Simulation: An Enabling Technology in Software Engineering

Alan M. Christie
Software Engineering Institute

This article suggests three reasons why the software engineering community could exploit simulation to a much greater advantage. First, the Office of the Secretary of Defense has indicated that simulation will play a significant role in the acquisition of defense-related systems to cut costs, improve reliability, and bring systems into operation more rapidly. Second, there are many areas where simulation can be applied to support software development and acquisition. Such areas include requirements specification, process improvement, architecture trade-off analysis, and product-line practices. Third, commercial simulation technology, capable of supporting software development needs, is now mature, easy to use, low cost, and readily available.

What Is So Great About Simulation?

Simulation can be applied in many critical areas. It allows issues to be addressed before they become problems. Simulation is more than just a technology because it forces one to think in global terms about system behavior and about systems being more than the sum of their components. Simulation can provide insight into the designs of processes, architectures, or product lines before significant time and cost have been invested and can be of great benefit in support of training. Simulation is being increasingly emphasized in the Department of Defense (DoD) community, where there is documented evidence that its impact on cost, quality, and schedule is nontrivial. I believe that the software engineering community needs to take a stronger role in exploiting the technology [1].

The DoD Emphasis on Simulation

The Office of the Secretary of Defense has recently initiated an effort focused on the use of modeling and simulation to support improvement of the acquisition process. Jacques Gansler, undersecretary of defense for acquisition and technology, states, "A directive which I issued this year [1998] requires the integration of modeling and simulation in our acquisition process—across functional disciplines—and throughout the lifecycle of systems. We are committed to reforming the acquisition system and recognize that an essential tool for accomplishing that reform will be modeling and simulation." [4] Although Gansler does not explicitly include the software acquisition process, there is no reason to doubt that software acquisition can benefit as much as any other DoD acquisition area, as will be explained further.

Gansler's remarks are reinforced by those of Patricia Sanders, director of defense, test, system engineering, and evaluation. In "Simulation-Based Acquisition," [5] she states that the DoD needs to become a smart buyer and that in evaluating what to buy, simulation will be a key component. She says that, "Without question, the Defense Department is moving toward greater use of simulation-based system acquisition." She indicates that,

> "The Defense Department envisions an acquisition process supported by the robust, collaborative use of simulation technology that is integrated across acquisition phases and programs. The objectives of Simulation-Based Acquisition (SBA) are to:
> 1. Reduce the time, resources, and risk associated with the acquisition process;
> 2. Increase the quality, military utility, and supportability of systems developed and fielded; and
> 3. Enable integrated product and process development from requirements definition and initial concept development through testing, manufacturing, and fielding." [5]

Sanders provides evidence from commercial and military programs to show that the use of simulation has had major positive impacts from the perspectives of cost, schedule, and productivity. Following are some of her examples.

> "Cost – … In the Joint Strike Fighter program, it is projected that virtual manufacturing techniques may save as much as 3 percent of the program's estimated lifecycle cost, which could be $5 billion.
> Schedule – The use of modeling and simulation tools and processes by the "big three" auto manufacturers has reduced the time from concept approval to production from 5 to 3 years. …
> Productivity – … It took 38 Sikorski draftsmen approximately six months to develop working drawings of the CH-53E Super Stallion's outside contours. In contrast, using modeling and simulation, one engineer was able to accomplish the same task for the

The Software Engineering Institute's work is supported by the Department of Defense.

Commanche helicopter in just one month. …" [5]

Clearly, the use of simulation in the above examples is different from that in software development; however, there are sufficient parallels that would tend to indicate that similar advantages can be accrued in the software arena. For example, although physical mock-ups are not used in software development, early prototypes are used to the same advantage, e.g., determining system characteristics prior to large investments in implementation.

There are other common problems shared between the physical systems and software systems. Examples are

- The management of changing requirements and predicting the consequences of such changes.
- The development and optimization of effective processes through which the product is built.
- The estimation and tracking of project costs and schedules.

In addition, in 1998, the DoD developed an overall action plan to integrate the various simulation-based acquisition activities ongoing at the DoD. This action plan was developed by a joint SBA task force whose aim is "an acquisition process in which the DoD and industry are enabled by robust, collaborative use of simulation technology that is intended to integrate across acquisition phases and programs." [6]

The Need for Simulation in Software Engineering

Why can simulation enhance traditional software engineering? An important factor is that it provides insights into complex process behavior. Like many processes, software processes can contain multiple feedback loops such as those associated with correction of defects in design or code. Delays that result from these effects may range from minutes to years. The complexity that results from these effects and their interactions makes it almost impossible for human (mental) analysis to predict the consequences. Unfortunately, traditional process analysis does not shed much light on these behavioral issues, and the usual way to resolve them is to run the process and observe the consequences. This can be an extremely costly way to perform process improvement.

Assessing the Costs of Software Development

At an applied level, simulation can support project costing, planning, tracking, and prediction. In a competitive world, accurate prediction provides a significant advantage. If cost estimates are too high, bids are lost; if too low, organizations find themselves in debt. In this context, simulation can provide not only estimates of cost but also estimates of cost uncertainty. Simulation is a powerful tool to aid activity-based costing and can incrementally accumulate costs to an extremely fine degree of resolution. In addition, it can assess the uncertainty of costs based on the interacting uncertainties of independent variables [7, 8].

Supporting Metric Collection

Simulation is effective only if both the model and the data used to drive the model accurately reflect the real world. There is a tight connection between the model and the data in the sense that a simulation can only be executed if it is supplied with numerical drivers, which forces the developer to identify points in the model where these drivers are needed. For example, one set of data that needs to be entered in the model may be what percentage of design documents pass review and what percentage must be returned for further work. Thus, in the construction of the model, points where metric data must be collected fall out as a bonus. This approach forces the collection of metric data in a consistent sense from a systems perspective—it is not merely "nice to have" data. Often, too much or too little metric data is collected because the analyst does not have clear guidelines on what is essential.

Building Consensus and Communication

When changes to a process are proposed, experience is likely to be the most important influence. However, experience may not be enough to correctly assess behavioral changes resulting from process modifications. One person's experience may not correspond to another's, and subjective judgment comes into play as to whose opinion is correct. Usually, the person with greater authority wins. With the ability to quantify the effects through simulation, a much greater degree of insight and understanding can be brought to bear on the decision-making process. Therefore, simulation can be a significant influence in communication and consensus building. In this context, alternate process designs can be considered in a quantitative manner with respect to such issues as bottlenecking, resource availability, throughput, and costs. These analyses should result in processes that, once installed, will have a considerably higher probability of satisfactory operation.

A Discrete Simulation Model

There are many approaches to simulation. Some simulations are based on the need to visualize the airflow across a wing section, whereas others designed for combat or flight training need a virtual reality component. However, the types of simulations presented here use symbolic networks of linked elements that model processes or products. For example, it is possible to model entities flowing through the departments of an organization or model information flowing between a set of integrated software tools. Techniques such as discrete event simulation and systems dynamics are often used here.

To make concrete the type of simulations to which I refer, Figures 1 and 2 show components of a discrete simulation model ("discrete" because the entities that flow through the system are modeled discretely). The model was developed with the Extend tool [2] and depicts a call-center type of process for a computer security incident response team (CSIRT). CSIRTs, such as the CERT® Coordination Center at the Software Engineering Institute (SEI), support organizations that have been compromised by unauthorized computer intrusions or that wish to obtain information about guarding against such intrusions. CSIRTs have to communicate with many victimized organizations,

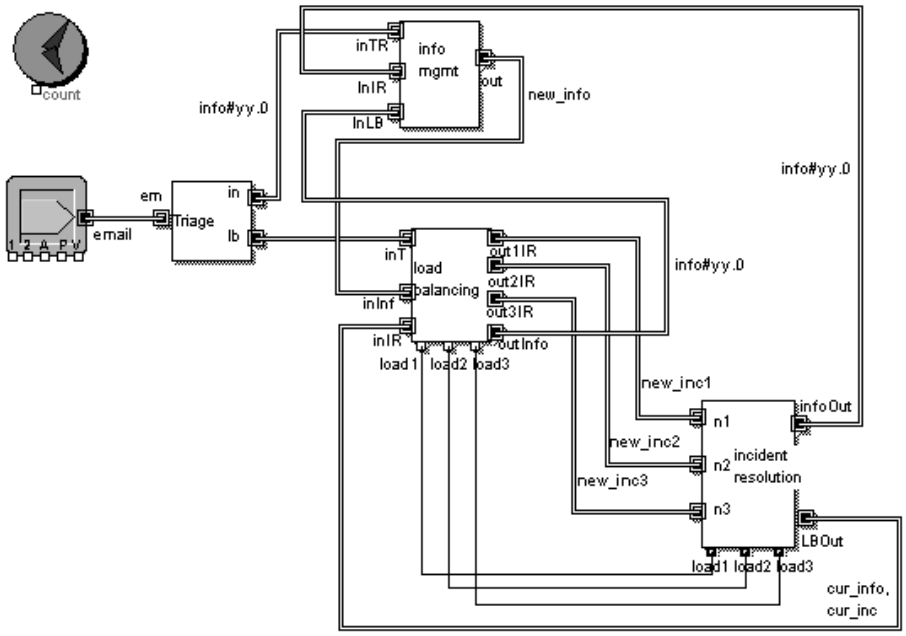


Figure 1. *Top-level view of a CSIRT process. This simulation models the flow of information through a CSIRT.* Triage *handles all incoming new messages.* Load balancing *distributes new requests for help among the incident-handling team.* Incident resolution *is the collective name for the team members who resolve incidents.* Information management *provides responses to routine requests for information.*

and one incident may be composed of numerous E-mail dialogs; hence, the need for a formal work-flow process to manage the large number of interactions while making sure that efficiency and responsiveness are maintained.

Figure 1 shows the top-level component of the incident-handling model.[1] New incidents are inserted into the process at the left, where they queue up to be handled by triage. In triage, E-mail is assigned to either the load-balancing function or to information management. In load balancing, the incidents are assigned to incident handlers based on the incident handlers' loads or their areas of expertise. Subprocesses take care of the details of the four activities identified in Figure 1 (triage, load balancing, incident resolution, and information management), which are all modeled in the simulation. Figure 2 illustrates the subprocess for the load-balancing area.

Upon running the model, various plots can be produced. In this example, both the queue in front of Incident Handler 3 (see Figure 3) and the number of completed E-mail completed for each of the incident handlers are plotted (see Figure 4).

Simulations such as this can be extremely useful for designing effective processes and for predicting the resources needed (both human and computer) so that the anticipated loads can be handled. This model contributed to generating synthetic incident data that supported performance tests on a CSIRT work-flow environment. Without such data, SEI would not have been able to assess performance at such an early state of the work-flow system's development.

For more background on technical issues associated with simulation modeling, consult [3].

Leveraging Simulation Across Applications

As illustrated in the next section, simulation can support a wide variety of applications; therefore, the marginal investment in simulation tools, training, and experience building diminishes as the technology is introduced to successively new applications.

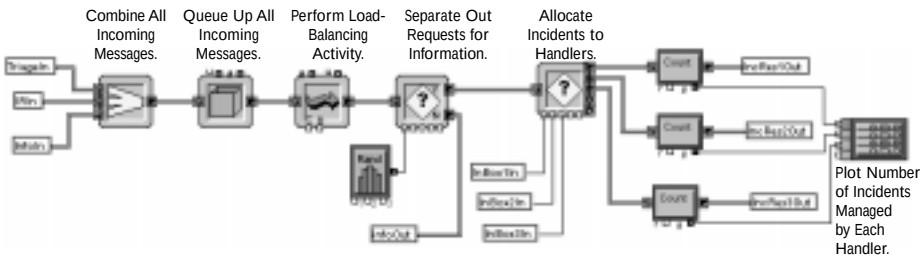
Target Applications for Simulation in Software Engineering

Simulation has been applied in many fields, such as aerospace and energy production, but to date, it has not seen broad practical application in software engineering. This may be because it is more difficult to accurately model human and organizational behavior than to model physical systems, or it may be that the emphasis on software process is a relatively recent phenomenon. Whatever the reason, it is unfortunate because the rewards from its use are myriad. In this section, I briefly review applications of simulation (in no particular order) and some of the benefits that can be obtained.

Requirements Management

Simulation can be extremely helpful in pinning down software system requirements early in the product lifecycle, particularly when examining temporal behavior. Simulation can mimic the performance characteristics of software components and their interactions, the effects of time delays and feedbacks, and of finite capacities and resource bottlenecks [9, 10]. The CSIRT example in Figure 1 illuminates these issues. Alternate architectures and designs can

Figure 2. *The load-balancing subprocess. The load-balancing activity attempts to assign incoming incidents to the appropriate incident handlers, i.e., those with lower current loads or those with expertise in the specific incident.*



be evaluated in a safe environment prior to implementation. In addition, requirements are rarely static but evolve as experience grows with product development. Thus, simulation is not only a valuable tool in defining the initial requirements but also can be used to test alternate modifications prior to their implementation. Finally, a system simulation can be viewed as a component of the requirements and can provide quantitative measures against which the target software system must comply.

The processes through which the requirements are managed also are critical. However, as far as modeling is concerned, such processes have much in common with other project management processes, e.g., design, development, and test. Thus, the discussion in the next section is relevant to requirements management.

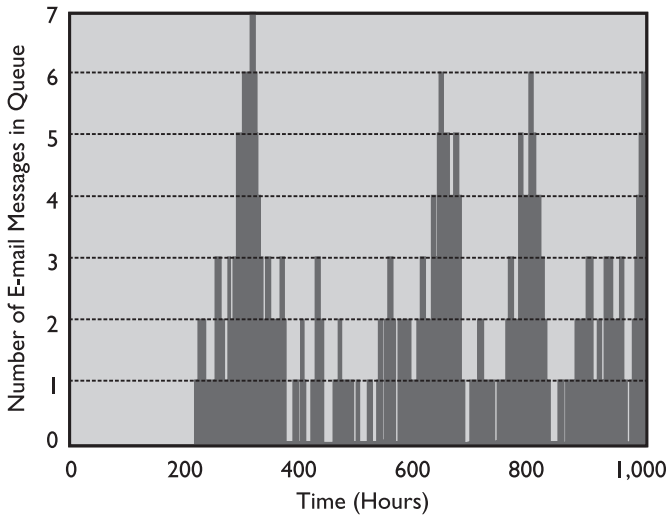
Project Management

Simulation can allow managers to make more accurate predictions about both the schedule and the accumulated costs associated with a project [11, 12]. This approach is inherently more accurate than costing models based on fits to historical data because it accounts for the dynamics of the specific process. With regard to schedule, simulation can account for dependencies between tasks, finite capacity resources, and delays resulting from probable rework loops. Some simulation tools also allow one to compute the accumulation of costs on an activity-dependent basis. These features are useful for generating proposals that are more accurate in cost and schedule and therefore more likely to keep a company in business.

Training

Because of the complex dependencies between attributes of organizational systems, these systems can respond in counterintuitive ways. (The classic example is Brooke's law, which states that hiring people late in a project can further delay the project.) Simulation can play an important role in sensitizing managers to the consequences of instabilities that result from the system feedbacks often inherent in badly designed organizational processes. Simulation-based training also can provide software development managers with the insights necessary to establish effective processes and to operate these processes in a stable manner. Thus, the focus is to train management in the design and operation of software processes, not in the technologies that support software development.

Figure 3. *The E-mail queue for Incident Handler 3.*



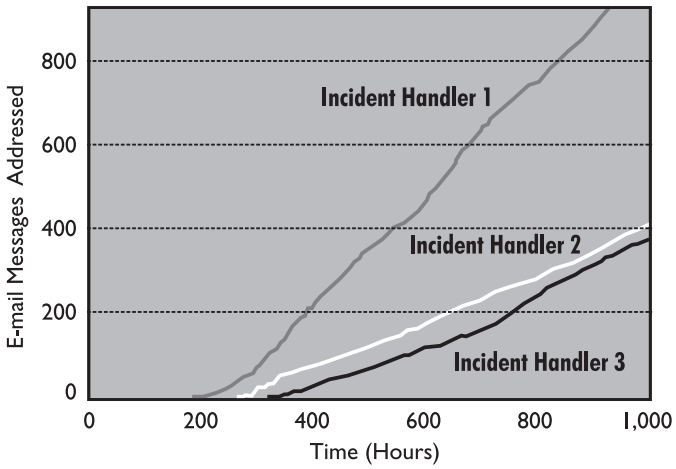


Figure 4. *The cumulative number of E-mail messages addressed by each of the incident handlers.*

Simulation-based training can be performed by individual managers who interact with the simulated software development activities. This person has control over certain control parameters (such as hiring rate and salaries), and the decisions made alter the course of the subsequent simulation history. Analysis of a training session can be performed after the session to see what went right and what went wrong in the decision-making process and to reinforce effective decision making.

To bring groups of managers to a central location for training can be both costly and time consuming. In the near future, such groups may be trained in a geographically distributed manner using a simulator with displays on the managers' local terminals. Interaction between trainees (which allows for joint decision making) can be provided through the use of collaboration technology. With the increasing interest in this technology, distributed training may soon become practical.

Process Improvement

Simulation can be used to support process improvement at all levels of the Capability Maturity Model® (CMM) but particularly at the higher levels [13,14,15]. Because simulation forces one to address metrics and process behavior in a methodical way (see "Supporting Metric Collection" section), one may argue that simulation can accelerate the introduction of process improvement. Consistent with the philosophy of the CMM, simulation capability at each CMM level incrementally builds on the simulation capabilities of the preceding levels and matches the needs of the software engineering practices at that level [16].

Traditionally, revised or new processes are improved through operational experience. This can be expensive and risky. Simulation can provide considerable insights into how a process will work prior to its implementation. These insights

can help the process designer assess alternatives and show that a specific process design performs in a manner that meets expectations. In this way, processes can be pretested, and buy-in is more likely obtained from management. Subjective criticisms are less likely, since quantitative simulation of validated models can produce specific and credible answers to perhaps hostile questions.

Architecture and Commercial-Off-the-Shelf Integration

Building complex software systems usually begins with addressing the system's architecture. Without a firm notion of how the major components of a software system interact, there is little likelihood that the system will reflect performance effectively. One would like to know early in the development cycle that such attributes as reliability, reusability, maintainability, portability, performance, and modifiability are above some acceptable level. There are complex dependencies between these attributes. For example, in improving performance, reusability might be sacrificed; or in improving portability, maintainability might require increased effort. Making trade-offs in this multidimensional space is not easy, but if they are not made at a high level of design abstraction, there is little chance they can be dealt with once coding begins. Simulation is a tool that can be used to examine some of these architectural trade-off issues [17]. Simulation can provide early insights into timing, resource usage, bottlenecking, and usability. In addition, one can rapidly gain insight into the implications of design changes by running simulations with varying independent parameters. Finally, one can assess sensitivities to parameter changes in a Monte Carlo (statistical) sense.

Product-Line Practices

Simulation makes considerable sense in the economic analysis of product lines. In particular,

> "Because product-line development involves changes in product composition and production, software size measures, such as lines of code, are not good predictors of productivity improvements. To estimate, track, and *compare* total costs of disparate assets, adaptation of other cost modeling techniques, particularly activity-based costing to asset-based software production, is needed." [18]

Some simulation tools incorporate activity-based costing such that, as entities flow through the simulated process, the cost associated with the processing of each entity at each stage can be accumulated. In this way, detailed cost predictions can be made with respect to different product-line strategies.

Risk Management

Projects are often vulnerable to risks resulting from things like requirements ratcheting, changing staff levels, funding cuts, and organizational disruptions. Simulation can help identify associated project risks early. By quantitatively predicting the consequences of alternate decisions, simulation can help design more objective, less risk-prone strategies. There also are risks associated with alternate system architectures or commercial-off-the-shelf integration strategies. By using simulation to examine the potentially complex interactions of alternate component configurations, the pros and cons of different design decisions can be identified.

Acquisition Management

Acquisition management is likely to be dependent on many of the practices described above, e.g., requirements management, project management, and risk management. Because all these practices can benefit from the use of simulation, acquisition management can, too. Specifically, simulation can help validate a contractor's estimates of costs and schedules and provide insight into the ability of the contractor's design to meet system requirements. Therefore, through the use of simulation, a project manager can predict potential contractor problems before they become reality. Simulation also has the effect of keeping the contractor honest in estimates of cost and schedule.

The subject of simulation-supported acquisition has been addressed in some detail by Walt Schacci and Barry Boehm [19, 20]. In their articles, they address the issues of how simulation can support the acquisition lifecycle. They give specific examples of potential applications and suggest that a research and development effort be established to explore issues such as virtual prototyping, incremental iterative acquisition supported by simulation, and the use of wide-area collaboratories.

Every Silver Lining Has a Cloud

As a cautionary note, it is well to remember that simulation is not a panacea. The predictive power of simulation is strongly dependent on how well the models are validated. Although many scientific and engineering fields can base their models on established physical law, organizational models have to deal with human and other less quantifiable issues. Not only is gathering data difficult when that data must come from human actors, the reproducibility of scenarios used to validate models cannot as easily be standardized as in experiments based on physical law.

Simulation is a simplification of the real world and is thus inherently an approximation. As indicated by S. Robertson,

> "It is not possible that a model is absolutely correct. Therefore, model [verification and validation] is concerned with creating enough confidence in a model for its results to be accepted. This is done by trying to prove that the model is incorrect. The more tests that are performed in which it cannot be proved that the model is incorrect, the more confidence in the model is increased." [21]

However, the usual alternative to simulation is to rely on human intuition, which Massimo Piattelli-Palmarini warns is often biased by "'mental blindspots' or 'mental tunnels' where we systematically make grave errors and get sidetracked into the wrong answer in certain kinds of problems." [22]

The State of Simulation Technology

Less than a decade ago, one could only develop a simulation by textual coding of the model. However, since the early 1990s, graphical simulation tools have become available. These tools

- Allow rapid model development by using, for example,
  - Drag-and-drop iconic building blocks.
  - Graphical element linking.
  - Syntactic constraints on how elements are linked.
- Are less error prone.
- Require significantly less training.
- Are easier to understand, reason about, and communicate to nontechnical staff.

Because of these features, network-based simulation tools allow one to develop large, detailed models rapidly. The focus thus becomes less on the construction of syntactically correct models and more on the models' semantic validity and the accuracy of their numerical drivers.

The simulation tools in today's marketplace are robust and reasonably inexpensive. Most tools cost in the range of $500 to $1,000. They therefore are accessible by organizations that wish to explore the applications described above. ◆

Acknowledgments

I thank Jim Withey, Bill Riddle, Anthony Earl, and Caroline Graettinger for their review of this article.

About the Author

Alan M. Christie is a senior member of the technical staff at the Software Engineering Institute. He actively promotes the application of process, collaboration, and simulation technologies to make software development more effective. He has extensive experience with process automation and barriers to its adoption. He published *Software Process Automation: The Technology and Its Adoption* (Springer-Verlag, 1995). He recently managed the implementation of a collaborative process support environment to meet the needs of computer security incident response teams. He also actively promotes simulation as an important element to understanding process behavior and to supporting process improvement. He has a master's degree in computer science and holds a doctorate in nuclear engineering.

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890
Voice: 412-268-6324
Fax: 412-268-5758
E-mail: amc@sei.cmu.edu

References

1. Kirby, K. and R. Sawhney, "Simulation: Shifting the Competitive Edge for the Next Generation," *MDC Update*, University of Tennessee, Vol. 6, 1997.
2. http://www.imaginethatinc.com
3. http://www.pitt.edu/~wjyst/whatissim.html
4. http://www.acq.osd.mil/ousda/speech/modeling.html
5. http://www.acq.osd.mil/te/speeches/sanders/simbasedacq.htm
6. http://www.msosa.dmso.mil/sia-sba/sba_sia_documents.asp
7. Summary of CAPI-Developed Simulations for the U.S. Postal Service, http://idt.net/~capi99/usps.htm
8. Gardner, L.L., M.E. Grant, and L.J. Rolston, "Using Simulation to Benchmark Traditional vs. Activity-Based Costing in Product Mix Decisions," *WSC '94: Proceedings of the 1994 Conference on Winter Simulation*, pp. 1050-1057.
9. Belscher, R., "Evaluation of Real-Time Requirements by Simulation-Based Analysis," *First IEEE International Conference on Engineering of Complex Computer Systems*, IEEE Computer Society Press, Los Alamitos, Calif., November 1995.
10. Lerch, F., et al., "Using Simulation-Based Experiments for Software Requirements Engineering," N. Mead, ed., *Annals of Software Engineering*, Vol. 3, 1997.
11. Kellner, M.I., "Software Process Modeling Support for Management Planning and Control," *First International Conference on the Software Process*, Redondo Beach, Calif., 1991, pp. 8-28.
12. Abdel-Hamid, T. and S.E. Madnick, *Software Project Dynamics*, Prentice-Hall, Englewood Cliffs, N.J., 1991.
13. Raffo, D.M. and M.I. Kellner, "Using Quantitive Process Modeling to Forecast the Impact of Potential Process Improvements," *Proceedings of the 10th International Forum on COCOMO and Software Cost Modeling*, Pittsburgh, Pa., October 1995.
14. Hansen, G.A., "Simulating Software Development Processes," *IEEE Computer*, January 1996.
15. Tvedt, J.D. and J.S. Collofello, "Evaluating the Effectiveness of Process Improvements on Software Development Cycle Time via System Dynamics Modeling," *Proceedings of the 19th Annual International Computer Software and Applications Conference*, 1995.
16. Christie, A. M., "Simulation in Support of CMM-Based Process Improvement," *Journal of Systems and Software* (forthcoming).
17. http://www.sei.cmu.edu/publications/documents/97.reports/97tr029/97tr029chap03.htm
18. http://www.sei.cmu.edu/plp/modeling_costs.html
19. http://sunset.usc.edu/SAMSA
20. Schacci, Walt and Barry Boehm, "Virtual Systems Acquisition: Approach and Transitions," *Acquisition Review Quarterly*, Vol. 5, No. 2, spring 1998.
21. Robertson, S., "Simulation Model Verification and Validation: Increase the Users' Confidence," *Proceedings of the 1997 Winter Simulation Conference*, pp. 53-59.
22. Piattelli-Palmarini, Massimo, *Inevitable Illusions: How Mistakes of Reason Rule Our Minds*, John Wiley, 1994. Also http://public.logica.com/~stepneys/bib/nf/piattell.htm

Note

1. In these diagrams, double lines represent the flow of things, and the single lines represent the flow of numerical data.