Chapter 2

Literature Review

2.1 Introduction

This chapter describes and evaluates the literature as it pertains to this research. Traditional characterizations of product development projects are described, followed by of characterizations which have appeared over the last decade. These descriptions provide the basis for the survey and evaluation of product development models for investigating the dynamic impacts of process structure on performance. This is followed by a more detailed evaluation of existing system dynamics models of projects for their contribution to this work. Finally, gaps in the current literature are identified as the starting point for the specific work of this research.

2.2 Traditional Characterizations of the Product Development Process

The traditional model of the product development process and organization is based upon a sequential and functional approach to development (Wheelwright and Clark, 1992; Zaccai, 1991). In the traditional paradigm the development process is a series of development activities from conceptualization to product introduction. An example of the traditional development process can be seen in Boeing's description of their development process for the 727-100 airliner in the 1960s (Maxam, 1978):

> "There were three distinct formal Engineering phases, following an informal series of concept studies, of the successful airplane program. Although these phases appeared to overlap and blend together this appearance was caused more by the individual scheduling of each element of the design than by the actual blending of the phases."

Boeing's description reveals their intention to keep the development phases separate and sequential. Many researchers have described the traditional process and given examples from industry (e.g., Wheelwright and Clark, 1992; Womack et al., 1990; Nevins and Whitney, 1989; Hayes et al., 1988). Clark and Fujimoto (1991b) describe this paradigm as appropriate "...when markets were relatively stable, product life cycles were long, and customers concerned most with technical performance."

Substandard project performance under the traditional paradigm generates friction among functional groups, little and poor coordination, and bottlenecks in the flow of products through the development process (Ulrich and Eppinger, 1994; Hayes et al.; 1988) . This can extend cycle times or incur additional resource use, thereby increasing costs. Increased resource use can be seen in Boeing's description of how its 727-100 program responded to changes: "...even after the Design Development Phase was underway for some time, the designer was still faced with many changes as more and more technical data was being generated, usually as the result of the testing process. It is also interesting to note that the Engineering schedules did not have the luxury of spare time, thus these late breaking changes had to be accommodated through overtime, work around, and a high dose of ingenuity of all concerned both inside and outside the Engineering organization." (Maxam, 1978).

In the traditional product development paradigm the three traditional measures of project success - time, cost, and quality/scope - are increased or decreased to improve total project performance. This can take the form of trading performance among the three measures in a zero-sum environment (Rosenau and Moran, 1993). But more mutually beneficial changes are often also available. This approach can be seen in Boeing's reduction of scope in the development of the 777 aircraft. Boeing completely avoided full-scale physical mock-ups of its 777 airplane by designing and testing with software (Stix, 1991). Gomory (1991) provides another example of reducing cycle time using scope reduction. A team of developers at IBM used previously developed standard components in the development of computer terminals instead of developing new components, thereby reducing cycle time by five months.

The Critical Path Method and PERT (program evaluation and review technique) are two traditional tools which are widely used to manage development projects. Although initially developed for schedule control they have been expanded to manage resources (and therefore costs). They are based upon the traditional paradigm of development. The Critical Path Method disaggregates the development process into activities which are related through their temporal dependencies. Each activity is treated as a monolithic block of work described only by its duration. The temporal dependencies describe the constraints which earlier (upstream) activities impose on later (downstream) activities. The constraints are described with relationships between the beginnings and completions of activities. The logic of the schedule can be represented in a network diagram. A simple example of a network diagram is shown in Figure 2-1.

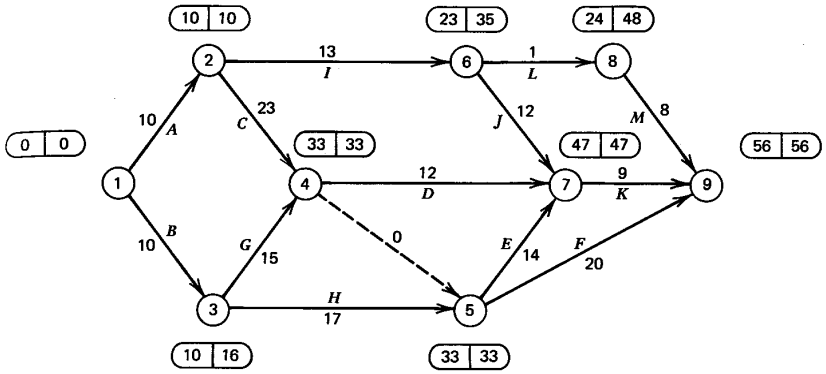

Figure 2-1: A Network Diagram for Critical Path Method Schedule Management (Halpin and Woodhead, 1980)

Critical Path Method calculations identify a project's critical path, which is the sequence of tasks whose combined durations define the minimum possible completion time for the entire set of tasks (Ulrich and Eppinger, 1994). Earliest and latest possible start and finish dates of all activities within a schedule determined by the critical path can be calculated, as can the available slack times. The results of this planning and analysis can be presented for broader communication with a Gantt chart. An example of a Gantt chart from Moder et al. (1983) is shown in Figure 2-2.

Figure 2-2: A Gantt Chart Representation of a Critical Path Method Schedule (Moder et al., 1983)

The Critical Path Method provides several tools for trading away good performance in one measure for improved performance in another. For example durations of activities along the critical path can be shortened by adding more resources (Ulrich and Eppinger, 1994; Wheelwright and Clark, 1992; Moder et al., 1983). The Critical Path method provides a time-cost-tradeoff method for analyzing the effectiveness of accelerating alternative activities. The effects of altering activity dependencies among activities to shorten the critical path can be investigated (Barrie and Paulson, 1984; Moder et al., 1983).

The Critical Path Method is easily understood and applied. It provides a set of fundamental tools for characterizing and managing a development project in temporal terms. However the method has critical limitations. The method assumes no rework of errors which are undiscovered when the phase is "completed" and that the rework of errors discovered within a phase's duration is incorporated into the phase duration estimate. The method cannot explicitly represent bilaterally coupled activities and therefore cannot describe loops, feedback, or iteration in a system. It also assumes that the development project remains unchanged over time. This prevents the method from modeling time-varying and endogenous factors such as developer skill, training, and coordination issues. Therefore the Critical Path Method is unable to model the highly coupled aspects and dynamic nature of the product development process. Finally, the Critical Path Method cannot describe the rational which underlies the structure description and therefore lacks depth of information content.

PERT (Project Evaluation and Review Technique) uses an approach to schedule management which is similar to the Critical Path Method. This method was developed for processes such as product development (Moder et al., 1983). PERT addresses one of the limitations of the Critical

Path method by incorporating the uncertainty inherent in the estimates of the durations of development activities into a scheduling tool. Three estimates of project duration are used for each activity to model the variability of durations. The PERT method calculates the probabilities of a project meeting specific schedule objectives. PERT incorporation of duration uncertainty makes it more valuable in managing less certain processes such as product development. However PERT requires lots of data and is limited in accuracy by the estimates of variability of activity durations. Like the Critical Path method, PERT cannot explicitly represent coupled loops or feedback, assumes the project is static, and cannot model causes of process behavior.

2.3 Recent Characterizations of the Product Development Process

Market and technology changes described previously and the limitations of traditional project management methods led to the development of a new image of effective product development (Nevins and Whitney, 1989; Hayes et al., 1988). Although this new image is still emerging its central features have been articulated by researchers and applied by industry. The new paradigm fundamentally alters both the product development process and organization. Researchers currently envision product development as a collection of highly coupled activities which are performed iteratively and often simultaneously by cross functional product development teams (Ulrich and Eppinger, 1994; Wheelwright and Clark, 1992; Womack et al., 1990). The dominant change in the development process from the traditional to the new paradigm is from sequential activities to concurrent activities (concurrent development). The dominant change in the development organization from the traditional to the new paradigm is from functional departments to cross-functional development teams.

2.3.1 Concurrent Development

Concurrent development's primary purpose is cycle time reduction. Concurrent development improves cycle times by planning, facilitating, and executing multiple development tasks simultaneously instead of sequentially as in the traditional development paradigm. This requires breaking each development activity into more smaller tasks and starting downstream tasks as soon as all prerequisites are available. Figure 2-3 illustrates the fundamental difference between the traditional ("Phased Approach") development process and concurrent design ("Overlapping Approach").

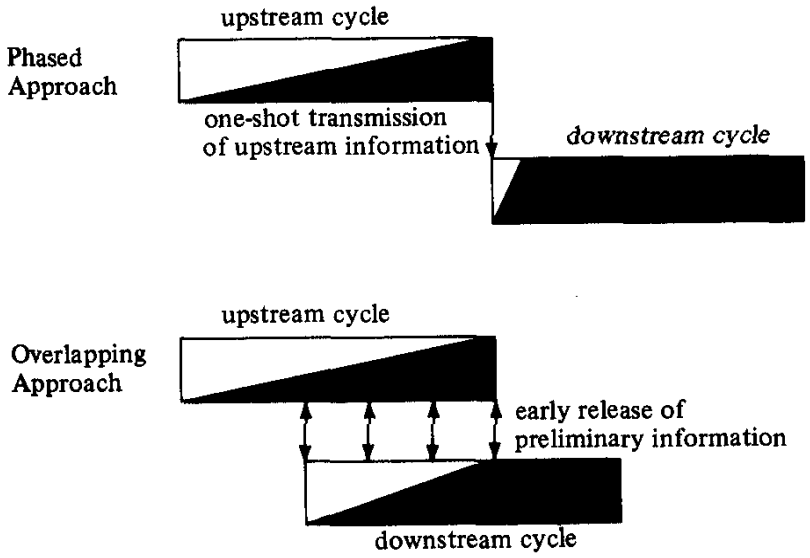

Figure 2-3: Sequential and Concurrent Product Development Processes (Hayes et al., 1988)

Boeing moved from the traditional development process used to develop the 727-100 to a concurrent approach for the development of its 777 aircraft. To apply concurrent development in the design portion of the project Boeing overlapped its fifteen previously-sequential design and engineering steps to allow work to proceed on many development activities simultaneously (Stix, 1991). The concurrent approach was also applied at the project level. For example Boeing separated the product definition phase into three tasks and began major assembly of the aircraft before the product definition was 90% complete (Peterson and Sutcliffe, 1992).

Large reductions in cycle time can be realized by applying concurrent development (Wheelwright and Clark, 1992, Womack et al., 1990; Nevins and Whitney, 1989). But the cycle time reduction comes at the cost of increased complexity. The disaggregation of development activities into smaller tasks increases the number of dependencies and the number of required information transfers. Because downstream tasks are often started on incomplete information the number of design iterations (rework) is also increased. Boeing's development of the 777 can illustrate the impacts of concurrent design. Boeing's process description for the 777 explicitly includes eleven iterative loops among nine development phases (Peterson and Sutcliffe, 1992). The additional coordination required for Boeing to use concurrent development was enormous. Boeing considered traditional development information systems wholly inadequate and therefore developed the 777 completely on computers. Coordination of the work of 4,000 engineers on the

130,000 different parts of the airliner is a major reason (Stix, 1991). As illustrated by the Boeing 777, managing increased iteration is an important issue raised by the use of concurrent development.

Steward (1981) and Eppinger et al. (1990) developed the Design Structure Matrix to investigate the iterative nature of product development. The Design Structure Matrix is a square matrix with the full set of development activities as both row and column labels. Each cell within the matrix represents a unidirectional dependency between two activities. Design Structure Matrices have been used to map (Smith and Eppinger, 1991) and predict (Morelli and Eppinger, 1993) information flows among activities. The matrix can be used to identify information flows as sequential, parallel, or coupled and for the efficient ordering of development activities. Chao (1993) applied the Design Structure Matrix to study the use of iterations in making time/quality tradeoff decisions. The focus was a portion of product development at a large semiconductor firm (DEC). Two strategies for making time/quality decisions (faster iteration and higher quality iteration) were proposed and tested.

Osborne (1993) applied iteration maps and the Design Structure Matrix to describe product development at a leading semiconductor firm (Intel) in terms of cycle time. Osborne investigated variability in cycle times. His conclusions about the impacts of iteration on cycle time are pertinent to this research. They include:

> "Iteration is a significant component of product development cycle time, typically about one third of project effort, but can represent as much as two thirds or as little as 13% of project effort.
>
> Few variables independently influence cycle time. Major project iterations are significant. Another key variable correlating with cycle time is project complexity in terms of man hours necessary to develop the product.
>
> ...Iterative modeling tools provide a means to think about the impact of changes on the total system."

Osborne's work demonstrates the need for additional investigation of the impacts of dependencies among development activities on cycle time. It also points to the need for a better understanding of how variables which impact cycle time can be identified and managed (coordinated). The Design Structure Matrix is a potentially useful tool in describing and investigating information transfer and iteration for cycle time reduction. But the Design Structure

Matrix cannot directly model the structure of a development process over time. Like the Critical Path method, the Design Structure Matrix assumes that the dependencies between phases are fixed or that the distribution is fixed. Osborne's research supports other work which suggests that iteration in product development is a primary cause of the dynamic nature of product development process (Cooper, 1994, 1993a,b,c; Ford et al., 1993; Seville and Kim, 1993; Kim, 1988). Iteration is therefore suspected to be a primary driver of cycle time performance as well as a measure of quality.

2.3.2 Cross-Functional Development Teams

A primary purpose of cross-functional teams is improved quality and effectiveness through improved coordination. Cross-functional development teams are groups of development specialists from different functional domains who work together on a single development project. The formation of cross-functional development teams is an extension of the move away from functional-based groups to the matrix structures used in the traditional development paradigm. Hayes et al. (1988) describe and Wheelwright and Clark (1992) later refine a more detailed model of this shift with intermediate steps defined by the level of influence of project managers. Restructuring product development organizations away from function-based groups and toward cross functional development teams has also become a widely used approach to reducing cycle time (Clark and Fujimoto, 1991b).

Boeing's 777 project provides an example of cross-functional development teams (Peterson, 1992; Stix, 1991). Boeing modified its matrix structure for the 777 project. Chief Engineers lead functional domains such as propulsion, avionics, structures, electrical, flight deck, and aerodynamics. They are responsible for functionality, reliability, maintainability, manufacturability, cost, and certification. Chief Project Engineers are responsible for the integration of at least one of the airplane's sixty-five individual systems. Additional Chief Project Engineers integrate these individual systems within the airplane as a whole and integrate the development project with external participants such as customers and certification testing organizations. Boeing formed over 270 cross-functional development teams within this structure. Peterson's (1992) description of the teams illustrates the cross-functional nature of their role: "These teams are defined around individual airplane systems, and are working cross-systems integration and vertical development issues (life cycle) simultaneously."

However several researchers (Bacon et al., 1994; Clark and Fujimoto, 1991b; Dean and Susman, 1991; Takeuchi and Nonaka, 1991) and many firms (e.g., see Clark and Fujimoto, 1991b, pg. 105) have realized that the formation of cross-functional teams alone does not improve cycle time. Wheelwright and Clark (1992) cite a case in which unsuccessful cross-functional teams *increased* cycle times. They identify overextended managers as a contributing factor in cross-functional team failures. Reasons cited by other researchers vary. Dean and Susman (1991) found friction between members of the team from design and manufacturing. Wheelwright and Sasser (1991) cite a lack of planning due to a lack of information. Nevens et al. (1991) identified a lack of cross-functional skills in team members and no one taking responsibility for coordination. Clark and Fujimoto (1991b) found an automobile development team consisting of only liaison people and no developers. The team failed because it was ignored by those developing the product.

The new development paradigm addresses the increased coordination needs of projects with cross-functional development teams. The apparent assumption is that cross-functional teams address the dynamic drivers of cycle time better than traditional structures. But the mixed results of applying cross-functional teams for improved project performance indicates that the use of cross-functional development teams does not adequately address dynamic aspects of development projects.

2.4 Summary of the Product Development Process Literature Review

The existing literature describes and documents recent fundamental changes in product development processes and organizations from a sequential functional approach to a concurrent team approach. In doing so it tightly links dynamic project features such as iteration with project performance. But the existing research does not described in detail or explain the relationship between dynamic features and performance. This deficit is apparent in the industry in which product development has been studied most extensively, the automobile industry. Cusumano and Nobeoka (1991) identify the need for additional research concerning coordination and cycle time in their review and critique of research of product development in the automobile industry. The current research develops an improved understanding of the relationship between dynamic project features and performance and how coordination can be used to improve performance.

2.5 System Dynamics Literature Review

Several models using the system dynamics methodology (Forrester, 1961) incorporate dynamic features into models of single product development projects. The feedback structures of system dynamics models describe the modeler's hypotheses about the dynamic behavior of the project and form a framework for describing their investigations of project behavior. The key loops abstracted from these models have been aggregated into six feedback structures.

2.5.1 Six Key Feedback Structures in System Dynamics Models of Projects

2.5.1.1 The Labor Structure

The labor structure includes three balancing feedback loops which increase labor effort as a response to schedule pressure is common in project management and central to several models of product development. The response can adjust the amount of effort (Labor Quantity), how hard people work (Labor Intensity) or both as shown in Figure 2-4.

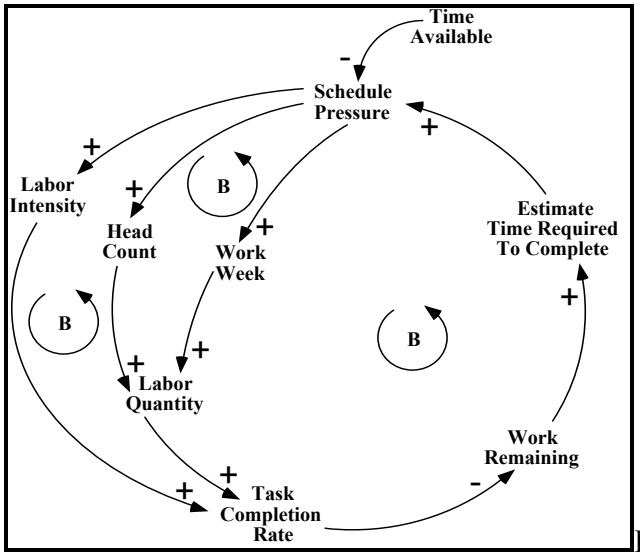

Figure 2-4: The Labor Structure

Product development models have explored these fundamental relationships more deeply in several directions. Some of the variables and relationships describe intermediate variables in the linkage between management policies and project performance. Projections of labor needs have been modeled by Cooper (1980). The impacts of Labor Quantity on Labor Intensity have been studied by Reichelt (1990). Cooper (1980) added types of labor and characteristics of labor. Abdel-Hamid (1984) used the impact of experience on labor productivity. Jessen (1990)

investigated project manager motivations with a resource-based model. Development resources other than labor (e.g. construction materials or testing machines) can be modeled with the same structure as the Labor Loops (e.g. Cooper, 1980). However existing models generally assume that labor is the dominant resource in product development projects.

The Labor Structure provides a method of describing the impacts of certain management policies. Those policies can be viewed as plans for altering the strength of certain relationships between variables which are represented by the arrows. For example, a project manager may quickly add more people to a project when it gets behind schedule. This policy is attempting to increase the strength of the causal link between the variables "Schedule Pressure" and "Headcount" and therefore increase the influence of the middle feedback loop.

2.5.1.2 The Schedule Structure

The schedule structure describes another common project management tool, slipping the deadline in response to Schedule Pressure (Figure 2-5). Many models include this feedback loop (Roberts, 1974; Richardson and Pugh, 1981; Abdel-Hamid, 1991).

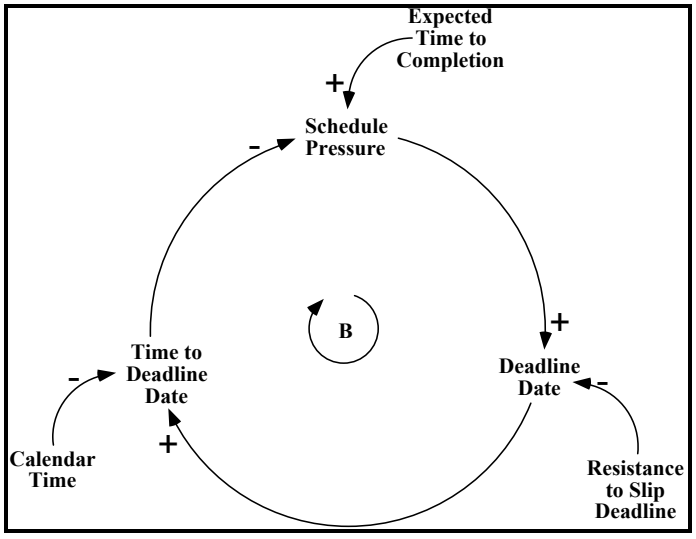

Figure 2-5: The Schedule Structure

The Schedule Structure uses a floating goal structure to describe a common feature of project management practice. In a floating goal structure the goal (the Deadline Date) drifts toward the system conditions (the Expected Time to Completion). In this example the purpose is to reduce

the Schedule Pressure. This drift is slowed by the Resistance to Slip Deadline Date, which can represent the development team's commitment to schedule performance. Project management policies can change the influence of the schedule structure on project performance by altering the Resistance to Slip Deadline Date, for example with liquidated damages which impose a cost on the project for slipping the deadline date.

2.5.1.3 The Rework Structure

Rework is the correction of errors required to make the product functional. It is distinguished from quality, which has its own loops, by the fact that rework **must** be done, whereas quality work is optional. For example, work to fix a software bug which prevents the saving of a word processing document is rework, whereas work to accelerate the saving of a word processing document augments quality. Rework is a part of all large development projects, although the amount varies widely (Cooper, 1993a,b,c). The Rework Structure describes the effects of this additional effort on a project's progress toward completion (Figure 2-6).

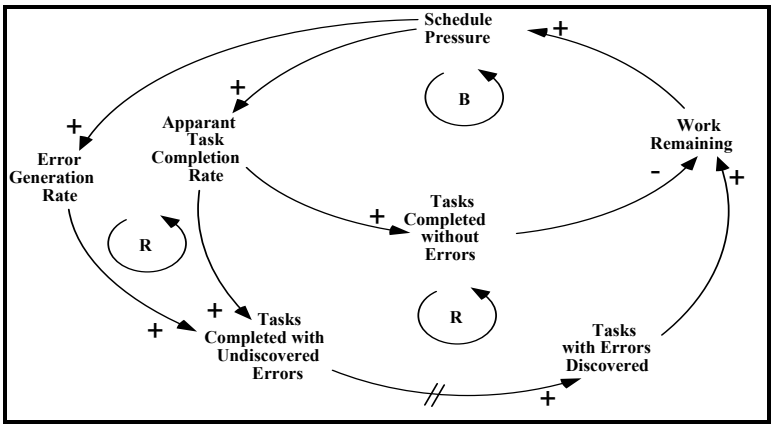

Figure 2-6: The Rework Structure

The balancing loop in Figure 2-6 represents the intended impact of a management response to an increase in schedule pressure - reduce the work remaining. The two reinforcing loops represent the impacts of the unintended side effects of the rework structure - the generation of additional errors which require correction.

Simpler versions of the Rework Structure have driven some project models (Seville and Kim, 1993; Jessen, 1990; Kim, 1988). But the separate modeling of Undiscovered Errors by Cooper

(1994, 1993a,b,c, 1980) and others has added significant new insights into the impacts of policies addressing rework on project performance. Cooper (1993a) estimates the delay in the discovery of errors to be approximately 1/4 to 3/4 of the time required to design the work the original time and concludes that this delay is one of the most important determinants of cycle time performance. Management policies can have a large impact on performance by reducing this delay, effectively weakening the influence of the lower reinforcing loop. Those policies could be rudimentary forms of communication between project phases such as quality assurance checking for errors or development team meetings. Coordination policies which increase flexibility may also weaken this loop though these relationships.

2.5.1.4 The Available Work Structure

The performance of projects can be constrained by the availability of work. Some phases of projects have internal constraints on work availability. For example, in the steel erection phase of a high-rise building project the second floor steel work cannot be done until the first floor steel work is completed. In multiple-phase projects performance can also be constrained by the amount of work released by preceding phases. For example, first floor steel erection cannot begin until the first floor steel is fabricated, which cannot begin until that steel is designed.

The limited availability of work to an individual phase of a project is the basis of some models of projects (queuing theory and the critical path and PERT methods). The constraints imposed by the available work impact the rate of basework completion through the availability of incomplete basework as shown in Figure 2-7.

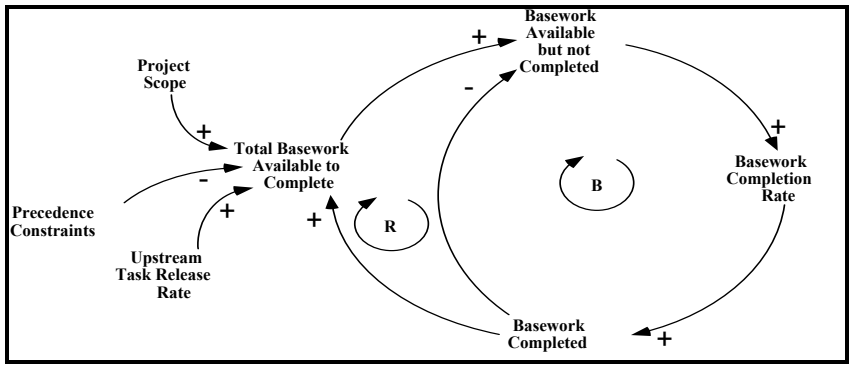

Figure 2-7: The Available Work Structure

Basework is work being performed on development tasks for the first time. The reinforcing loop in the available work structure represents the release of new work for completion due to completing work within the development phase or by an upstream development phase. The balancing loop in the available work structure represents the reduction in the basework available but not completed (the work queue) due to the completion of work. The Upstream Tasks Released and Precedence Constraints are the basis for modeling multiple interdependent phases of a development process. These precedence relationships describe the progress possible by a downstream phase based upon the progress made by the upstream phases on which it is dependent (external constraints) and the progress possible based upon progress within the phase itself (the phase's internal constraint). The critical path and PERT models are based on descriptions of these external constraints which relate the start and finish times of phases (Moder et al., 1983). Typical relationships are independent (no relationship), sequential, and concurrent with a difference in start times. Cooper (1980) describes a large system dynamics model of shipbuilding in which each development project has at least seven phases. In this model Availability of Prior Work is explicitly modeled as a constraint on Subsequent Work Progress although the specific structure is not available. Reichelt (1990) expands the description of this relationship between the engineering and construction phases and relates it to the effects of design changes. Richardson (1982) and Richardson and Paich (1981, 1980) used product development as the first of two phases in the investigation of cycle time. These models were built to replicate specific processes and are therefore very project-specific. Homer et al. (1993) subsequently developed more general descriptions of the constraints on work progress imposed by both preceding phases and the work within the phases itself, as will be described in the next section. Two models (Seville and Kim, 1993 and Ford et al., 1993) have attempted to generalize the phases of a development project. In both cases only two phases were modeled and the relationships between the phases were not generalized.

2.5.1.5 The Quality Structure

The quality structure represents the project management functions which effect the optional repetition of development tasks (Figure 2-8). The optional or flexible nature of the work reflected in the quality structure distinguishes it from the rework structure, which is required for basic product functionality. The Quality Standard reflects the level of fulfillment of customer objectives (Fiddaman, Oliva and Aranda, 1993) as well as the number of imperfections released with the product. The quality structure reflects a very real aspect of project management which is

not reflected in the rework structure: the voluntary setting of a product performance standard and adjustment of the development process to meet that standard. This can be particularly influential because project managers can influence voluntary iteration with their decisions but must perform rework.

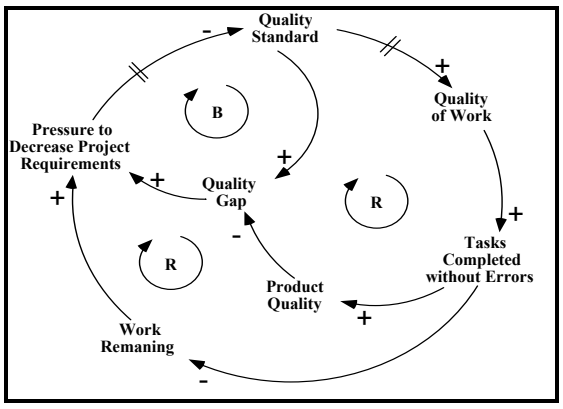

Figure 2-8: The Quality Structure

The right reinforcing loop in the quality structure represents one of the intended impacts of setting quality standards - improve product quality and raise the quality standard. The balancing loop represents the resistance to raising quality standards caused by an increasing quality gap. The lower reinforcing loop represents a second benefit of managing quality standards - the faster completion of the project.

Ford et al. (1993) separated required and voluntary iteration in product development and used a Quality Standard to drive the amount of iteration for quality, while keeping rework not optional. Higher Quality Standards resulted in longer cycle times, depicting a traditional perspective of a quality-for-time tradeoff faced by project managers. Fiddaman, Oliva and Aranda (1993) explored the evolution of the customer requirements using two of Kano's requirement types and a noise factor.

2.5.1.6 The Scope Structure

The final feedback loop is the scope structure. It represents the adjustment of project size (Figure 2-9). Examples of scope reduction adjustments which are not quality adjustments are narrowing the market and therefore product performance requirements and deleting design for manufacturability.

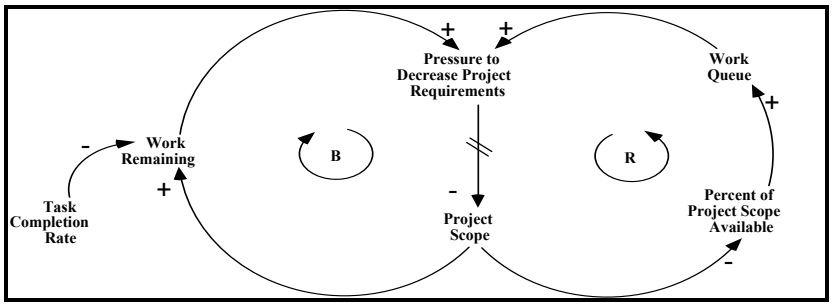

Figure 2-9: The Scope Structure

The balancing feedback loop describes the typically-intended impact of scope reduction, to accelerate progress by reducing the amount of work to be done. The reinforcing loop is typically unintended and sometimes goes unrecognized. Less scope may release work that was waiting on the now-deleted work, thereby increasing the work waiting to be done (the Work Queue). Cooper (1980) included adjustments to scope in the shipbuilding model. The dynamics of Fiddaman, Oliva and Aranda's (1993) investigation the growth of project scopes across projects, may influence a single project. Richardson and Pugh (1981) study the impact of when project scope is estimated on cycle time and manpower required.

The six feedback structures that represent existing system dynamics models of product development and the loops which result from their combinations cannot be clearly shown on a single diagram. The complexity of the combination of these loops exceeds the bounded rationality of humans to simulate or predict with any accuracy. This helps explain the difficulty in managing development projects and investigating the impacts of management policies on performance. It also supports the need for computer-based simulation models for investigating coordination.

2.5.2 System Dynamics Models of Product Development Projects

Several researchers have built system dynamics models of product development. Roberts (1974) built a relatively small (30 equations) project model which investigated the management of R&D projects. The primary material which flows through the product development process is generic "job units". Completion of job units is based upon available manpower and productivity. Management decisions are based upon perceived progress, which includes both actual progress and perceptual errors. These decisions include manpower changes, which directly impacts the job unit completion rate.

Cooper (1980) and Reichelt (1990) described the construction and use of large system dynamics models by Pugh-Roberts Associates of large scale shipbuilding operations for claims settlement. Cooper's model included product development in each of the two projects modeled. This model focused upon rework caused by customer changes. Manpower was a primary cost driver and therefore key to the model structure. The model simulates the major phases of shipbuilding operation. Cooper modeled (1980) and subsequently elaborated on (1994, 1993a,b,c) the impacts of rework in projects on cycle time. The model simulates the initial completion of development tasks (basework) and corrective action (rework). A delay in discovering defects slows the completion of unflawed development tasks. The process structure propagates change across project phases with interdependent schedules and disruptions which reduce quality and require rework. Reichelt (1990) describes the dependence of downstream project activities on the completion of upstream activities in a two-stage process. Cooper and Reichelt's work adds three valuable concepts: 1) the ability of customers to influence cycle time and increase coordination needs in product development processes, 2) the distinction between direct (first-order) and indirect (higher-order) impacts and 3) competition for resources among product development activities.

Richardson and Pugh (1981) developed and explained in detail a model focusing upon the management of single R&D projects. Richardson and Pugh use rework to expand on the Resource Effectiveness portion of the fundamental structure used by Roberts. They distinguish between work performed satisfactorily and tasks requiring rework. Both satisfactory tasks and rework are modeled explicitly. This allows the incorporation of new and potentially important influences on cycle time and coordination such as the error rate in development activities and the rate at which errors are discovered. Richardson and Pugh identify the impractical nature of some cycle time reduction policies such as assuming no rework or assuming a constant project scope. They use their model to illustrate the effects of different assumptions and policies on cycle time.

Abdel-Hamid (1984) modeled software development to better understand project management in light of cost overruns, late deliveries, and user dissatisfaction. The model simulated software production as influenced by human resources management, planning, and controlling. In this model schedule pressure influences resource quantity through the prediction of the work force size necessary to complete the project on schedule. The model's schedule pressure influences resource effectiveness through productivity, the error generation rate, and worker allocation to quality assurance.

Jessen (1990) investigated the behavior and impacts of project manager motivations with a model based upon resources, rework, targets and resource strategy. This model focused on the roles of goal seeking (balancing) feedback loops (pg. 250) in projects. It expands the description of the motivational structures in projects.

Homer et al. (1993) modeled project process structure explicitly by introducing "gate functions" to describe the constraints on work progress imposed by both preceding phases and the work within phases. This model uses graphical table functions to describe these precedence relationships in more detail than possible with the Critical Path or PERT methods. For example two phases can be described as very tightly coupled with the upstream phase limiting the work available throughout the duration of the downstream phase, not just limiting the start or finish of work as in the Critical Path method. Homer et al.'s model uses both available work and resources to constrain progress. The development process structure in the model described in this work has its foundation in the Homer et al. model.

Seville and Kim (1993) built a model based on Kim's earlier model (1988) of product development at a computer hardware company. These models simulated the flow of product development tasks through two stages: product design and process design. Seville and Kim use different levels of coordination to differentiate between "lean production" and "mass production" development organizations (as in Womack et al., 1990). They model coordination between product and process engineers with an exogenous Coordination Fraction decision. Seville and Kim contributed an explicit structure for modeling the coordination effects on resource quantity and effectiveness. They also used a two-stage aging structure and modeled the impacts of factors such as productivity to each stage separately.

Ford et al. (1993) studied the interface between two product development groups within an electronic entertainment equipment manufacturer (Ford and Paynting, 1995). The model focused on the relationships among coordination, schedule, and quality. They explicitly modeled required rework due to errors and optional iteration to meet a quality goal. This allowed them to incorporate the influences of schedule pressure on decisions about iteration for quality. This illustrates modeling cycle time reduction as a tradeoff between time and project quality. Ford et al. add a distinction between required and voluntary iteration in product development.

2.5.3 Evaluation of System Dynamics Models of Product Development Projects

The existing system dynamics literature has a rich history of modeling development projects. All these models contribute to the description and documentation of the tight linkage between development resources, resource management, and project performance. Many of these structures have been tested and applied adequately to be used as building blocks in the current work. But the current research has several important deficits.

First, the literature rarely addresses the development process directly or how it impacts project behavior. As a specific example of a deficit in the current research some features of static models of development such as the inter-phase precedence relationships used in the Critical Path Method or alternative structures have not been adequately incorporated into dynamic models of product development. Several researchers indirectly describe the tight linkage of development process structure and project performance. But the available research does not explicitly describe the linkages between development processes, policies, and their impacts, i.e. no one has proposed and tested *how* process structure and coordination impact performance. This is partially because the assumptions and specific process structures used to model the development process in the few models which include process structures are not available. The current research addresses these deficits by explicitly modeling development processes and their impacts on project performance.

Second, no current model incorporates all the significant structures developed for other systems and which apply to projects into a single model and tested their combined impacts on project behavior. This research develops such a model.

Third, current system dynamics models of projects model only a few specific project phases. No current model can be easily altered to describe many different types of projects with different numbers of phases and relationships among phases. This research develops such a model.

2.6 Summary of Literature Evaluation

The product development literature documents the shift from a sequential functional development paradigm to a more concurrent cross-functional team paradigm. It identifies the increased impact of dynamic project features such as increased feedback through many iterations and time delays. It also identifies the cross functional team as a primary tool for improved coordination for improved project performance. However the product development literature does not address in depth the cause-effect relationships within individual projects which link dynamic features to performance. The system dynamics literature investigates in depth development resource structures and their impacts on performance. But little work has been done to explicitly model the development process structures and their impact on project performance. These two deficits are addressed by the current work.