Jones, **The SPR Feature Point Method**, Software Productivity Research, Inc., Pub. 1986.

[4] C. R. Symons, **Software Sizing and Estimating: Mk II FPA (Function Point Analysis)**, New York, NY: John Wiley & Sons, Inc., Pub. 1991.

[5] F. P. Brooks, **The Mythical Man-month: Essays on Software Engineering**, 2nd ed., Reading, MA, Addison-Wesley, Pub. 1995.

[6] M. H. Weier, "Numbers Crunch", **InformationWeek**, Issue 1,218, Jan. 26, 2009, pp. 25-29.

[7] D. M. Ahern, A. Clouse, R. Turner, **CMMI Distilled: A Practical Introduction to Integrated Process Improvement, 2nd ed.**, Boston: Addison-Wesley, Pub. 2001.

[8] Q. W. Fleming and J. M. Koppelman, **Earned Value Project Management, 3rd ed.**, Washington: PMI, Pub. 2006.

[9] B. W. Boehm, "Software Risk Management: Principles and Practices", **IEEE Software**, Jan., 1991, pp. 32-41.

[10] B. W. Boehm, "A Spiral Model of Software Development and Enhancement", **Computer**, May, 1988, pp. 61-72.

[11] A. Cockburn, **Agile Software Development**, Boston: Addison-Wesley, Pub. 2002..