Dynamic Modeling of Product Development Processes

by
David N. Ford[1] and John D. Sterman[2]

Introduction

Developing products faster, better and cheaper than competitors has become critical to success in many markets whether the product is an office building, software package, or computer chip. This has made the performance of product development projects an increasingly important area of competitive advantage. In response to these pressures many industries have shifted from a sequential, functional development paradigm to a concurrent, team based paradigm. Increasing concurrence and cross functional development also dramatically increases the dynamic complexity of product development (Smith and Eppinger, 1995). But the mental models used by developers and managers to evaluate, estimate and manage projects have not generally improved to include dynamic influences on performance. The resulting lack of understanding (Diehl and Sterman, 1995; Sterman, 1994; Paich and Sterman, 1993) and inadequate decision heuristics (Kleinmuntz, 1993) have contributed to the frequently cited poor management of development projects (Womack, Jones and Roos, 1990).

Many aspects of projects influence performance including the development process, resources, project scope, and targets. To be complete such a causal dynamic project model must explicitly model and integrate the influences of all these features on performance. Three of these four features have been modeled extensively by system dynamics researchers. For example Abdel-Hamid (1984) built a system dynamics model of software development in which progress is driven by the daily workforce of software developers (resource quantity) and software developer productivity (resource effectiveness). Some attention has also been given to the secondary effects of changes in project scope (Cooper, 1980; Reichelt, 1990, Abdel-Hamid, 1984; Fiddaman, Oliva and Aranda, 1993). But little investigation of how the demands and constraints of different development processes influence project performance has been made. The general exclusion of development process structures from project models implicitly assumes that development processes have no impact on project performance. Yet other researchers have shown that the availability of work as described by the precedence relationships within and between phases is an important constraint on project performance (Wheelwright and Clark, 1992). The simplifying assumption can also lead to grossly unrealistic performance predictions under extreme conditions. For example a project model driven solely by resource constraints allows schedule and quality performance improvement as resources grow until, at the limit of infinite labor, the project is predicted to be completed in an infinitesimally small period of time. Therefore system dynamics models of product development projects need explicit process structures and their impacts on performance.

A more suitable description of development dynamics must include iterative flows of work, distinct development activities and available work constraints. The existing system dynamics models of projects which include process structures have focused on the roles of two development activities, basework and rework (Cooper, 1980; Kim, 1988). Ford, Hou and Seville (1993) expanded this approach to model three development activities (basework, rework and optional iteration to improve quality). However these models do not adequately

---

1 Associate Professor, Department of Information Sciences, University of Bergen, N-5020 Bergen, Norway

2 Professor of Management Science, Sloan School of Management, Massachusetts Institute of Technology, 50 Memorial Drive E53-351, Cambridge, MA 02142 USA

include other important development activities identified in the product development literature including quality assurance and coordination (Wheelwright and Clark, 1992).

This paper describes a product development project model which explicitly models all four performance drivers - process structure, resources, targets and scope. We calibrated and tested the model for the case of a medium-scale semiconductor product development project. We focus on the method we used to elicit, describe and gain insight into the dynamic precedence relationships which characterize development processes. Methods for collecting and modeling the information required to capture precedence relationships in dynamic project models have not previously been developed. The importance of integrating process structure with resources, scope, and targets in dynamic models of project management and future research is discussed.

The Product Development Project Model

Our model simulates the performance of a multiple-phase development project. Each phase is customized to reflect a specific stage of product development such as preparing construction drawings for building an office complex, writing software code or testing computer chip prototypes. Each phase is represented by a generic structure which is then calibrated to the particular phase of development it represents. Individual project phases interact in several ways: 1) work progress in upstream phases constrains progress in their dependent downstream phases, 2) errors inherited by downstream phases from upstream phases corrupt downstream work which must be corrected, 3) the correction of errors requires coordination between the phase that discovered the error and the phase that generated the error, 4) schedule, quality, and cost performance in individual phases influence the conformance of the entire project to the project targets.

We describe development within a project with four activities: basework, quality assurance, rework and coordination. Basework is the completion of a development task the first time. The inspection of tasks for defects is Quality Assurance. Subsequent work to correct flaws or iteration to improve quality are referred to as Rework. Coordination is the integration of the product development project among phases. An example of coordination is when designers work with marketers to refine product specifications. The model can be used to represent many different types of development projects by customizing the characterizations of these features. Customization reflects how differences in process structures, resources, scope and targets influence the four development activities.

Our model uses three features to describe the development process in a single phase: circular iteration, multiple development activities and available work constraints. Circular iteration is described with the stock and flow structure. In our model development tasks flow into and through three states: tasks Completed but not Checked, tasks Known to require Rework, and Tasks Checked and Released. Tasks are completed for the first time through the performance of Basework. They accumulate in the Completed not Checked stock. If no tasks are flawed or those flaws are not found during Quality Assurance the tasks leave the Completed not Checked stock and pass through the Release Tasks flow into the Checked & Released stock. This represents delivering tasks to the managers of downstream phases or to customers. Flawed tasks are also discovered through the Quality Assurance activity. Tasks found to be flawed move through the Find Flawed Tasks flow from the Completed not Checked stock to a stock of Known Rework. These tasks are corrected through the Rework Tasks activity and returned to the Completed not Checked stock. Flaws can be generated during both Basework and Rework. A task being reworked to correct an existing defect may become flawed during rework.

We use Minimum Activity Durations to describe each of the development activities in a specific development phase. The Minimum Activity Duration is the shortest time required to complete a task if all required information, materials and resources are available and no flaws are generated. It describes the purest time constant the process imposes on progress by answering the question "How fast could a task be completed if everything needed was available?" For example "How fast could a structural member be installed if all the best equipment and installers were available and knew where and how to install the member?"

We describe available work constraints with internal and external precedence relationships. The Internal Precedence Relationship captures the degree of sequentially or concurrence of the tasks aggregated together within a phase, including possible changes in the degree of concurrence as the work progresses. This relationship describes the amount of work that can be done by a phase based on the amount of work which has been completed by that phase. External Precedence Relationships are used to describe available work constraints between development phases. An External Precedence Relationship describes the amount of work that can be done in a downstream phase based on the percent of work released by an upstream phase. For example the testing of a prototype cannot begin until the prototype is built and the amount of detailed design work which can be completed is constrained by the amount of product definition work which has been completed and released to the detailed design phase.

We also link development phases with coordination. Coordination describes the inter-phase effects of releasing and inheriting errors. For example, detailed designers may need to meet with product architects to explain why certain product specifications cannot be met within the time and budget available, and then revise the specifications.

Model Testing

The model includes structures describing resources (labor), targets such as overall and phase-specific deadlines, and scope as well as the process structure described above. Complete model equations and documentation are available from the authors; see also Ford (1995). Validation of the process portion of the model structure was considered particularly important, since these structures are absent in the system dynamics literature. Direct structure tests (Barlas, 1989; Forrester and Senge, 1980) included discussions with developers and managers to confirm that our model structure closely reflected their process. Behavior-reproduction tests (Forrester and Senge, 1980) were also used to validate the model by comparing simulations to reference modes developed from field data. The model was calibrated to the previously mentioned Python project. Reference modes were developed from records of the Python project. Specification items were selected as the basis for development tasks in the Product Definition phase. Numbers of specification items and changes to those items in sequential versions of the product specifications were counted to generate time series data of the progression of the Product Definition phase. Due to the complexity of the Python chip the Design phase consisted primarily of writing the computer code used to lay out physical chip components in the available space. The firm provided the full source code to us for model calibration. The code was written in several blocks. Each block included a record of the original completion date and notes describing subsequent revisions. These dates and notes were used to generate time series data for the basework and rework performed on each block of code. These were then aggregated to generate time series for the Design phase. Similar approaches were used for the other development phases.

Conclusions

Capturing how development processes affect project performance differently than resources, scope, and targets by explicitly modeling those processes is a natural application of the system dynamics methodology and provides significantly improved descriptions of development team mental models and project constraints. The significant effects these process structures have on performance demonstrate the need for integrated project models which include processes, resources, scope and targets. Future research can expand our investigations of the influences of available work constraints and particularly interactions of process, resources, targets and scope on quality, cost and schedule performance. Improved descriptions of development processes in models can also improve the use of those models as learning tools to improvement of development team mental models.

References

Abdel-Hamid, Tarek K. (1984). The Dynamics of Software Development Project Management: An Integrative System Dynamics Perspective. doctoral thesis. MIT. Cambridge, MA.

Barlas, Yaman. (1989). Multiple test for validation of system dynamics type of simulation models. European Journal of Operational Research. 42 (1989), 59-87.

Cooper, Kenneth, G. (1980). Naval Ship Production: A Claim Settled and a Framework Built. In *Interfaces*, 10:6, Dec., 1980. the Institute of Management Sciences.

Diehl, Ernst and Sterman, John (1995). Effects of Feedback Complexity on Dynamic Decision Making. Organizational Behavior and Human Decision Processes, 62(2), 198-215.

Fiddaman, Thomas, Oliva, Rogelio, and Aranda, R. Rembert (1993). Modeling the Impact of Quality Initiatives Over the Software Product Life Cycle. Proceedings of the 1993 International System Dynamics Conference, Cancun, Mexico.

Ford, David N. (1995). The Dynamics of Project Management: An Investigation of the Impacts of Project Process and Coordination on Performance. doctoral thesis. Massachusetts Institute of Technology. Cambridge, MA.

Ford, David, Hou, Alex, and Seville, Donald (1993). An Exploration of Systems Product Development at Gadget Inc. Technical Report D-4460, System Dynamics Group, Sloan School of Management, Massachusetts Institute of Technology. Cambridge, MA.

Forrester, Jay W. and Senge, Peter M. (1980). Tests for Building Confidence in System Dynamics Models. *TIMS Studies in the Management Sciences*. v. 14, pp. 209-28.

Homer, Jack, Sterman, John, Greenwood, Brian, and Perkola, Markku. (1993) Delivery Time Kim, Daniel H. (1988). Sun Microsystems, Sun3 Product Development/Release Model. Technical Report D-4113, SD Group. Cambridge, MA. MIT.

Kleinmuntz, Don N. (1993). Information Processing and Misperceptions of the Implications of Feedback in Dynamic Decision Making. *System Dynamics Review*. v. 9, n. 3, Fall, 1993.

Paich, Mark and Sterman, John (1993). Boom, Bust, and Failures to Learn in Experimental Markets. *Management Science*. vol. 39, no. 12, pp. 1439-58. Dec., 1993.

Reichelt, Kimberley S. (1990). Halter Marine: A Case Study in the Dangers of Litigation. Technical Report D-4179, System Dynamics Group, MIT Sloan School of Management. Cambridge, MA.

Smith, Robert P. and Eppinger, Steven D. (1995). Identifying Controlling features of Engineering Design Iteration. forthcoming in Management Science.

Sterman, John (1994). Learning in and about Complex Systems. *System Dynamics Review*. vol. 10, no. 2-3, pp. 291-330.

Wheelwright, Steven C. and Clark, Kim B. (1992). *Revolutionizing Product Development, Quantum Leaps in Speed, Efficiency, and Quality*. The Free Press. New York.

Womack, James P, Jones, Daniel T. and Roos, Daniel (1990) *The Machine that Changed the World.* Rawson Associates. New York.