MODELING THE PERFORMANCE AND RISKS OF EVOLUTIONARY ACQUISITION

David N. Ford and COL John Dillard, USA (Ret.)

Evolutionary acquisition mandates incremental development for all programs. This policy seeks to improve development project performance, but may increase some risks. Computational modeling using systems dynamics reveals that evolutionary acquisition can increase concurrency and the need for coordination. The result is earlier delivery of the first increment, but later and more costly delivery of subsequent increments than in a single-step methodology. Modeling reveals and explains how deliberate work deferral reduces the initial increment's cost and schedule, but rework and transaction costs cause inefficiency in successive increments. Program managers must be aware of the risks of evolutionary acquisition and take additional steps to mitigate them with disciplined change-control measures, organizational accommodations, and accountability for configuration management.

Keywords: *Evolutionary Acquisition, Spiral Development, Computational Modeling, Simulation*

Image designed by Harambee Dennis »

Evolutionary Acquisition
Spiral Development

Historically, many Department of Defense (DoD) acquisition projects have used a single-block development approach in which each phase of the development process is completed once, and all capability requirements are included in the performance of that process. However, the uncertainty of requirements and complexity of technologies have contributed to large and frequent cost overruns and completion delays. In response, the DoD promulgated *evolutionary acquisition* (EA) as policy in 2000, and soon after, *spiral development* as the preferred acquisition strategy. The EA's primary goal is to reduce product cycle-times by dividing and phasing requirements to provide initial capabilities sooner.

AS LONG AS FUTURE THREATS ARE UNKNOWN OR UNSPECIFIED, ALL REQUIREMENTS ARE NOT (AND CANNOT BE) KNOWN AT THE BEGINNING OF THE PROJECT.

Evolutionary acquisition would seem to greatly alleviate project risks. Incomplete information and uncertainty about system complexity, ambiguous or changing requirements, and the integration of maturing technology have long been primary development risks. Requirements also evolve in response to evolving threats. As long as future threats are unknown or unspecified, all requirements are not (and cannot be) known at the beginning of the project. Technology risk lies in the possibility that future technology development will be unsuccessful, late, or more costly than expected. The EA addresses requirements and technology risks by allowing requirements to evolve over time and by developing only mature technologies, requiring the use of Technology Readiness Levels (TRL) to assess technology maturity. Amorphous spirals eventually become defined project increments when their requirements, technologies, etc., become clear and specific. Thus, at the heart of EA is the iterative and exclusive use of mature technologies to address known and achievable requirements.

However, despite its potential, evolutionary acquisition has proven challenging to implement. For example, a RAND Corporation study (Lorell, Lowell, & Younossi, 2006) found that "evolutionary acquisition and spiral development approaches promote constant flux in all these program attributes, leading inevitably to cost estimating difficulties and cost growth" (p. 102). Research by the authors (Dillard & Ford, 2007) found additional challenges in implementing evolutionary acquisition—specifically in the realms of organizational impacts, institutional biases, transaction costs, and decision process. But we also found examples of its successful employment. (The body of this work is expounded upon in a companion article, and in our full report at: http://www.acquisitionresearch.net/_files/FY2007/NPS-AM-07-002.pdf). If this approach is to be successfully implemented, PMs must understand the potential improvements in performance provided by evolutionary development, its own inherent risks, and how they are related.

PROBLEM DESCRIPTION

Our case studies and other anecdotal data indicate that significant project performance risks are inherent in spiral/incremental development and that these risks affect project planning and execution decisions. Solutions are not obvious, largely because the evolutionary process as a program strategy is more complex overall and is comprised of many more interdependent activities than a single-step to full-capability approach. Some of these relationships are easy to recognize, such as the impact of delaying the start of a second development block until the delivery of the initial block. But many relationships and their impacts are difficult to recognize and predict, such as the impacts of the concurrency of a second development block with the first and the amount of rework generated by different amounts of overlapping. Our case study research and *computational modeling* indicate that these hidden, secondary impacts of EA can have more significant influence on project performance and risk than revealed in EA policy. Thus, we must recognize and describe the EA relationships that drive performance and risk to understand their impacts.

Single-block development to full requirements is a traditional and relatively well-understood acquisition approach that can provide a baseline for the evaluation of evolutionary development. Therefore, the current work focuses on two questions, which contrast evolutionary development and single-block development:

- What are the impacts of an evolutionary development approach in contrast to a traditional single-block development strategy?
- How might successful evolutionary development project performance differ from the successful management of single-block development projects?

A DYNAMIC MODEL OF EVOLUTIONARY DEVELOPMENT

Evolutionary development is a complex process that evolves over time and can be better understood through formal modeling of the most important components and relationships that drive performance and risk. The formal model structure and rigor of calculations can simulate and forecast performance and risk better than informal tacit predictions by humans due to the number and complexity of the components and their relationships. Therefore, we applied a computational experimentation approach to investigating acquisition projects, integrating theory and practice in a computational tool that allows controlled experimentation through *simulation*. The current work reflects project, product development, and management theories. We also reflect practice in the model through the use of a case study to build and validate the model structures and model calibration and testing. We applied the system dynamics methodology for model development and use. System dynamics uses a computational experimentation approach to understanding and improving dynamically complex systems. The system dynamics perspective focuses on the roles of accumulations and flows, feedback, and nonlinear relationships in managerial control. The methodology's ability to model many diverse system components (e.g., work, people, money), processes (e.g., design, technology development, quality assurance), and managerial decision-making and actions (e.g., forecasting, resource allocation) makes it useful for investigating acquisition projects. Forrester (1961) develops the methodology's philosophy and Sterman (2000) specifies the modeling process with examples and describes numerous applications. System dynamics has been applied to projects for several decades and has built a collection of validated development project structures (Lyneis & Ford, 2007), several of which are used in the current work.

The authors based the model on previously developed system dynamics models of product development in several industries and the military—models that have been developed and tested over several decades (e.g., Cooper, 1980; Abdel-Hamid, 1988; Ford & Sterman, 1998; 2003). Thus, the model is founded on well-established and tested components. Although these previous models have developed structures for many components and aspects of acquisition, they have not yet been used to investigate acquisition approaches such as spiral/incremental development as used by the DoD.

A CONCEPTUAL MODEL OF EVOLUTIONARY DEVELOPMENT

In the model, four types of work flow through each block of an acquisition project: requirements, technologies, product component designs, and products. Within a development block, each type of work flows through a phase in which developers complete a critical aspect of the project: 1) develop requirements, 2) develop technologies, 3) design product components (advanced development), and 4) manufacture products. Requirements also flow through the final phase: 5) user product testing. Development phases and information flows in a single-block structure (as depicted in the model) are shown in Figure 1. Arrows between phases indicate primary information flows. The beginning of all phases—except the development of requirements—is constrained by the completion of previous ("upstream") phases. Figure 1 also identifies the five major reviews within a single acquisition block.

Development processes are constrained by both the physical and information relationships among the activities and phases within a development block. These constraints include development activity durations and precedence relationships, information dependencies leading to iteration (Smith & Eppinger, 1997), the availability of work (Ford & Sterman, 1998), coordination mechanisms (Hauptman & Hirji, 1996), the characteristics of information transferred among development phases (Krishnan, 1996), and the number, skill, and experience of project staff (Abdel-Hamid, 1988).

FIGURE 1. INFORMATION FLOWS IN A SINGLE-BLOCK ACQUISITION PROJECT

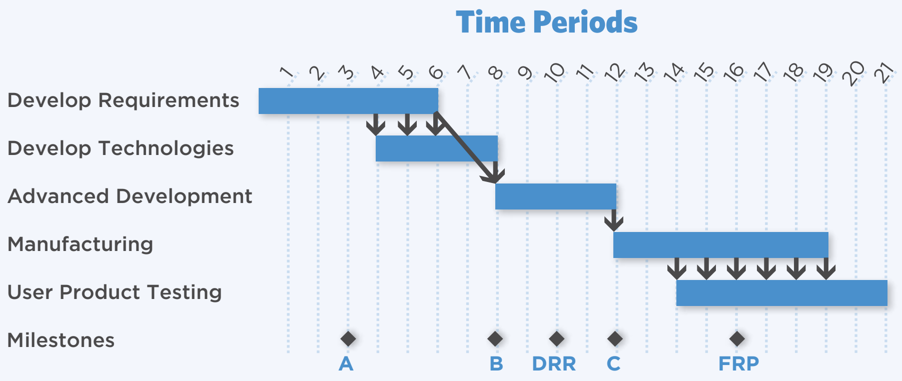
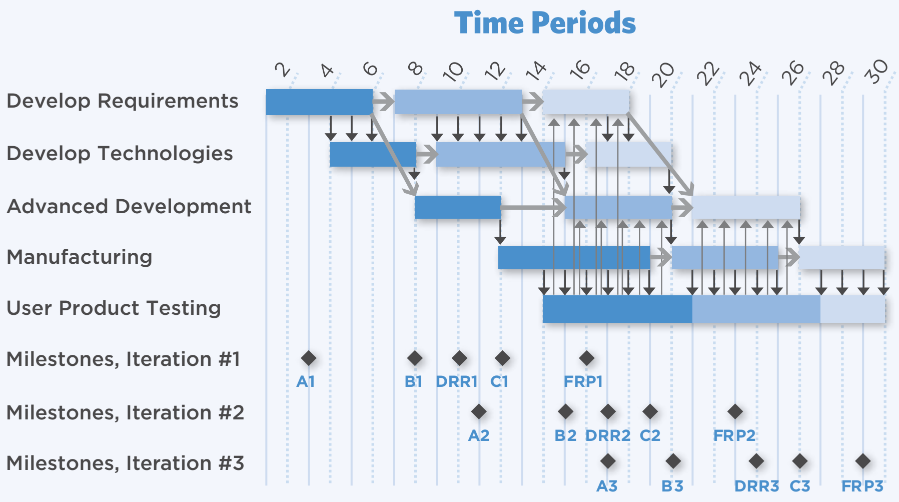


Figure 2 depicts an acquisition project with multiple increments or blocks. The first block is the same as in Figure 1. Subsequent blocks have the same basic information flow, but can also be delayed by the completion of phases in previous blocks or constrained by the progress in their own blocks. Importantly, in addition to the flow of information downstream through phases (downward pointing arrows in Figure 2), multiple iteration acquisition also provides opportunities for information to flow upstream—such as from User Product Testing of an earlier iteration to Develop Requirements or Advanced Development in a subsequent iteration (upward pointing arrows in Figure 2).

FIGURE 2. INFORMATION FLOWS IN A SPIRAL DEVELOPMENT PROJECT: A CONCEPTUAL MODEL



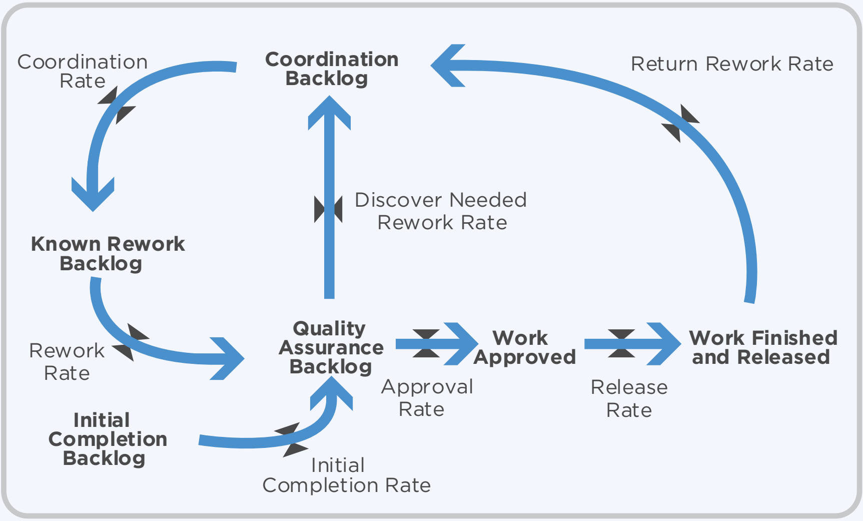
In the model, the structure of each block is the same, although parameter values are varied to reflect different acquisition projects and strategies. For example, all phases include start-up work that is not directly applied to generating development products. Each phase also includes the requisite decision review work that also does not directly generate product. This is consistent with Government Accountability Office recommendations and DoD policy to manage and oversee each development block like an individual project (Milestone B to Milestone C). One impact of this requirement for each phase to include start-up and review work when multiple development blocks are used is a significant increase in the total amount of work required to provide a given set of requirements to warfighters. As will be shown with the model, this work has a significant impact on project performance that may impact the types of projects in which evolutionary development can be effective.

A FORMAL MODEL OF EVOLUTIONARY DEVELOPMENT

The conceptual model described above was used to build a formal computer simulation model of an acquisition project that can reflect traditional and evolutionary development strategies. The model represents workflows through a project phase as a value chain of alternating backlogs and development activities with two rework cycles (Figure 3). The value chain is described with the boxes and pipes with valves along the bottom of Figure 3. The value chain passes from the Initial Completion Backlog through the Initial Completion Rate into the Quality Assurance Backlog, through the Approval Rate into the stock of Work Approved, and through the Release Rate to the accumulation of Work Finished and Released. The rework cycle is inherent in development projects and has been modeled and used extensively to explain and improve project management (Lyneis, Cooper & Els, 2001; Ford & Sterman, 1998; Cooper & Mullen, 1993; Cooper, 1980; 1993a, b, c, 1994). Each phase includes an intra-phase rework cycle from quality assurance through coordination, rework, and back to quality assurance. Development blocks also include inter-phase rework cycles from quality assurance through work approved, work finished and released to downstream phases, rework discovery in downstream phases (not shown in Figure 3 for clarity), the return of information on work to be reworked, coordination, rework, and back to quality assurance.

Given the arrangement of development activities in a phase described above, progress is constrained by the rate at which work packages move through the flows that connect the stocks. Four development activities and several development features control rates. The initial completion, quality assurance, coordination, and rework rates are each constrained by the rate allowed by the availability of work or the rate allowed by the resources applied (described later). The rates allowed if the development process has infinite resources (i.e., uncapacitated conditions) are described with an average processing time assuming all labor, equipment, knowledge and understanding are available. Project progress depends largely on how much work gets trapped in the rework cycles versus how much “leaks out” of the rework cycles through approval. The fraction of work discovered to require rework is used to model project complexity. More complex projects are assumed to require more iterations for completion.

FIGURE 3. WORK BACKLOGS AND FLOWS THROUGH A DEVELOPMENT PHASE



The model, shown earlier in Figure 2, simulates two types of development resources. Either resource type can constrain progress by limiting the development rate. Direct resources are the people and associated equipment required to perform the development work, i.e., to develop requirements, develop technology, design products, manufacture products, and test requirement satisfaction for use. Indirect resources perform project management and associated work that support and facilitate development. Total direct resources are assumed fixed and allocated based on the backlogs of work available to be developed (the stocks represented as boxes in Figure 3). In contrast, indirect resources (also assumed fixed) serve the performance of activities (the development rates, the pipes with valves in Figure 3) and are distributed proportionately based on the size of those development activities.

Projects are measured in three dimensions: schedule, cost, and performance risk. Schedule performance is measured in the time required to have a given number or fraction of requirements tested and approved by users. Cost is measured in dollars based on the size of direct and indirect work forces and the duration of phases and blocks. Performance risk is measured with the average percent of the requirements provided (approved by users) at any given time. This average reflects the combination of multiple requirements. Some of the requirements may have binary performance, i.e., they work or they don’t work. Other requirements may have discrete steps or continuous performance relative to requirements, such as weight or unit manufacturing cost. All the requirements can be considered met completely when the average percent of the requirements provided is 100 percent for a development block.

MODEL CALIBRATION AND TESTING

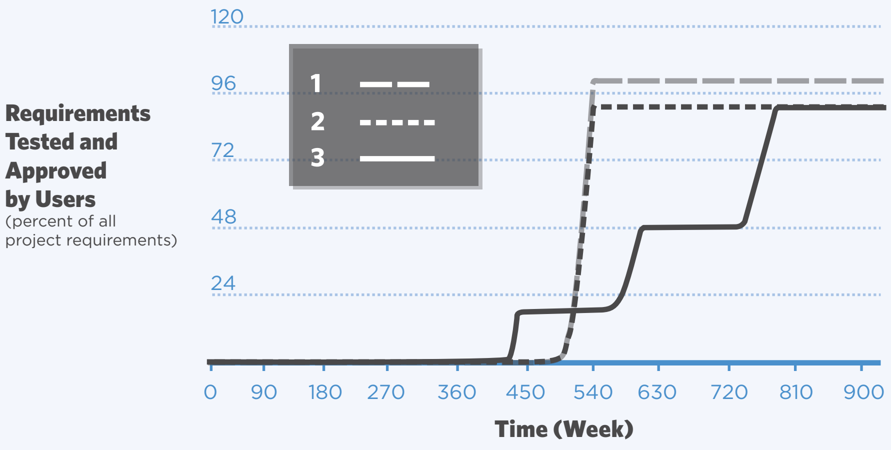
The formal model was calibrated to the Javelin project described in Dillard and Ford (2007). Basing the model on previously validated models, the literature and data collected about acquisition projects improve the model's structural similarity to actual acquisition projects as practiced. Model behavior was tested with extreme input values, such as no discovery of errors and very large resource quantities and productivities, as well as more typical conditions. Model behavior remained reasonable across wide ranges of input values, including extreme values. These tests increase confidence that the model generates realistic project behavior patterns due to the same causal relations found in the type of projects investigated (i.e., generates "the right behavior for the right reasons").

The model also reproduces the known system behavior. The simulated behavior of the Javelin project is consistent with the phase durations provided by the project manager, supporting the ability of the model to reflect the dynamics of the Javelin project. The simulated cost of the Javelin project ($722 million) is also consistent with the data provided by the project manager, supporting the ability of the model to reflect the Javelin project's cost performance. As an additional test of model usefulness, the size of the development staff was doubled for the Javelin calibration project. If the model reflects actual projects, this change should speed up development but increase costs, as more resources generate products faster but at much higher cost. Doubling the number of developers saves 30 weeks (100 percent of requirements satisfied in week 491 instead of week 521) but increases costs dramatically from $722 million without the larger development staff to $1,327 million (an 83 percent increase). Based on these and additional tests, the model is considered useful for the investigation of the impacts of acquisition strategies on project performance.

MODEL USE

We investigated the impacts of evolutionary development on acquisition project performance by simulating the same project using a traditional single-block development strategy and an incremental development strategy, and by comparing and contrasting the behavior of the two projects. The calibration project case (Javelin) fully satisfied all its requirements. However, not satisfied, or partially satisfied requirements reflect the project's risk and are, therefore, important performance measures. Therefore, to facilitate the comparison of project performance using different strategies, a Base Case project was created that does not fully satisfy all requirements. Figure 4 on Page 152 shows the Performance Risk Profile of three project simulations: 1) the Javelin calibration project (wide dashed line #1), 2) the Base Case project (Javelin without 100 percent satisfaction) using a single-block strategy (narrow dashed line #2), and 3) the Base Case project using an incremental development strategy, with the requirements and work distributed evenly across three development blocks (thick solid line #3).

FIGURE 4. PERFORMANCE RISK PROFILE OF A CALIBRATION, BASE CASE, AND EVOLUTIONARY PROJECT



The Table compares the performance of these three simulated projects. The first two performance measures reflect schedule performance, with the project duration required to satisfy the first requirement and the project duration required to satisfy all the requirements. The third performance measure reflects cost performance with the estimated development cost. The last two performance measures reflect project risk, with the percent of the total project requirements satisfied by a specific deadline. For the Table, the deadline was chosen to be the time when the Base Case project using the traditional strategy satisfied all of the project's final requirements.

TABLE 1. PERFORMANCE COMPARISON OF THREE SIMULATED ACQUISITION PROJECTS

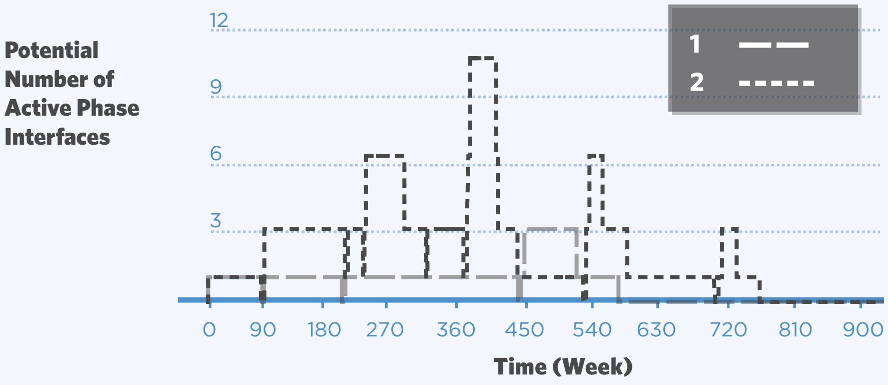
		Project Scenario		
Performance Measure	**Units of Measure**	*Javelin*	*Base Case Traditional*	*Base Case Spiral*
Duration to first requirement satisfied	Weeks	*471*	470	***397***
Duration to maximum requirements satisfied	Weeks	520	**518**	*762*
Total development cost	$1.0 million	722	**719**	*1,555*
Requirements satisfied by deadline	Percent	**100**	91	*18*
Final requirements satisfied	Percent	**100**	*91*	*91*

The results in the Table identify important impacts of incremental development on acquisition project performance when compared to a traditional single-block strategy. Underlined bold values in the Table indicate the best performance among the three projects for each performance measure. Values in italics indicate the worst performance among the three projects for each performance measure. Notice that the Base Case, spiral project is best in only one performance measurement (Duration to first requirement satisfied) but is worst in three other performance measurements (Duration to max. requirements satisfied, Total development cost, and Risk—requirements satisfied by deadline). These data demonstrate the ubiquitous tradeoffs in performance that different strategies present. If all performance measures were valued equally, evolutionary development would appear to be a poor choice as an acquisition strategy. However, not all performance measures are of equal value in all acquisition projects. These model results identify the one performance measure that must be most important for an evolutionary development strategy to improve total project performance—Duration to first requirement satisfied.

The first step in improving the management of evolutionary development is to understand the managerial implications inherent in such development. Phases must be coordinated with external stakeholders and other development phases. Each pair of concurrent phases creates a potential interface that requires coordination. Coordination needs of traditional versus evolutionary development were contrasted using the active development phases of the Base Case project, first assuming that a single development block was used and then assuming that evolutionary development was used. Figure 5 shows an estimate of the phase interfaces that must be managed based on the number of active phases in the simulation described previously.

Although the number of interfaces with external stakeholders and between development phases is project-specific, the impact of evolutionary development on project management requirements is clear. Evolutionary development (narrow dashed line #2 in Figure 5) requires significantly more coordination than single-block development (wide dashed line #1 in Figure 5).

FIGURE 5. POTENTIAL COORDINATION REQUIREMENTS WITH SINGLE-BLOCK AND EVOLUTIONARY DEVELOPMENT

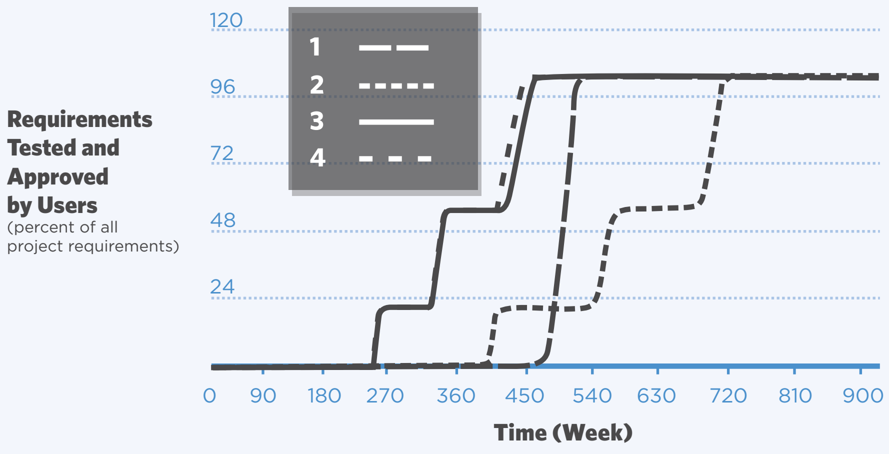


THE CRITICAL ROLE OF PROGRESS BOTTLENECKS

The management of the constraints on development progress is critical to evolutionary development project success. Bottlenecks that constrain development progress can be caused by several different parts of a development project and can be located in many different places. This can be illustrated by simulating projects using evolutionary development with different amounts of resources—a common project-management tool. To investigate the impacts of different resource policies on project bottlenecks and progress, we simulated the Javelin Project assuming four conditions:

- a single-block approach (wide dashed line #1 in Figure 6)
- an evolutionary approach (narrow dashed line #2 in Figure 6)
- an evolutionary approach and additional developers (solid line #3 in Figure 6)
- an evolutionary approach with additional developers and additional project management (medium dashed line #4 in Figure 6).

FIGURE 6. IMPACTS OF ADDING RESOURCES ON PERFORMANCE



The addition of developers reduces the duration of Block 2 (second and third steps are earlier), but does not significantly change the duration of the first block. This is because the first increment is constrained by process, not the number of developers. This result illustrates the importance of identifying and understanding the progress bottleneck. In this case, the addition of developers does not significantly reduce the first development block and would not be a very effective policy (or use of resources) if a project manager was attempting to accelerate the time to First Unit Equipped with the capabilities provided by the first block. Adding resources where they do not relax a progress constraint does not improve performance (an old lesson). But, the discovery of which project features constrain progress, at what points, and exploiting that knowledge is particularly difficult in evolutionary development because of the increased project dynamics (a new lesson). In contrast, the addition of developers improves performance if the management objective was to speed the time to the First Unit Equipped with capabilities from the second block. Again, discovering and exploiting which project features constrain progress, and at what points, is critical for improving evolutionary development performance.

The addition of project management resources in addition to developers (medium dashed line #4 in Figure 6) also illustrates the challenges and importance of identifying and understanding progress bottlenecks in evolutionary development. This policy only impacts the third development block. This is because in the model, as calibrated, the first two development blocks have adequate project management; therefore, the addition of more project management does not improve performance. In contrast, the third development block is (at least partially) constrained by project-management resources, and benefits by the addition of more project management. In this case, the location of the bottleneck shifts from developers to project managers and is different in different development blocks. The fundamental lesson from the model is the same: Understanding the location of progress bottlenecks is particularly difficult but vital for successful evolutionary development management.

THE COUNTER-INTUITIVE COST BEHAVIOR OF THESE SIMULATED PROJECTS ILLUSTRATES THE CHALLENGES AND IMPORTANCE OF IDENTIFYING AND UNDERSTANDING PROGRESS BOTTLENECKS IN EVOLUTIONARY DEVELOPMENT PROJECTS.

The estimated costs of the four simulated Javelin projects shown in Figure 6 are: 1) single-block: $704 million, 2) spiral development: $939 million, 3) spiral development with additional developers: $1,761 million, and 4) spiral with additional developers and project management: $1,753 million. The first increase in cost from a single-block development ($704 million) to a spiral development ($939 million) is due largely to increased transaction costs (e.g., oversight) and has been discussed previously. The second increase in cost from spiral development ($939 million) to spiral development with more developers ($1,761 million) is also expected and is due to the larger workforce. However, the decrease from spiral with more developers ($1,761 million) to spiral with more developers and more project management ($1,753 million) is counterintuitive. How can adding more resources (project management) decrease project costs? An analysis of the model structure reveals that when project management resources constrain progress, adding those resources can reduce project duration, allowing an earlier release of the (expensive) developers from the project. Without the additional project management, some developers are unable to be fully utilized due to project management issues that are not being addressed. The additional project management resources relaxed that progress bottleneck, thereby allowing improved use of developers, faster completion of the project, and reduced costs. The counter-intuitive cost behavior of these simulated projects illustrates the challenges and importance of identifying and understanding progress bottlenecks in evolutionary development projects.

CONCLUSIONS

A simulation model was used to investigate the impacts of evolutionary development on acquisition projects and the management of evolutionary development. Evolutionary development was found to have several important impacts on acquisition projects when compared to a traditional single-block development approach. Ceteris paribus (all other things held constant or equal), the model found, or supported other findings of, the following impacts:

- Incremental/spiral development can provide the First Unit Equipped with some (but not all) requirements satisfied faster than single-block development.
- Incremental/spiral development provides satisfied requirements to users in multiple steps or increments, whereas single-block development satisfies all requirements in a single step.
- Incremental/spiral development costs more than single-block development to satisfy the same requirements.
- Incremental/spiral development has a high risk of not satisfying all requirements by the time single-block development can satisfy all requirements.
- The drivers of and constraints on evolutionary acquisition project performance can be more difficult to identify than those influencing single-block development projects.

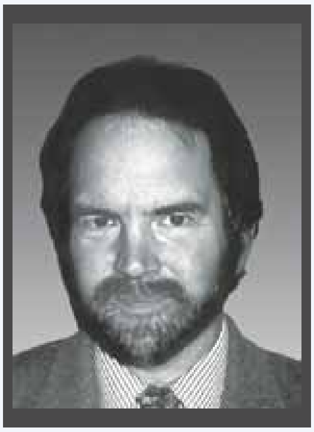
Evolutionary development was also found to have several significant impacts on acquisition project management. Investigations with the model found that (ceteris paribus):

- The concurrent use of multiple development blocks in evolutionary development significantly increases the number of development phases and activities that must be managed and coordinated at any given time compared to single-block development. This increases the project management needs for successful acquisition in evolutionary development projects when compared to single-block projects.
- As in single-block development, progress in evolutionary development requires the identification and understanding of progress bottlenecks. However, the concurrence and resulting complexity of development in evolutionary projects causes the types and locations of bottlenecks to vary widely and be more difficult to identify and address than those in single-block development.

- Progress bottlenecks can cause counterintuitive behavior, such as reductions in project cost by adding resources at a bottleneck. Exploiting the opportunities provided by these behaviors requires a deep understanding of the project structures and dynamic interactions that drive and constrain progress.

These results indicate that evolutionary development requires more, different, and more difficult project management than single-block development. They also suggest that project management should focus on the identification and management of causal paths, information feedback, and progress bottlenecks based on the structure of the development project. By doing so, project managers can improve the design and management of evolutionary development in DoD acquisition projects and can, thereby, capture the benefits and mitigate the risks of evolutionary acquisition.

Author Biographies



Dr. David N. Ford is an associate professor in the Zachry Department of Civil Engineering, Texas A&M University. In addition to teaching, he researches development project strategy, processes, and resource management. Dr. Ford earned his doctorate from the Massachusetts Institute of Technology and his master's and bachelor's degrees from Tulane University. He has over 14 years of engineering and project management experience in industry and government.

(E-mail address: DavidFord@tamu.edu)

COL John T. Dillard, USA (Ret.), managed weapons development efforts for most of his 26-year career in the military. He is now a senior lecturer with the Graduate School of Business and Public Policy at the U.S. Naval Postgraduate School in Monterey, California. COL Dillard is a 1988 graduate of the Program Manager's Course, Defense Systems Management College.

(E-mail address: jtdillard@nps.edu)

REFERENCES

Abdel-Hamid, T. (1988). Understanding the “90% syndrome” in software project management: A simulation-based case study. *The Journal of Systems and Software, 8*(4), 319–330.

Cooper, K. G. (1980). Naval ship production: A claim settled and a framework built. *Interfaces, 10*(6), 20–36.

Cooper, K. G. (1993a, February). The rework cycle: Why projects are mismanaged. *PM Network, 7*(2), 5–7.

Cooper, K. G. (1993b, February). The rework cycle: How it really works ... and reworks. *PM Network, 7*(2), 25–28.

Cooper, K. G. (1993c). The rework cycle: Benchmarks for the project manager. *Project Management Journal, 14*(1), 17–21.

Cooper, K. G. (1994). The $2,000 hour: How managers influence project performance through the rework cycle. *Project Management Journal, 15*(1), 11–24.

Cooper, K. G., & Mullen, T. (1993). Swords and plowshares: The rework cycles of defense and commercial software development projects. *American Programmer, 6*(5), 41–51.

Dillard, J., & Ford, D. (2007). *Too little too late? Modeling the risks of spiral development.* In Proceedings of the 4th Annual Acquisition Research Symposium. Monterey, CA: Naval Postgraduate School.

Eppinger, S. D., Whitney, D. E., Smith, R. P., & Gebala, D. A. (1994). A model-based method for organizing tasks in product development. *Research in Engineering Design, 6*(1), 1–13.

Ford, D., & Sterman, J. (1998, Spring). Modeling dynamic development processes. *System Dynamics Review, 14*(1), 31–68.

Ford, D., & Sterman, J. (2003, September). Overcoming the 90% syndrome: Iteration management in concurrent development projects. *Concurrent Engineering Research and Applications, 111*(3), 177–186.

Forrester, J. W. (1961). Industrial dynamics. Cambridge, MA: MIT Press.

Hauptman, O., & Hirji, K. K. (1996). The influence of process concurrency on project outcomes in product development: An empirical study of cross-functional teams. *IEEE Transactions on Engineering Management, 43*(2), 153–178.

Krishnan, V. (1996). Managing the simultaneous execution of coupled phases in concurrent product development. *IEEE Transactions on Engineering Management, 43*(2), 210–217.

Lorell, M. A., Lowell, J. F., & Younossi, O. (2006). Evolutionary acquisition: Implementation challenges for defense space programs. Santa Monica, CA: RAND Corporation.

Lyneis, J. M., Cooper, K. G., & Els, S. A. (2001). Strategic management of complex projects: A case study using system dynamics. *System Dynamics Review, 17*(3), 237–260.

Lyneis, J. M., & Ford, D. N. (2007). System Dynamics Applied to Project Management: A survey, assessment, and directions for Future Research. *System Dynamics Review, 23*(4), 157–189.

Simon, H. A. (1981). The sciences of the artificial (2nd ed.). Cambridge, MA: MIT Press.

Smith, R. P., & Eppinger, S. D. (1997). Identifying controlling features of engineering design iteration. *Management Science, 43*(3), 276–293.

Sterman. J. D. (2000). Business dynamics, system thinking and modeling for a complex world. New York: Irwin McGraw-Hill.