PRODUCTIVE ITERATION IN STUDENT ENGINEERING DESIGN PROJECTS

by

Ramon Costa

A thesis submitted in partial fulfillment
of the requirements for the degree

of

Master of Science

in

Industrial and Management Engineering

MONTANA STATE UNIVERSITY
Bozeman, Montana

July 2004

APPROVAL

of a thesis submitted by

Ramon Costa Ahicart

This thesis has been read by each member of the thesis committee and has been found to be satisfactory regarding content, English usage, format, citations, bibliographic style, and consistency, and is ready for submission to the College of Graduate Studies.

Dr. Durward K. Sobek II

Approved for the Department of Mechanical and Industrial Engineering

Dr. Vic A. Cundy

Approved for the College of Graduate Studies

Dr. Bruce R. McLeod

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a master's degree at Montana State University, I agree that the Library shall make it available to borrowers under the rules of the Library.

If I have indicated my intention to copyright this thesis by including a copyright notice page, copying is allowed only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for permission for extended quotation from or reproduction of this thesis in whole or in parts may be granted only by the copyright holder.

Ramon Costa

July 2004

ACKNOWLEDGEMENTS

The National Science Foundation grant entitled “CAREER: The Role of Representation in the Synthesis Process”, under Grant No. REC – 9984484, provided funding for this thesis.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

ABSTRACT

Iteration in design has different meanings, ranging from simple task repetition to heuristic reasoning processes. Determining productive iterations is important to improve the design process on cost, time, and quality, but currently there is no categorization of iterations conducive to this goal. After exploring the possible causes and attempts to address them, I propose to classify iterations as rework, design, or behavioral. The framework suggests that design teams, to improve productivity, should try to eliminate rework by increasing the resolution of design information (design iterations) without skipping design levels and by developing alternative solutions (behavioral iterations) in parallel before selecting one.

Analysis of journal data from twelve student projects helps identify design processes that achieve higher quality in less time. Factor analysis groups common variability into factors. A multivariate linear regression model of three factors explains 91% of productivity variance within the study sample. Factor scoring coefficients are then used to translate the regression model coefficients back to activities and design levels. Results indicate that generating ideas and defining the problem at a system level are the key discriminating variables between more or less productive design teams in the sample, which supports the recommendation of increasing the resolution of design information without skipping intermediate levels.

If we consider selecting an alternative for the final solution as the main design decision students make in the sample projects, then work on non-selected alternatives before selecting the final design can be used as a proxy for effort allocated to behavioral iterations. A linear model using work on non-selected alternatives shows that generating ideas at a system level relates to higher productivity while refining design details and evaluating existing design configurations associate with lower productivity. Then behavioral iteration relates to higher productivity only if alternatives are developed to the system level by generating ideas on how to address interface and configuration issues.

The framework presented in this thesis helps differentiate between productive and less productive iteration patterns and provides guidelines to prevent rework by allocating more effort in productive iteration, namely behavioral and design iteration.

CHAPTER 1

INTRODUCTION

One of the most common assumptions regarding engineering design is its iterative nature. Definitions of iteration vary greatly in the engineering design literature, but one of the most common interpretations defines iteration as the repetition of design activity. Brainstorming ideas, evaluating alternatives, and dimensioning drawings are all examples of design activity. The notion of repeating such activities connotes a feeling of lack of progress, tediousness, a grueling process of doing something over and over. Such notions would seem inconsistent with a creative process such as design.

How can design then be iterative in nature and a creative process at the same time? One answer may be that we are missing an approach to identify activity repetitions that are positive, creative, and productive, from iterations that merely reflect the grueling process of fixing errors to make an idea work. There seems to be a gap in engineering design literature in this respect that might contain a potentially useful source of improvement in engineering design practice and teaching.

A general framework to identify different iteration types may help bridge the gap. The framework should address which iterations are productive and which ones are not, and how they relate to each other. In addition, it should define each iteration type to get a sense of how these iterations could be identified in real design processes, and whether the occurrence of these iterations is a valid indicator of more productive design strategies.

Such a framework is the subject of this thesis, which builds upon an ongoing research effort directed by Dr. Sobek that attempts to better understand the role process plays in engineering student design and sets its aims at improving design learning and teaching. The scope of the overall research effort is multidisciplinary and involves the main design experience engineering students receive before entering the labor market, referred to as capstone design courses (Sobek, 2002). The present thesis explores the relation between certain design process attributes and design productivity as an outcome through an iteration framework (Costa and Sobek, 2003, 2004).

The ability to recognize more productive iterations is important because product development organizations are under increasing pressure to provide innovative products at a lower cost in shorter time. However, to simultaneously increase quality and throughput while reducing cost requires a better understanding of the underlying principles of design, such as the role of iteration. Also, from an educational standpoint, a deeper understanding of the design process will enable educators to better equip engineering graduates to work productively and thereby supply industry with more capable designers.

An Iteration Framework

As a possible means to fill the aforementioned gap in the literature, this thesis presents a framework that separates productive and less productive iterations. Iterations that are less productive are termed rework, and they include repeated patterns of activity

that produce little progress towards a completed design. Progress is charted on two dimensions: the object of the activity and the level of detail. The object of the activity includes alternatives explored before selecting a final design, sub-problems, and different sets of constraints. The level of detail refers to whether the activity is performed from a more conceptual standpoint or deals with the final details needed to complete a design, such as dimensioning drawings, setting tolerances, or material selection. Any activity repetition that contributes to the design in terms of exploration of alternatives or sub-problems, or in progression towards a detailed design could be classified in two distinct iteration types.

Repeated patterns of activity that involve work towards a fully detailed design are termed design iterations, as they provide further design details to the final solution. On the other hand, iterations that explore alternatives, sub-problems, or different sets of constraints, are termed behavioral iteration because the designer's behavior, composed of activity and level of detail, repeats but on a different object. Thus, the framework delineates three types of iteration: rework (repeated activity that changes neither design level nor scope), design iteration (repeated patterns of activity at a different design level), and behavioral (repeated patterns of activity on different problem scopes) (Costa and Sobek 2003).

The iteration framework, combined with insights from engineering design literature that address alternative exploration and selection timing, as well as level of detail explored before the decision is made (Sobek, Ward, Liker 1999; Busby, Lloyd 1999), provides some guidelines to reduce overall design time for a given quality level – which

translates to higher productivity. First, rework iteration should be reduced or eliminated, which requires an approach that deems rework unnecessary. The framework implies that specific applications of behavioral and design iterations might relate to higher productivity processes. It seems that, to avoid rework, design iterations need to be performed without skipping intermediate design levels. Also, behavioral iterations need to be performed on alternative designs before selecting one as final. As a preliminary validation of the framework, data from student engineering design projects were statistically analyzed to determine whether these propositions have validity; specifically, whether the iteration types relate to design productivity.

Empirical Validation

Design journals were reintroduced in 2000 as a required deliverable for all mechanical engineering seniors enrolled in the capstone design course at Montana State University. These journals are paper notebooks in which students annotate their design activity in real time. Students are guided in the technique of journaling but not conditioned towards specific contents to avoid a possible source of bias. The journals serve then as a teaching tool and as a data source for this research effort. They provide a rich source of data that can be used to validate the proposed theoretical framework.

To extract the data, the journals are coded using a specifically designed rubric that captures the activity and the level of detail in which the activity occurs (Sobek, 2002). Since 2000, more than 40 student design journals have thus been coded, from which this

thesis draws its data sample. The coding rubric identifies four activities and three levels of detail, referred to as design levels. Activities include problem definition, idea generation, engineering analysis, and design refinement. Other non-design activities are also coded under report writing, project management, or project presentation, but not considered in the data for the thesis. Design levels are concept, system, and detail, with system involving the configuration of the solution and the interfaces between different components of the design. A third attribute was considered for this thesis in order to capture the second dimension in which the design can evolve: scope, or the object of design activity. Alternatives explored that were not selected as part of the final design provided the workable definition of scope used to extract data from the design journals.

Students also recorded in their journals the time spent per entry, which means the amount of time per coded activity is available for analysis. The percent weekly time spent per activity at each design level provides a measure of effort allocation that can be used as an indirect measure of design iteration from which to detect patterns in design level change. These patterns are characterized by conceptual activities during the first few weeks of the project, and detail analysis dominating most of the second half of the project. System level design appears sparsely, mostly in a timeframe between the conceptual and detail dominated periods, which indicates a main change in design level from conceptual to detail work in which system level work does play an intermediate, transitional role. Analysis will indicate whether effort allocated at this intermediate system level work relates to higher productivity. If so, it would provide some empirical evidence that design iteration characterizes more productive design processes.

To investigate behavioral iterations, the percent time spent only on work done on design alternatives that were not selected as the final concept was analyzed. To determine which alternative corresponds to the final design, the data was visually inspected to see which alternative appears alone after a certain point, which corresponds to the alternative selection milestone. Work on non-selected alternatives appears by definition mainly before this milestone and constitutes work done on a scope different than the final alternative. If effort allocated to these non-selected alternatives relates to more productive design teams, then the data would provide some evidence that performing behavioral iteration before concept selection is a productive use of designer time and effort.

These techniques used a measure of productivity derived from previous work on design quality as the dependent variable. A customer satisfaction questionnaire was developed and deployed for projects to measure client satisfaction. Also, for the same sample of projects, practicing engineers were given a copy of the final report and asked to evaluate the overall goodness of the designed solution along five dimensions (Jain, 2003). To obtain a measure of productivity for the thesis, the scores each project received for client satisfaction and from the external evaluation were averaged and then divided by the total design time for the team.

The specific research questions, then, that form the focus of this thesis are:

- Are design iterations productive if system level design is not skipped?
- Does exploring alternatives before selecting one as the final design, i.e., performing behavioral iterations, relate to higher productivity?

- Were the more productive teams in the sample exploiting the advantages of these two types of iteration?
- Did the more productive teams also have less rework?

The analysis performed to answer these questions consists mainly of statistical linear models: multiple linear regression and factor analysis. Factor analysis projects the data points onto a set of coordinates defined by the principal components, which are oriented to maximize explanation of common variance among variables. The analysis outputs factor scores for each of the projects in the sample, which are independent and based on the variance common to at least two variables. The factor scores can then be used as explanatory variables instead of the original variable set. Also, some ranking techniques were used in specific instances.

Thesis Overview

The following pages present the results of this work. Chapter 2 first presents an overview of iteration in engineering design with special emphasis on definitions, causes, and means to minimize their impact found in the literature, as well as the framework to help identify necessary from unnecessary iterations. Chapters 3 and 4 then present an overview of the research methodology and the framework's preliminary empirical validation. Chapters 5 and 6 present the analysis on design and behavioral iterations respectively, and how they support the framework's implications for more productive iteration. Analysis results are then discussed in reference to project phases in Chapter 7.

The thesis concludes with a recommended effort allocation strategy for capstone design projects.

In summary, this thesis presents a framework to identify which iteration patterns are productive and which ones are not, and a series of implications that, with some empirical support, lead to recommendations on how to leverage productive iteration types to reduce the need to rework.

CHAPTER 2

ITERATION IN ENGINEERING DESIGN

Iteration, as a process attribute, is a term frequently used to describe design. It is commonly accepted that design is iterative in nature, but what do we mean by iteration? And if we are not completely sure of what iteration is, could we iterate more than needed? Can we reduce engineering time by identifying unnecessary iterations and eliminating their root cause, or by performing some iterations in parallel rather than sequentially? Exploring the nature of iteration and identifying its causes and proposed countermeasures may help improve design methods because iterations shape the outcomes of design in terms of cost, time, and quality.

After reviewing definitions and causes of iteration from the literature, I conclude that a need exists for a new categorization of iterations that helps answer the question of need. Is all iteration inevitable, or can some of it be avoided using different design strategies? Does the nature of different instances of iteration suggest new and more effective approaches to design problems?

The pages that follow briefly summarize current definitions of iteration extant in the design theory literature, and discuss the causes of iteration with approaches to minimize its negative impact. Subsequent sections present an iteration framework to help identify productive from wasteful iterations and the research methodology that provides some empirical support for the iteration framework.

Definitions of Iteration

A common approach in the literature is to consider iteration as repeating design activity (Fig. 1). For example, Ulrich and Eppinger (2000) formally define iteration as repeating an already completed task to incorporate new information (such as performing finite element analysis followed by design revision, then repeating the analysis on the revised design).

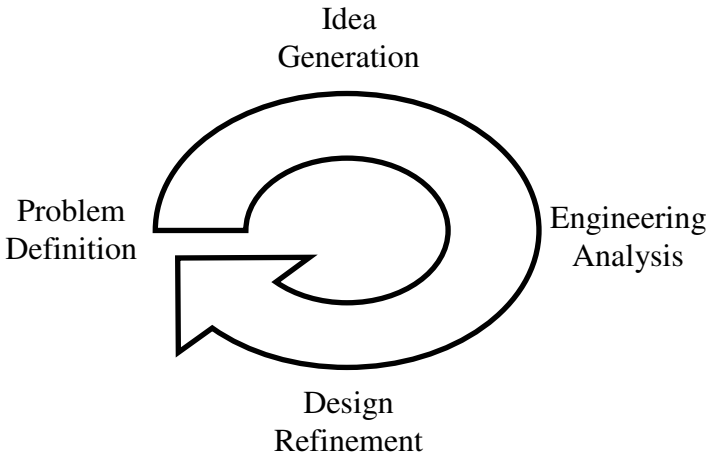


Figure 1. Hypothetical Activity Repetition Cycle.

A second approach defines iterations more broadly as a heuristic reasoning process. Adams and Atman (1999), for example, describe a broad cycle of gathering information, processing that information, identifying possible design revisions, and executing those revisions in pursuit of a goal. Adams and Atman's perspective of iteration is less concerned with repeating design activity, and more focused on the thought process that

justifies the need to perform the activity in the first place. Further expanding this definition, Ullman, Wood, and Craig (as cited in (Adams, Atman 2000)), define iterations as the cognitive processes that the designer uses when performing activities that change the design state. Emphasis is on how the designer reasons about the design, as opposed to how she performs specific activities. Thus, definitions of iteration focus on designer behavior and range from the repetition of activities to more abstract patterns of designer behavior.

Based on these definitions researchers have developed taxonomies for iteration to help better understand the nature of the phenomenon. For instance, in conjunction with the designer behavioral definitions of iteration, Adams and Atman (2000) differentiate between diagnostic iterations that define and evaluate design tasks, and transformative iterations that synthesize new information. Other approaches classify iterations based on information/task interdependence, such as Smith and Eppinger's sequential (1997b) versus parallel iterations (1997a), or Denker, Steward, and Browning's (2001) interdependent task cycles.

The definitions undoubtedly help understand different aspects related to iteration. But, at the same time, they also increase the meanings associated with the term and potentially transform it into an umbrella for all necessary and unnecessary, value-adding and wasteful, iterations. Further, because these definitions and categories were attempting to answer other questions, they contribute to the understanding of iteration but do not seem conclusive on the need to iterate.

A common implicit assumption in these definitions is that iteration is unavoidable and should be planned into design processes even though there are some drawbacks, such as increased development time. Iteration, it seems, is seen as a necessary evil in design and is not challenged. Such a perspective may systematize inefficiencies associated with unnecessary iterations.

An important assumption made in some definitions is sequential decision-making (Smith and Eppinger 1997a,b; Denker, Steward, Browning 2001). For instance, parallel iterations do not indicate simultaneous iterations, but rather simultaneous execution of interdependent or dependent tasks that influence each other's inputs and thus generate task repetition. Both the sequential and parallel categories assume sequential decision-making. In dealing with information interdependency, Denker et al. (2001) propose breaking the interdependency cycle by assuming a value for a parameter and then sequentially repeating the dependent tasks alternately until reaching a final value for the different task outputs. It is important to note this since sequential decision-making is one of the key inducers to iteration (Krishnan, Eppinger, Whitney 1997).

Causes of Iteration

The causes of design iteration stem from two sources: information characteristics and design process. Certain characteristics of information have been found to influence iteration, such as:

- New information that surfaces during design (Ulrich, Eppinger 2000),

- Stability or probability to change and information precision (Terwiesch, Loch, De Meyer 2002),
- Information interdependency that results in interdependent tasks (Denker, Steward, Browning 2001),
- Information overload due to human bounded rationality or cognitive limitations that do not allow a designer to consider all design details at once (March, Simon 1958), and
- Information that is wrong due to design error – the most obvious source of rework.

The bounded rationality argument for iteration is particularly interesting. Cognitive limitations do not allow human designers to process all relevant information at a detail level means that designers must iterate through different design levels–one cannot determine how to get to L.A. and to a specific address within it at the same time because of the different levels of detail needed. As Ullman (as quoted in Ball, Evans, Dennis 1994) indicates, it is necessary to develop solutions at different design levels because the alternative solutions are too complex for human cognition to cope with at a detail level. Similarly, designers often differentiate the design scope to make the design manageable by breaking it up into sub-problems and then performing similar sets of design tasks on each of them. Also, designers' understanding of the problem can co-evolve with the solution which can trigger changes in the (sub)problem definitions. With each change in problem definition, designers must usually repeat a set of tasks (i.e., iterate).

Iteration can also occur due to characteristics inherent in the design process. Some design processes can result in a good deal of iteration because they impose a sequential decision-making approach, as in communicating information that is precise but unstable (Krishnan, Eppinger, Whitney 1995). Or, the design team may take a depth-first approach characterized by diving into the details of the design early in the process and resolving interface issues as they arise at the detailed level (Ball, Evans, Dennis 1994). Iteration can also result from not developing the system's architecture before diving into details, or by planning design activities without taking into account information dependencies and interdependencies.

A Framework of Productive versus Non-productive Iteration

The definitions of iteration are helpful in better understanding iteration in engineering design, but do not help in distinguishing between productive and non-productive iterations. The discussion above on the causes of iteration suggests that some iteration is "necessary" in that it is inherent to design work; for example, the need to work on different design levels, or iteration resulting from increased depth of understanding of the design problem. However, some iteration would appear to be "avoidable" because, in theory, actions could be taken to reduce or eliminate it; for example, iteration due to lapses in judgment or systemic issues such as a flawed design process. It seems that identifying iterations that are inherent in design would be the first step to eliminate unnecessary iterations.

If we define iteration as the repetition of an activity, it is possible to develop three iteration types based on whether the design level or the problem scope change, as shown in Table 1. The three iteration types differ in the manner in which they contribute, or not, to design completion. Design iterations progress the design in terms of level of detail, while behavioral iterations progress the design in terms of scope of problem being addressed. I consider both these types of iteration potentially productive. Rework iteration is repetition of activity that does not move the design closer to completion because neither of the attributes changes—the designer repeats a set of tasks on exactly the same problem scope at precisely the same level of detail as an earlier effort. Most would not consider this a productive use of time.

Table 1. Iteration Type Defined by Design Process Attributes Compared to First Task Performance.

	Design Process Attributes		
Iteration Type	Activity	Design Level	Scope
Rework	Repeats	Same	Same
Design	Repeats	Changes	Same
Behavioral	Repeats	Same	Changes

To better understand the definitions of these iteration types, and what the process attributes mean, the following paragraphs describe instances of each iteration type, starting with *rework iteration* (repeating an activity at the same design level on the same object or scope). The most common reason for repeating an activity at the same scope and design level is to correct an error. For instance, when in 1996 the European Space Agency reused software from Ariane 4 in the Ariane 5 project, an undetected design error resulted in the loss of the rocket plus the half a dozen commercial communications

satellites it was supposed to set in orbit (Press release from the European Space Agency's Inquiry Board). The design work to replace the rocket would be considered rework iteration because neither scope (same mission) nor design level (a working control software) has changed; yet the design task was repeated.

Rework iteration should change the design state in the same manner in the second task execution as it did in the first, so if a designer executes the activities correctly, there is no need for this type of iteration. Rework iteration, then, should receive the same treatment as errors that produce defects in a manufacturing environment would receive: their causes should be prevented or detected at the source so errors do not become defects, or in this case iterations that cost money (Shingo, 1992).

However, if the activity's design level is not the same as in the first execution, then we consider this a sign of *design iteration*, because as activity progresses through design levels (Adams, Atman 1999) the design evolves toward the desired final state. As an analogy, consider the problem of finding directions to a specific address in Los Angeles. Consulting a map of the U.S. will provide information on L.A.'s location, but is not very helpful for finding directions within the city. Repeating the activity (looking for directions) at a lower design level, or from a lower height for this analogy, will provide different information useful to find the address of interest. An L.A. street map will ultimately help reach the address, but it will be of no use if we do not know where L.A. is. Similarly, design iteration repeats activities to generate meaningful information at different design levels. Design iteration indicates the designer is progressing through design levels, defining and refining a solution while moving from the initial concepts to a

detailed final design. The activities repeat on the same scope, but task content differs significantly as it carries the solution to a different resolution level.

Finally, *behavioral iteration* means proceeding through the same activity at the same design level, but applied to a different scope. In other words, the behavior repeats but on a different (sub)problem. Designers often divide a problem into pieces and proceed with a similar pattern of design activity performed on each of the subdivisions as object. For example, if a team works on the design of a vehicle's power train system while a different team develops the air conditioning system, the teams may perform similar design activities but on different scopes. It is especially important when the scopes correspond to alternative design options, since this exploration allows for better informed alternative selection.

Framework Implications

Such a classification of iteration helps identify necessary iterations in design. Based on the assumption of repeating an activity, rework iteration does not move the design to an increased state of completion because there is no evolution on either design level or scope, and thus might not be necessary. Design iteration represents design evolution towards completion through design levels, while behavioral iteration portrays this evolution across scope. Design and behavioral iteration are performed in tandem to evolve the design to its final state and are expected to eliminate or reduce the causes of rework iteration and thus deem it unnecessary.

In short, the iteration framework implies that increasing the resolution of design information (design iteration) without skipping design levels, and developing alternative solutions (behavioral iteration) in parallel before selecting one, helps design a quality product without the need to rework, which translates to fewer design hours for the same quality level and thus higher designer productivity.

Minimizing the Impact of Iteration

A number of authors have proposed countermeasures to minimize the adverse effects of iteration. This section discusses a number of them in light of the framework just presented.

Terwiesch, Loch, and De Meyer (2002) compared iterative approaches to design with set-based concurrent engineering approaches (Sobek, Ward, Liker 1999). They concluded that iterative approaches tend to increase both development lead time and cost (as tooling rework and engineering hours increase), but were necessary where precise but unstable information was used. Iteration could be decreased by using less precise but more stable information, but this application is limited because such designers may not be able to use the information if it is too imprecise.

To counterbalance the effects of unstable information, Krishnan, Eppinger, and Whitney (1995) define iterative overlapping as the problem of interchanging preliminary information to reduce lead time. To perform interdependent tasks simultaneously, the

authors suggest starting with value intervals (less precise, more stable information) and iteratively narrow the range until reaching a final value.

Krishnan et al. (1997) suggest an alternative countermeasure to iteration in dealing with loss of quality due to sequential decision-making, which is to decide in a way other than sequentially. Krishnan et al. (1997) present two characteristics of design parameters to be taken into account to reduce iteration: task and sequence invariance. The objective is to iterate on parameters that are dependent on both the task performed and task sequence. There are several process countermeasures that deal directly with the assumption of sequential decision-making.

Sobek, Ward, and Liker (1999) argue that Toyota's development system follows three principles that differentiate it from some of its competitors: they map the design space, integrate by interception of feasible spaces, and establish feasibility before commitment. Based on these principles, set-based concurrent engineering (SBCE) breaks from the paradigm of deciding sequentially in a point-to-point solution search. In SBCE, engineers refrain from making design decisions at a detail level before exploring alternatives at higher design levels to determine whether the solutions are feasible when considered against downstream task constraints (Ward, Liker, Christiano, Sobek 1995). The evolution along design levels to determine solution feasibility seems to indicate design iteration. At the same time, SBCE also involves behavioral iteration in that the sets of alternatives allow for the simultaneous evaluation of a range of problem definitions, which with my framework we could potentially identify as sets of scopes.

The authors also indicate that a set-based approach reduces backtracking the design, i.e., rework iteration.

Similarly, Ball, Evans, and Dennis (1994) argue that taking a breadth-first approach, in which all design activity is performed at all design levels (design iteration with behavioral iteration at each design level), reduces rework iteration because it allows for a better exploration of interactions at higher design levels (Anderson, as quoted in Ball et al. 1994). The breadth-first approach performs behavioral iteration at different design levels considering scope only as the subdivision of the design into modules. Both SBCE and breadth-first approach involve the execution of behavioral and design iterations, avoiding sequential decision-making that leads to performing behavioral iterations sequentially – which has a negative impact on lead-time, as opposed to working on different scopes simultaneously or in parallel.

Another approach to minimizing the impact of iteration is to reduce information interdependence between modules at their interfaces. The designer needs to perform system level activity in order to define the boundaries for the modules. Standardizing and making these interfaces explicit can aid in keeping the information transfer across boundaries low, as well as synchronizing or integrating the modules frequently (Cusumano 1997), thus possibly extending the limits to concurrency as Hoedemaker et al. present (Hoedemaker, Blakburn, Van Wassenhove 1995).

Smith and Eppinger (1997a) use design structure matrix to model design including strength of coupling between activities, and present advice on how to better plan activity execution based on the information obtained. In essence, the results indicate that

designers should not execute a task until all the tasks that strongly influence it are finished, and should perform long tasks later so there is less chance of repeating them. Denker et al. (2001) extend the model to limit the number of iterative cycles through better task planning and sequencing.

Problem Solution Co-evolution

Several of the recommendations from the work cited above assume that activities can be performed sequentially. This is likely not a good assumption in many cases. One reason for this is that often defining the problem and generating solutions do not occur in sequence. Rather, the definition of the problem evolves in parallel with the solution. A designer gains an initial understanding of problem, and develops a preliminary solution. But in order to evaluate the solution, he finds he needs more information about the problem. This, in turn, leads to refinements to the solution, and so the process continues, with the designer's understanding of the problem evolving as solutions are generated and evaluated for their suitability. As will be illustrated next, the proposed framework is potentially useful for identifying necessary versus avoidable iterations in this more complex situation.

Consider a common occurrence in product development: changing customer requirements (Karlsson, Nellore, Soderquist 1998). Product developers often assume that customers know exactly what they need, which creates the notion that design is searching for the right solution for the perceived customer need. Though changing requirements is quite common, planners often do not consider it a likely event and therefore do not plan

for it, so that when product specifications change, customers are blamed for triggering rework. The proposed framework helps us think about it differently.

Imagine a designer presenting the customer with a detailed design of a multipurpose knife that satisfies the customer's need as the designer perceived it, with the unstated but very real hope that the customer will accept it as-is. After reviewing the design, the customer points out that, against requirement, the blades bind when more than one tool is already unfolded. The observation results in a rework iteration to address the lack of compliance with requirements. In contrast, consider the same situation (designer presenting the customer with the newly designed multipurpose knife) but now, after seeing what a possible solution looks like, the customer decides that so much functionality is confusing and sends the designer back with a shorter list of functional requirements and an additional requirement for a more user-friendly folding mechanism to prevent injury when transitioning between tools. The designer goes back to work thinking the customer holds back information without realizing that maybe the customer did not understand the implications of the design requirements until exposed to a possible solution. The designer has to iterate but under different circumstances: in the second scenario the scope has changed and triggered a behavioral iteration.

The process of generating solutions and sharing them with the customer can trigger the redefinition of the problem (Busby, Lloyd 1999), which will drive the designer to accommodate the new information by repeating the same tasks at their corresponding design level on the modified problem definition. It is important to differentiate between the two instances of iteration because we have no control over the customer changing

requirements upon gaining new information (i.e., about the solution space), but we can avoid faults in our own design process. Failure to include information that constrains the design and is available to the designer does not trigger behavioral iteration because it is neither a subdivision of the design space nor a requirement change. For example, ignoring restrictions on materials use (e.g., lead or asbestos) and having to repeat design tasks to include this information falls in the category of rework iteration, not of behavioral iteration, and it is often used in product development to define responsibility for the increase in development or product costs due to design changes.

The proposed framework presented helps identify an alternative approach to working with customers. Instead of presenting a single, near-complete design, showing the customer alternative preliminary designs earlier in the process might help accelerate problem and solution definition, with less disruption.

While the framework seems to have validity theoretically, it would be much more beneficial to validate it empirically. The next chapter describes the overall research approach, including the data sample, variables, and analysis methods.

CHAPTER 3

OVERVIEW OF RESEARCH METHODOLOGY

To provide at least partial validation of the framework's usefulness, data from student engineering design projects were analyzed to test the implications stemming from the framework. The present chapter introduces the sample, which was collected and partly coded in prior work (Jain 2003; Sobek 2002), and provides an overview of the analysis methods.

Sample and Variables

The sample consists of twelve mechanical engineering capstone design projects from 2001 and 2002, with each 15-week project involving three to four senior-level mechanical engineering students. In total, 40 students were involved in the sample projects. All projects had a faculty advisor assigned whose role was to guide the team in its work, help the team keep on schedule and within budget, ensure that the academic requirements of the course were met, and provide technical advice when necessary. Teams could select their senior design project from a list of projects proposed by external sponsors. Design teams also interacted with the sponsoring organization, usually through a liaison that provided access to and information from the sponsoring organization. Liaisons had various levels of involvement, from program management functions throughout the project to some guidance at the beginning of the project.

Students were required to record their design activity on paper journals, indicating the date and beginning and end times of project related activities. Input to the journals was monitored to ensure students followed minimum form and content requirements. The recorded activity is limited to the paper journal, and any record of computer use has the form of printouts and explanations of purpose and outcome of the computer work. Students were free to include any design activity they considered necessary and were not conditioned to record using a particular representation. The journals were retained at semester's end. The journals from the 40 students in the sample provide more than 4,000 pages of written entries that involve almost 4,700 design hours.

Design Process Attributes

Our research on student engineering design identifies design process attributes along two dimensions, activities and design levels (Sobek 2002). The research categorizes *activities* into four areas: problem definition, idea generation, engineering analysis, and design refinement. It also distinguishes three *levels of design*, namely concept, system, and detail, to indicate the progression of design work from ambiguous to specific, with a middle step focused on the final design's configuration or architecture. Table 2 summarizes activities and design levels.

In addition to activity and design level, I included a third process attribute, *scope*, for the present study. Scope identifies the object of design on which work is performed. It relates to design modules or components, different problem definitions that are

circumscribed by different constraints, or alternatives solution approaches considered before selecting one approach and developing it to completion. Within this framework, scope refers to the (sub)problem at which a given design activity is targeted (e.g., the designer might be working on concept design for overall problem, or the concept design of a subsystem).

Table 2. Activity and Design Level.

Activity	
Problem Definition	Gathering and synthesizing information to better understand a problem or design idea
Idea Generation	Qualitatively different approach(es) to a recognized problem
Engineering Analysis	Evaluation of existing design/idea(s)
Design Refinement	Modifying or adding detail to an existing design/idea
Design Level	
Concept	Addressing a given (sub)problem with preliminary ideas, strategies, and/or concepts
System	Defining subsystems for a particular concept, and defining their configuration and interfaces
Detail	Quantifying specific features required to realize a particular concept

To illustrate the three design process attributes, consider a design team meeting to brainstorm ideas for the communications network of a motor vehicle's electrical system (e.g., using a Controller Area Network or a Local Interconnect Network). In the framework, we would classify the activity as idea generation and the design level as concept, and label the scope as communications network. If the design team partitions the network into sub-networks and defines the interactions among these sub-networks,

the activity and scope are the same but now the design level has dropped to system. Once the team partitions one of these sub-networks into components, the scope will have changed while the activities and design level (system) will remain the same. If the team considers test data to evaluate network performance, the activity would now be engineering analysis rather than idea generation.

Journal Coding

Research assistants coded the journals to indicate the activity the student performed and the corresponding design level for every journal entry (Table 3 shows the coding notation; definitions for each design level and activity were presented in Table 2). Also, the time spent per entry was recorded according to the indications found in the journal, which became the main quantitative data for analysis. Entries with times not specifically noted in the journal were assigned times based on indications from other journals, the amount of journal space used to record the entry as compared to other entries with noted times, or by using a rule of allocating ten minutes of effort per unit of space used on the journal page.

Table 3. Codes for any Activity at a Design Level.

	Problem Definition	Idea Generation	Engineering Analysis	Design Refinement
Concept	C/PD	C/IG	C/EA	C/DR
System	S/PD	S/IG	S/EA	S/DR
Detail	D/PD	D/IG	D/EA	D/DR

The principal investigator (Dr. Sobek) reviewed the coding to ensure both internal and inter-evaluator consistency. The coding and times for individual journals were entered into a database, then the times were aggregated to the project level by summing from individual journals for that project.

Figure 2 presents a sample journal entry to illustrate journal format and coding. In this sample journal entry, the initial text was coded as detail problem definition and the top sketch as detail idea generation. However, the sketch that illustrates the relation between the cutter, the potato, the sausage, and the core was considered system design refinement. It was not considered idea generation because the same idea appears earlier, only now the designer considers the layout again before exploring possible design problems at a detailed level. The notes on the sketch were coded as detail problem definition and the sketch at the bottom as detail idea generation.

In addition to design level and activity type, a third process attribute of interest is scope, which refers to alternatives, sub-problems, or different sets of constraints. Given the nature of the projects and the resolution of the journal data, scope was identified as the alternative concept the team or individual works on for each journal entry. Focusing on alternatives for the complete project also helped identify the specific solution the team selected as the final design and thus identify effort allocated to alternatives that were not selected, which measured behavioral iteration. Two people reviewed the projects in the sample noting the overall alternative the teams were working on to reduce bias. Disagreements were discussed to reach a decision for each affected entry. Design work not related to any stated alternative was coded as generic work, such as generic data-

gathering entries or design work entries without enough information to determine which alternative they belong to, unless the context made it clear. Design work applicable to more than one alternative was classified under all pertinent alternatives.

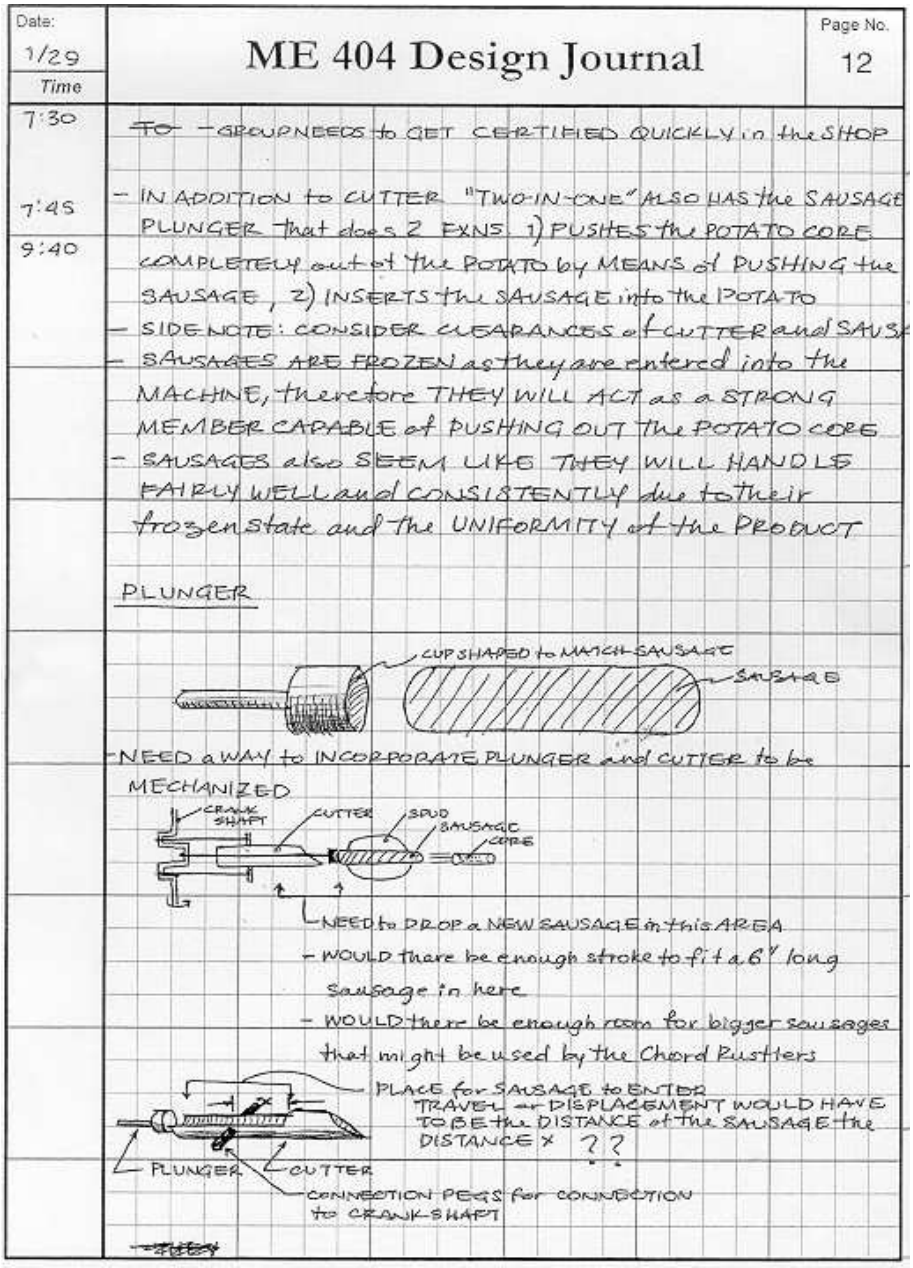


Figure 2. Sample Journal Entry

Table 4 presents an excerpt of coded data for one team member's journal. The design team considered two alternatives to address this project's design objectives and Alternative 2 was selected as the final design, which is apparent on the table as March 19th contains the last entry for Alternative 1. The rest of the project presents entries for alternatives 0 or 2. Some of the entries are allocated to all three categories.

Table 4. Sample Scope Data

Alternatives						
No Specific Alternative				0		
In-line system					1	
Horizontal system						2
Date	Level	Activity	Time			
2-Feb	Concept	Problem Definition	3.25	0		
6-Feb	Concept	Problem Definition	1.25	0		
7-Feb	Concept	Problem Definition	0.78	0		
12-Feb	Concept	Idea Generation	1.40	0	1	
13-Feb	Concept	Problem Definition	0.50	0		
14-Feb	Concept	Problem Definition	0.15	0		
14-Feb	Concept	Idea Generation	0.60	0	1	
15-Feb	Concept	Problem Definition	0.75	0		
16-Feb	Concept	Problem Definition	0.25	0		
16-Feb	System	Idea Generation	0.10	0		
16-Feb	Detail	Problem Definition	0.30	0		
22-Feb	Concept	Problem Definition	0.50	0		
22-Feb	System	Idea Generation	2.00		1	
5-Mar	System	Idea Generation	2.00	0	1	2
19-Mar	System	Idea Generation	0.50	0	1	2
20-Mar	System	Design Refinement	2	0		2
20-Mar	Detail	Engineering Analysis	0.5	0		
20-Mar	Detail	Design Refinement	0.5	0		
22-Mar	Detail	Engineering Analysis	3	0		
22-Mar	Detail	Design Refinement	0.5	0		
23-Mar	Detail	Engineering Analysis	2.75			2
23-Mar	Detail	Design Refinement	0.75			2
24-Mar	System	Idea Generation	0.75			2
25-Mar	Detail	Design Refinement	3.5			2
28-Mar	Detail	Problem Definition	1			2
28-Mar	Detail	Engineering Analysis	0.33			2
28-Mar	Detail	Design Refinement	4.5			2

Productivity Measurement

To evaluate a project's outcome, Jain (2003) developed two validated outcomes assessment instruments. Practicing engineers evaluated the final reports for each project in the sample and assigned scores using a carefully designed rubric intended to measure the overall quality of the design (*Q*). Two basic metrics (requirements and feasibility), two advanced (creativity and simplicity), and a metric of the overall impression of the design solution compose the rubric. Each metric is on a scale of 1 to 7, with 7 being the best. The quality score (Q) is the average of the five metrics across evaluators.

In addition, Jain (2003) administered a client satisfaction questionnaire and computed a customer satisfaction score (*S*) for each project using two metrics: quality and overall. The quality metric relates to design objectives and customer expectations, while the overall metric addresses feasibility of implementation, willingness to implement, and overall satisfaction with the design outcome. Each metric is on a scale of 1 to 5 (5 being best). The client satisfaction score (S) is the sum of the two measures (Sobek, Jain 2004).

The present study uses a productivity measure (*P*) calculated by:

$$P = \frac{(S+Q)/2}{\sum t} \qquad (1)$$

where $\sum t$ is the total number of student engineering hours spent on design activities in the project. Productivity measures in the sample range from 0.0012 to 0.0035 because the magnitude of the denominator is large (200-800 person hours per project) relative to the numerator (less than 10).

The data used for analysis consists of the productivity measure as the dependent variable and several effort allocation measures as predictors. The measures for effort allocation range from total time spent per activity at its design level for the entire project to percent of time for a given week allocated to a specific activity. Table 5 shows the total effort allocation per design level and activity combinations as a percentage of total design hours spent on all the projects in the sample.

Table 5. Percent of Total Design Effort in the Sample Per Activity and Design Level.

	Concept	System	Detail	Activity Totals
Problem Definition	11.71%	1.41%	8.24%	21.36%
Idea Generation	4.10%	2.45%	3.14%	9.69%
Engineering Analysis	2.73%	0.98%	23.99%	27.70%
Design Refinement	1.10%	2.84%	37.31%	41.26%
Design Level Totals	19.65%	7.67%	72.68%	100.00%

Analysis Methodology

Activity, design level, and scope help distinguish effort allocation strategies as they characterize design time. Unveiling the relation between these strategies and productivity is the role of data analysis. The data analysis approach and how it compares to other quantitative techniques previously used in the study of design processes is now presented.

A number of engineering design researchers have used quantitative approaches to better understand design processes. Some examples include Steward's Design Structure Matrix (DSM) (Steward 1981a,b), Markov chains, and fuzzy logic.

Design structure matrices have been used widely to decompose and integrate components in a design, team members in an organization, activities in a process, and parameters in design decisions (Browning 2001). Specifically, DSM has helped model design iteration (Smith and Eppinger 1997a; Yassine, Whitney, Lavine, Zambito 2000), assess the probability of rework (Yassine, Whitney, Zambito 2001), and predict system interactions (Van Eikema Hommes, Whitney, Zambito 2001). Smith and Eppinger (1997b) combine DSM with a Markov chain model to study sequential versus parallel iteration in design. The applications of DSM to engineering design and product development continue to grow.

Fuzzy logic techniques, such as the Method of Imprecision, have been used to model uncertainty in early design stages (Wood, Antonsson 1989). The method is designed to mathematically represent uncertainty in design, which helps deal with uncontrollable noise factors to achieve a more robust design, select better alternatives based on customer and designer preferences, or reduce overall performance uncertainty (Otto, Antonsson 1994; Antonsson, Otto 1995).

This study differs from previous work in three important ways. First, the approach uses data from 12 actual design projects, in contrast to design process models based on stylized versions of the design process. Some of these stylized models can identify specific activities in the process (unlike our more generic activities) and thus lend

themselves more to tools such as DSM to identify dependencies among activities. In this sample each project is different, and these specific activities are not as easily identifiable, at least to a meaningful level of resolution, so DSM was deemed inappropriate as an analysis tool in this case. Second, designer productivity is the primary dependent variable. Productivity combines quality and development cost (that is, the level of quality achieved for expended effort) as opposed to treating either measure alone. Third, unlike many studies of actual design processes, statistical analysis tools are used to gain insight into the data rather than qualitative, case-based techniques.

The three main statistical tools used to provide empirical support for the recommendations were hypothesis testing (mainly as F-tests of equal variance and t-tests of equal means), multivariate linear regression analysis, and factor analysis. Multivariate linear models explore the explanatory power of activities at design levels, factor scores, and rotated factor scores.

As part of the analysis approach, factor analysis groups common variability among variables and assigns it to factors. The factor scores for each of the projects become the predictor variables in multivariate linear regression analysis with productivity as the dependent variable. The contribution each activity has to productivity can be inferred by multiplying the standardized scoring coefficients from the factor analysis by the regression coefficients, as will be shown in detail in Chapter 5.

For the implication that behavioral iterations should explore alternatives before committing to one, factor analysis helped identify which activities were collinear and which ones were not. The model to support this implication used activities at their design

level as explanatory variables as opposed to factor scores, which was possible because individual-based data was used rather than aggregated team data. Individual-based data provides enough degrees of freedom to use activities as explanatory variables. Analysis details will be presented in Chapter 6.

Finally, ranking the projects on productivity along two dimensions helped measure timing and provided important information to implement the implications on actual student design projects. The first dimension measures effort allocation through the project's timeline and differentiates teams on whether they front loaded their effort or not. The second dimension measures the degree to which they relied on activities that indirectly measure rework on the second half of the timeline. The measures combined help identify strategies to improve the productivity of student engineering design teams through improved design processes.

CHAPTER 4

EMPIRICAL VALIDATION OF THE ITERATION FRAMEWORK

The next three chapters present analyses and results that test the three major implications of the iteration framework presented in Chapter 2. The general approach was to analyze the journal data from the sample previously described (Chapter 3) using statistical analysis tools. Journals provide happenstance data, which limits the statistical tools applicable to techniques that preclude inferring causality. The focus of the analysis was linear modeling due to the relative simplicity of the relations it uncovers, specifically multivariate linear regression and factor analysis.

Specific instances of iteration are extremely difficult to extract from the journal data. So the analysis focuses on the presence of certain kinds of activities as indicative of one or more of the three iteration types. For rework iteration specifically, as will be argued later in this chapter, identifying (and thus quantifying) specific instances of rework from the journal data was not possible. However, rework most often occurs at the detailed level, and would fall under the "design refinement" code. So the code "D/DR" is used as an indicator (though not a direct measure) of rework since presumably greater time in D/DR activities likely means greater time in rework.

Following similar reasoning, if the implication that performing design iterations without skipping design levels relates to higher productivity is valid, then we would expect that projects in which design activities span multiple design levels would associate with greater productivity. Relating that to the coding system, we would expect activities at the system-level to appear in higher productivity projects, because absence of system-

level activity suggests that design teams skip from concept to detailed design. Presence of system-level design activity is indicative that design iteration occurred without skipping levels. The analysis approach and results for whether design iteration associates with productivity are presented in Chapter 5.

Lastly, to test whether behavioral iteration associates with productivity, the analysis focuses on a key behavioral iteration—design work on alternative design concepts. But again, because extracting exact iterations from the data is very difficult, the analysis concerns overall design activity that would indicate that behavioral iteration is occurring; specifically, data only from design work on alternatives that did not make the final selection. The data would support the implication that performing behavioral iterations relates to higher productivity if design activity on these non-final concept alternatives relates to productivity. This analysis, results, and discussion are presented in Chapter 6.

The remainder of this chapter discusses the nature of iteration as observed in the design journals, particularly the nature of precedence among the design activity types identified in the journal coding, as justification for using certain codes to indicate rework. The final section of this brief chapter presents some data that indicates a negative association between indicators of rework and productivity.

Precedence in Iteration Cycles

To determine whether rework iteration is negatively related to productivity, it is helpful to identify precedence between activities and repetition of an activity cycle. The

aforementioned phenomenon of problem and solution co-evolution has implications in the interpretation of what repeating an activity cycle means. The differentiation between activities that occur on either problem or solution space allows grouping problem definition with engineering analysis, as they deal with issues on the problem space, and idea generation with design refinement because they work directly on solutions (solution space). The difference between the grouped activities is sequence: to evaluate or to modify an existing idea, the idea needs to exist; therefore both engineering analysis and design refinement can only occur after idea generation (Fig. 3).

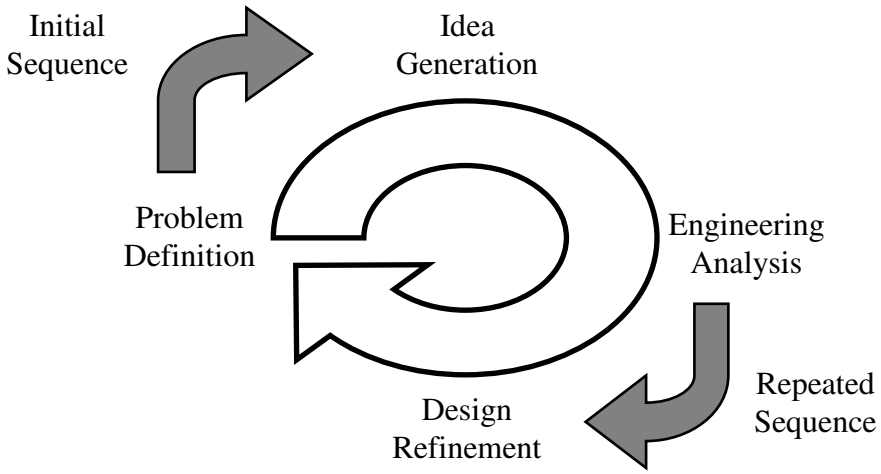


Figure 3. Sequence Repetition Embedded in Activity Definition.

Since idea generation is any qualitatively different approach to a recognized problem, problem definition needs to occur at least in parallel with idea generation. These definitions and their implications are embedded in the code used to extract information from the data, so as expected the analysis corroborates this interpretation.

But the data also show that these precedence relationships do not imply completion of the preceding task and only apply to starting points, after which the activities tend to occur in parallel. Such precedence could be interpreted as a symptom of problem solution co-evolution, since it is not a direct consequence of the definition, and might represent a different connotation to task interdependence that does not necessarily fit with design models in which activities interact only through inputs and outputs at the beginnings and ends of tasks.

Design Refinement as a Measure of Rework Iteration

The definition of iteration as repetition of an activity sequence presents a useful identification of first time occurrence in contrast with repetition. From the four process attributes, only problem definition and idea generation can occur first on any given scope and design level at the problem and solutions space respectively.

Identification of repetition in activities that transform design data is particularly interesting because of the difficulty to directly measure time spent on rework in the sample projects. Large cycles of rework are easily identified but seldom occur, whereas smaller manipulations of design data that intend to amend the design to overcome difficulties found late in the design are not as easily identified given the data's resolution. For example, a journal may indicate that the student designer is creating engineering drawings in CAD; however, it often does not indicate whether the designer is specifying the details of a design (e.g., unspecified hole diameter is specified to 0.25 in.) or changing

the details of a design (e.g., hole diameter of 0.25 in. changed to 0.375 in.). These "minor" iterations can compound to large amounts of time spent just to make the design work.

In light of this lack of resolution in the data, it was not possible to measure rework effort directly. Rather, effort allocated to detail design refinement was considered as an indirect measure of rework. Although this definition exploits the advantages of the current coding scheme it is not univocal because not all effort allocated to detail design refinement is rework – some of it is considered necessary to tune the final design. However, manipulations of design details late in the project that intend to get the design to an acceptable quality level fall mainly under the D/DR rubric (see Table 3 in Chapter 3). The more effort allocated to D/DR, the more indication that the designers are beyond tuning their design but are rather trying to fix problems derived from poor alternative selection or oversight of interactions and interfaces. The data indicates there might be an important component of D/DR that is not fine-tuning the design but rather patching it.

Most design effort in the sample projects is allocated to detail engineering analysis (24% of total design time for the sample) and detail design refinement (37% of total design hours). Therefore, around 64% percent of the sample's total design time is dedicated to activities that imply repetition and occur at the most detailed level. Detail design refinement in particular presents a strong negative association with average productivity, explaining 91% of its variance, probably because refining design details does not associate with average quality but does associate to total design time. This suggests that a large component of the 40% effort allocated to detail-level design

refinement is attempting to get the design to an acceptable quality level rather than tuning details.

If detail design refinement can be considered a reasonable though indirect measure of unnecessary repetition, it seems that rework iteration relates to lower productivity. Therefore, allocating more hours to activities that associate with higher productivity is expected to substantially reduce hours in detail design refinement or rework iteration (and thus total hours) for the same quality outcome.

CHAPTER 5

DESIGN ITERATION

One of the main implications of the framework presented in Chapter 2 is that increasing the level of detail of the design data without skipping abstraction levels is a productive form of iteration. In the context of the coded journal data, this means repeating activities at different design levels, making sure to include the system level. Thus a positive association between design activity performed at the system and productivity would provide support for the framework. The next sections present the analysis method to determine associations between activity patterns and productivity, results of analysis, and a discussion of the limitations and implications on productive design processes.

Analysis Approach

The initial step to provide evidence for not skipping design levels was to perform factor analysis on the activities at their corresponding design level. Factor analysis isolates latent constructs, which are not directly measurable and are unknown a priori, that explain the common variance among a set of variables. Using the factors as substitutes for the variables helps reduce dimensionality and allows more error degrees of freedom for subsequent analysis. Factor analysis of the 12 predictor variables in Table 5 resulted in four latent constructs. The standard scoring coefficients for each construct

were multiplied by the corresponding variable value for each project and summed to obtain one score for each factor for each project.

I then created a multivariate linear regression model to determine whether the factors can predict productivity. The independent variables are the factor scores, with team productivity as the dependent variable. Thus, each factor's regression coefficient represents its contribution to a unit increase in productivity. In order to relate the regression model back to the original variables, the regression coefficients were multiplied by the factor scoring coefficients to arrive at a productivity coefficient for each of the original variables.

Analysis Results

Factor analysis resulted in four factors that combined explain 84% of the total variance in time spent on activities at the three design levels. Table 6 presents the breakdown of variance explanation for each of the factors. Factors are independent from one another, so there is no covariance. Refer to Appendix A for the scree plot of eigenvalues, the factor pattern matrix, and the scoring coefficients matrix.

Table 7 lists the communality estimates of the twelve variables in descending order. The communality estimates represent the percentage of a variable's variance that is common to at least another activity and is explained by the four factors. The remaining variance for each variable is unique, thus not explained by the factors. The degree of communality ranges from 92 to 68%, which means these variables have well over half of

their variance common with at least one variable. Analysis was performed using the SAS System for Windows 9.0 (commercially available at www.sas.com).

Table 6. Variance Explanation by Factor.

Ranked Factors	Percent of Productivity Variance Explained
Factor 1	28.5 %
Factor 2	26.5 %
Factor 3	17.0 %
Factor 4	12.0 %

Table 7. Communality Estimates.

Ranked Design level and Activity	Percent of Common Variance
Conceptual problem definition	92
Conceptual design refinement	92
Detail idea generation	91
System engineering analysis	89
System design refinement	87
System idea generation	85
Conceptual engineering analysis	85
Detail design refinement	84
Conceptual idea generation	84
System problem definition	79
Detail engineering analysis	72
Detail problem definition	68

Each project's factor scores were derived by multiplying the matrix with the original variable values by the vector of standardized scoring coefficients. This transforms the

data set from 12 observations of 12 independent variables, to 12 observations of four latent process attributes, or factors (see the factor scores matrix in Appendix A).

As a preliminary step before regression analysis, the factor scores were analyzed to identify possible outliers that would distort the results. This preliminary analysis identified one project that obtained a score for Factor 3 that, assuming normality, lay outside of the 95% t-distribution confidence interval. The possible outlier score may be explained by the nature of the project, which involved equipment selection rather than the design of a mechanical device.

After removing this project from the sample, linear regression for all variable combinations resulted in a best-fit model that includes factors 1, 3, and 4, and explains 91.5% of the variance in productivity (see Table 8). Factor 2 had a p-value of 0.83 and was removed from the model. The intercept was set to zero because it was not statistically significant, which added an error degree of freedom.

Table 8. Multivariate Linear Regression Statistics for Design Iteration.

R^2	0.9153				
Adjusted R^2	0.7691				
Standard Error	0.0025				
Observations	11				
ANOVA	Degrees of Freedom	Sum of Squares	Mean Square	F	P-value
Regression	3	0.00056	2E-04	28.84	0.00026
Residual	8	0.00005	6E-06		
Total	11	0.00061			
	Coefficients	Standard Error	t Stat	P-value	
Factor 1	-0.003807	0.000145	-26.30	4.8E-09	
Factor 3	0.001474	0.000089	16.56	1.8E-07	
Factor 4	-0.001413	0.000093	-15.20	3.5E-07	

The extremely low p-values indicate the coefficients differ significantly from zero at any reasonable confidence level. Also, the high R-squared value suggests the model is an excellent fit for the data. The model has 8 error degrees of freedom, which is satisfactory given the small sample size but is still under the 10 considered a minimum acceptable. The regression coefficients indicate the contribution of one factor to a unit increase in productivity while holding all others constant. The coefficient's sign indicates whether the factor relates to increased or decreased productivity.

Productivity Coefficients

To translate the factor coefficients back to the twelve activities at their design level, I multiplied the regression coefficients of the three remaining factors by the standardized scoring coefficients from the factor analysis. The resulting productivity coefficients shown in Table 9 indicate the expected contribution from a given variable to a unit increase in productivity holding all others constant.

The data in Table 9 indicate that generating ideas at the system level strongly relates to increased productivity, far more than any other activity, with defining the problem at the system level following at about one-third the contribution. Only two other activities contribute positively to productivity: detail level problem definition activity and concept level analysis. The rightmost column will be addressed in the chapter on project phases.

Table 9. Productivity Coefficients by Design Level and Activity.

Design Level and Activity	Productivity Coefficient	Characteristic Phase
System idea generation	0.0031	Transition
System problem definition	0.0011	Transition
Detail problem definition	0.0005	Transition and Back End
Concept engineering analysis	0.0004	Transition
System design refinement	0.0001	Transition
Detail engineering analysis	0.0001	Back End
Detail design refinement	0	Back End
Concept idea generation	0	Front End
Concept problem definition	-0.0001	Front End
System engineering analysis	-0.0010	Transition
Detail idea generation	-0.0012	Transition and Back End
Concept design refinement	-0.0019	Transition

At the other end of the scale, refining the design at a conceptual level is associated with lower productivity. This seems to indicate that, rather than spending time refining the concept, designers that improve the state of information (Hazelrigg 2003) by transitioning to the system level — that is, increasing the amount of detail on interfaces and product structure — achieve quality designs in less time. System level engineering analysis and detail idea generation also relate to decreased productivity, but to a lesser degree (Costa and Sobek, 2003).

In the middle are activities that seem to have little effect on productivity. These are: conceptual level problem definition and idea generation, detailed level engineering analysis and design refinement, and system level design refinement.

Limitations and Model Adequacy

While the variable values represent a great deal of data (for instance, 12 data points represent 400-500 pages of journal data and thousands of hours of student work for one project), the number of projects in the sample relative to the number of variables of interest results in few degrees of freedom, which might distort the results. The adequacy of the model will indicate if there are other aspects that affect the validity of the model obtained.

The assumptions of normality, independence, and constant variance need be addressed. While the original activities at their design level are dependent, factor analysis is a valid remedial measure because factors are orthogonal. At any rate, collinearity does not affect the predictions of the model as long as the data used to make inferences follows the same multicollinearity pattern, which would probably be the case as the data used would come from another coded project (Neter, Kutner, Nachsteim, Wasserman 1996).

This model does not seem to violate the assumption that residuals are normally distributed around a zero average (Fig. 4). A correlation test for normality does not provide evidence to reject the null hypothesis that the residuals are independent and identically distributed following a Normal distribution. The correlation value between the empirical quantiles (the residuals) and the theoretical quantiles is 0.977 which is higher than the critical value of 0.928, given a sample size of 11 observations and a confidence level of 0.05.

Studentized residuals were used to detect possible patterns and outliers, considering the values of each residual with respect to its own variance to give recognition to differences in sample in the sampling errors of the residuals (Neter et al. 1996).

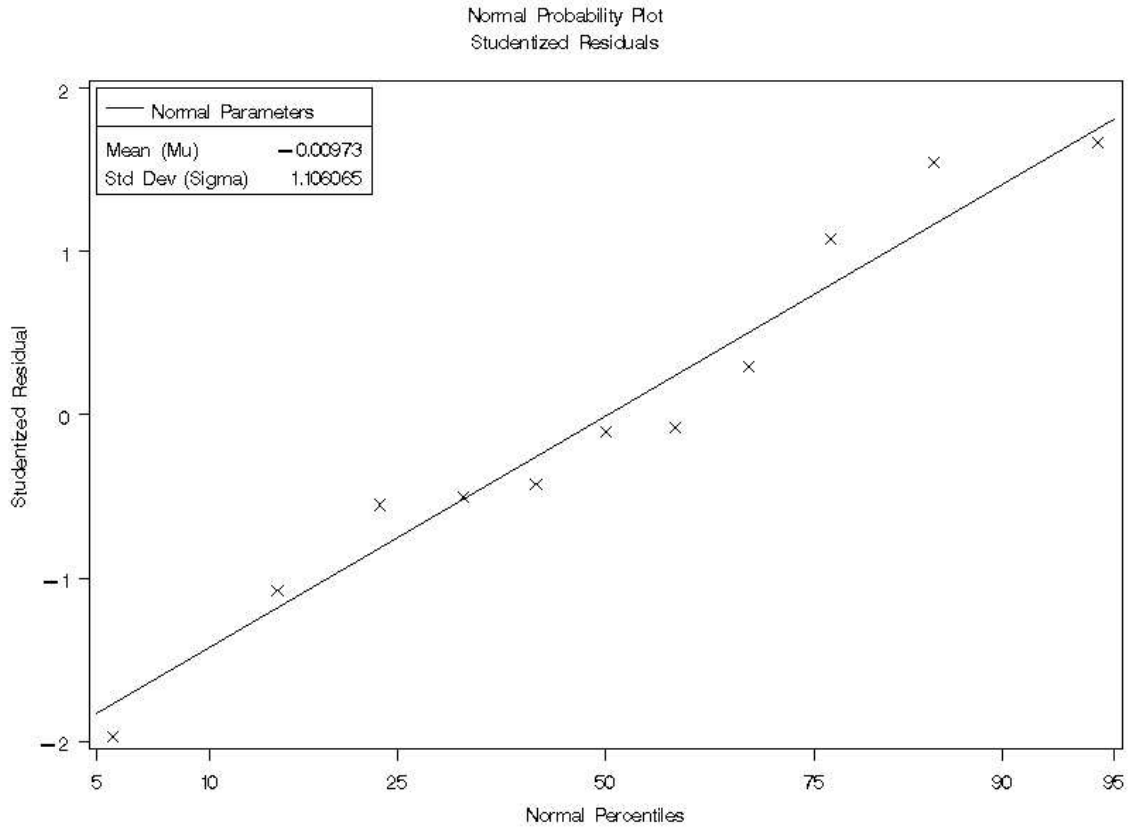


Figure 4. Normal Probability Plot for the Studentized Residuals for the Design Iteration Model.

The same applies to the assumption of constant variance. There is no apparent pattern in the graph and therefore no reason to believe the model is not a good fit (Fig. 5). A modified Levene test, which can detect if the variance of errors increases or decreases with the explanatory variable, showed no reason for concern. It should also be noted that

the Mallow's Cp value for this three-variable model is 3.0365, which indicates the model presents low bias.

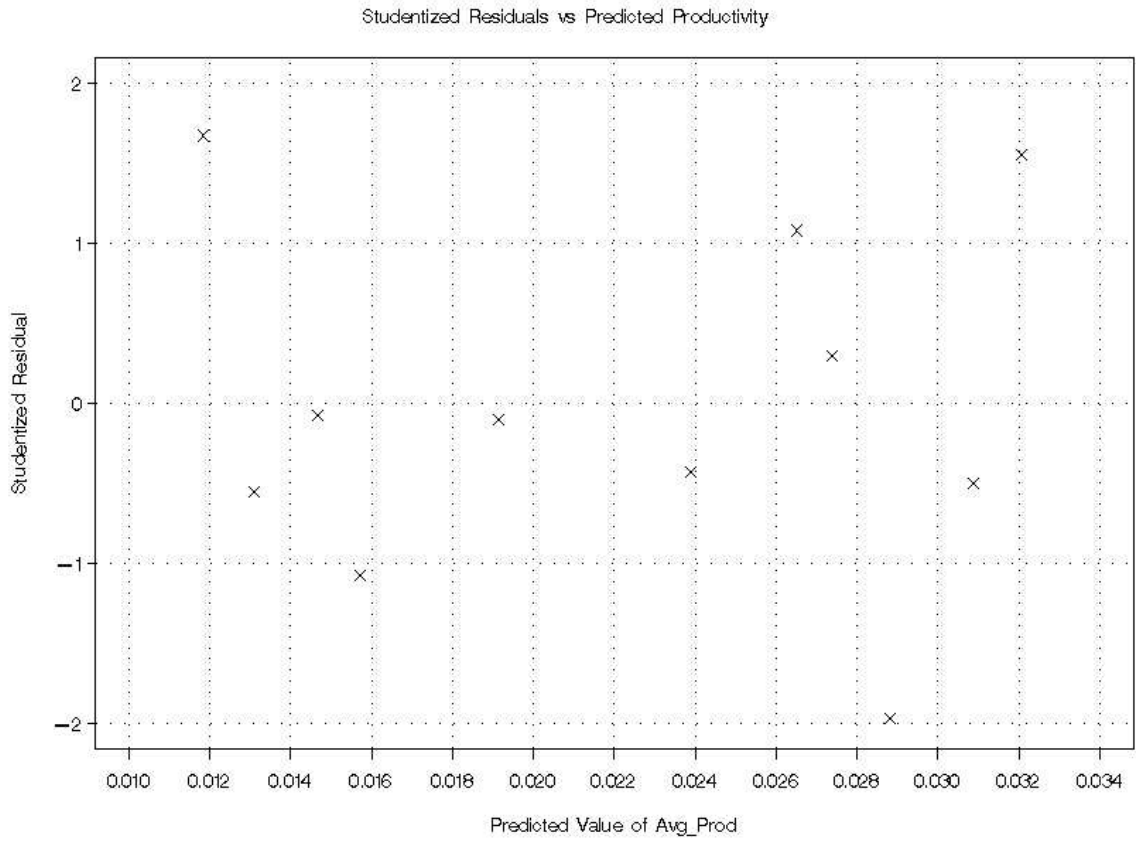


Figure 5. Studentized Residual vs. Predicted Productivity for the Design Iteration Model.

Since both assumptions are respected, the model can be assumed a good fit for the data and that the most relevant limitation is that the number of error degrees of freedom is relatively low. In order to address the limited error degrees of freedom a second model was generated now using individual data rather than the team aggregates. The following section further explains this model.

Model Based on Team Member Data

The model using team member data instead of the team aggregates was also constructed using factor scores. As in the team aggregate data, three factors were significant (though their compositions vary from the 3 factors in the team data model). An outlier was detected and removed out of the 34 observations leaving 29 error degrees of freedom. These three factors explain 74% of the variance in productivity (Table 10).

Table 10. Multivariate Linear Regression Statistics for Design Iteration on Team Member Data.

R^2	0.7442				
Adjusted R^2	0.7178				
Standard Error	0.0040				
Observations	34				
ANOVA	Degrees of Freedom	Sum of Squares	Mean Square	F	P-value
Regression	3	0.00135	0.00045	28.13	<0.0001
Residual	29	0.00046	0.00001		
Total	33	0.00181			
	Coefficients	Standard Error	t Stat	P-value	
Intercept	0.02049	0.00069622	29.44	<0.0001	
Factor 1	0.00452	0.00070283	6.44	<0.0001	
Factor 3	-0.00273	0.00069642	-3.92	0.0005	
Factor 4	-0.00371	0.00069679	-5.33	<0.0001	

The normal probability plot for this model shows that there is no reason to suspect the normality assumption is violated (Fig. 6). However, the plot of studentized residuals against the predicted values seems to show evidence of non-constant variance. Variance tends to increase in the middle range of productivity, with an almost symmetrical pattern around the 0.022 productivity score. This “tulip” pattern could indicate that there is an

additional aspect related to productivity that the model does not include (Fig. 7). The small sample size for the team aggregates does not allow us to determine whether this missing aspect also applies to the previous analysis.

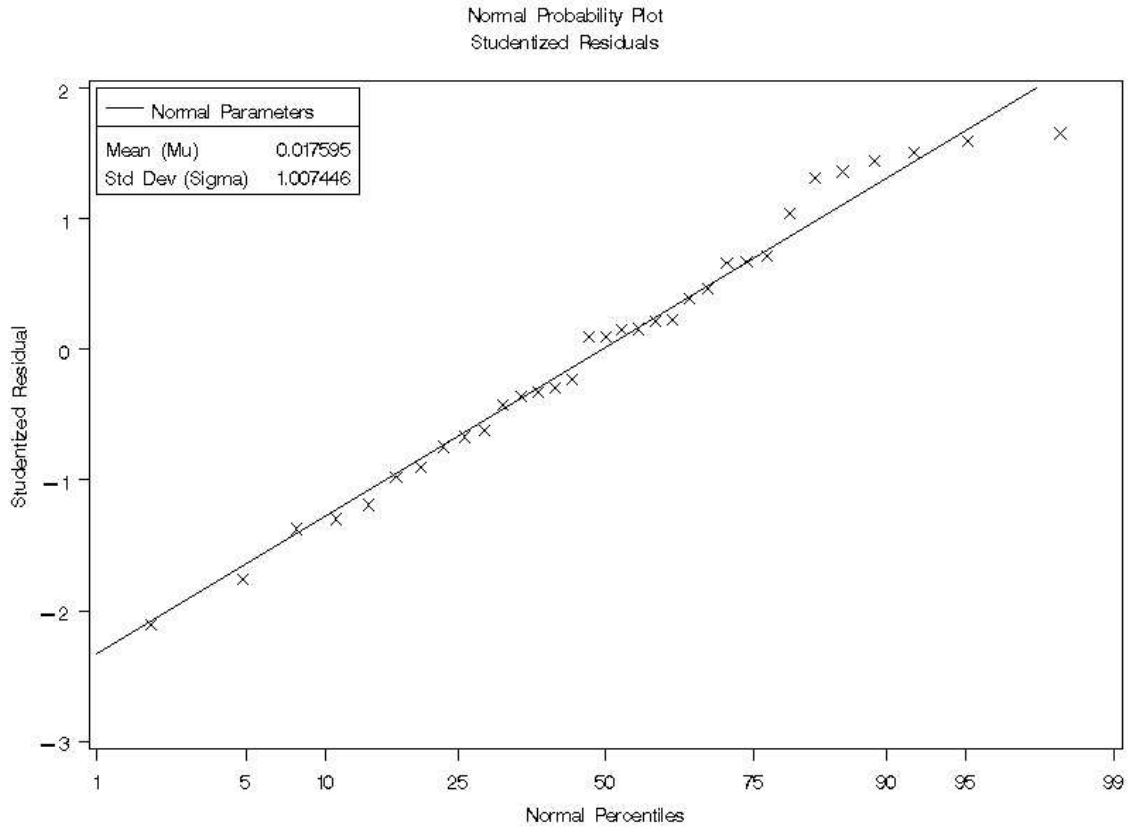


Figure 6. Normal Probability Plot for the Studentized Residuals for the Design Iteration Model Using Team Member Data.

Due to the different composition of these three factors and the different values for their linear coefficients, the resulting productivity coefficients vary from the team aggregate analysis. Table 11 shows the main productive activity is defining the problem at the conceptual level.

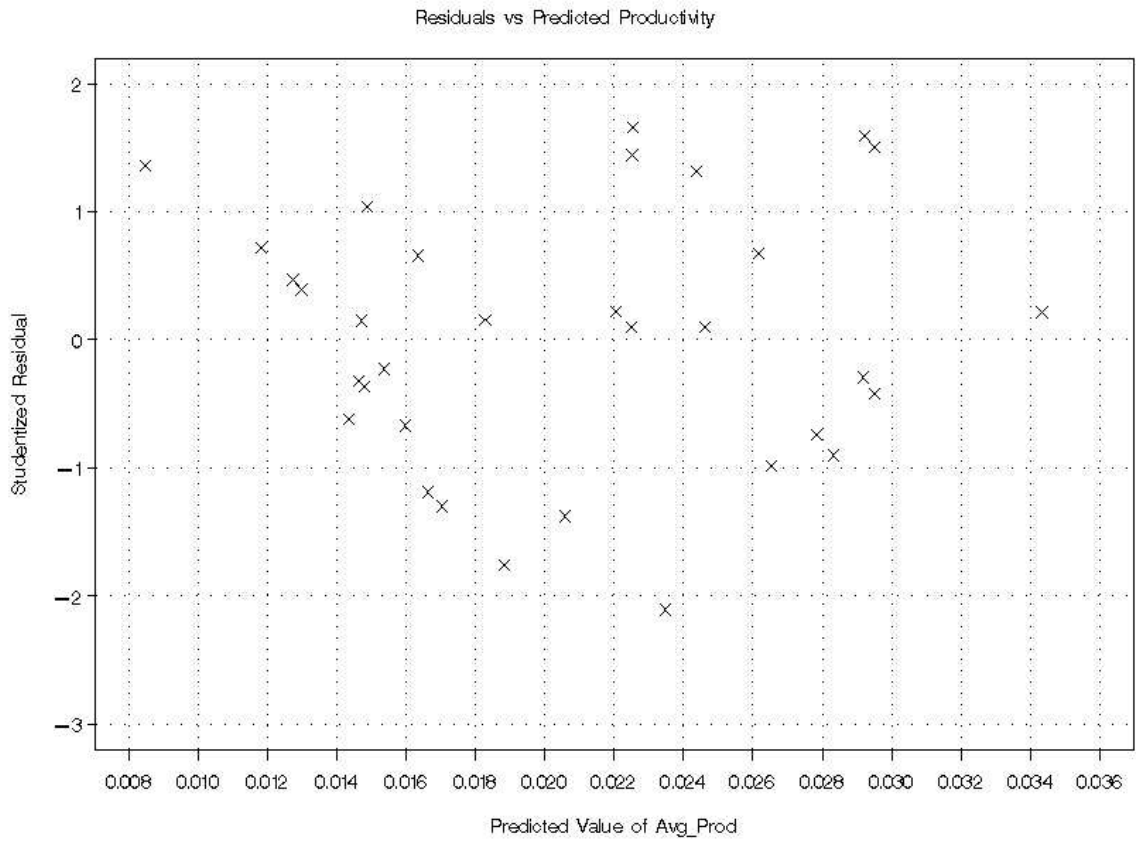


Figure 7. Studentized Residual vs. Predicted Productivity for the Design Iteration Model Using Team Member Data.

The second activity at its design level that relates to higher productivity is generating ideas at the system level (same code ranked highest contributor to productivity in the previous analysis on team aggregated data). Other activities have lower contributions to productivity but overall relate to it positively. The only activity that relates negative, by a substantial amount, is engineering analysis at the system level (again, this same code appeared as a negative contributor to productivity in the previous analysis on team aggregated data).

Table 11. Productivity Coefficients by Design Level and Activity for Individual Data.

Design Level and Activity	Productivity Coefficient	Characteristic Phase
Conceptual problem definition	0.0035	Front End
System idea generation	0.0024	Transition
Concept engineering analysis	0.0014	Transition
Detail idea generation	0.0010	Transition and Back End
Detail problem definition	0.0007	Transition and Back End
Concept idea generation	0.0006	Front End
System design refinement	0.0005	Transition
System problem definition	0.0005	Transition
Detail engineering analysis	0.0004	Back End
Concept design refinement	0.0001	Transition
Detail design refinement	0	Back End
System engineering analysis	-0.0019	Transition

Chapter 7 addresses the timing of design activity in the sample and defines three phases for the 15-week design projects: front end, transition, and back end. At this point it is worth noting that results from the team aggregated data analysis indicate that the transition phase is the distinguishing factor in a design team's productivity. Neither front nor back end activities, with the exception of concept problem definition, associate with productivity, probably because the data does not present much variability across the sample of projects in these phases. The processes differ the most at the transition phase. This observation changes for the analysis done at the team member level, where the activity that dominates the front end of the project becomes the most strongly associated with productivity, with the back end activities relegated to a small explanatory power.

In summary, the data provide some empirical support to the framework's implication that performing design iterations without skipping design levels (system level work) relates to higher productivity processes.

CHAPTER 6

BEHAVIORAL ITERATION

Behavioral iteration refers to repeating activities on different scopes. One manifestation of behavioral iteration is development work on alternative designs before selecting one. The iteration framework presented in Chapter 2 implies that performing behavioral iterations on non-selected alternatives serves as an exploration of the design space that is expected to improve the state of information (Hazelrigg, 2003) before making a decision, i.e., selecting an alternative, and thus reduce the need to revisit the decision or have to rework the design because of poor alternative selection.

Design texts generally recommend developing design alternatives in parallel as opposed to sequentially (i.e., one alternative after another) because of the implications for lead-time. The data sample also presents patterns that closely resemble parallel rather than sequential exploration of alternatives. The focus on this thesis is, however, the relation between alternative exploration and productivity. To quantify behavioral iteration, the data was stratified according to what the team was working on (scope) and then each stratum was analyzed individually, with emphasis on design work related to alternatives that were considered but not selected, as presented in the following section. If the data present a linear relation between the activities, or associated factors, and productivity for this scope, then the data would provide some empirical evidence of the implication's validity.

Analysis Approach

Three categories stratify the data based on scope: effort on no specific alternative, which includes all preliminary work done before or during the development of actual alternatives; effort on the chosen alternative, which also appears throughout the project; and work performed related to alternatives that were not selected. This last category represents behavioral iteration performed before selecting an alternative because similar activities were performed on these alternatives as were performed on the selected alternative. Thus activities are repeated but on different scopes. The results from analysis on this stratum will substantiate the claim that behavioral iteration is part of a productive design process.

Data were considered both at the team aggregate and at the individual designer level. Teams in our sample tend to divide design work and coordinate their parallel efforts, but this division mostly involves scope rather than design level or activity. While a designer might agree with the team what to work on, it is seldom agreed at what design level and which activities the designer should perform – the designer's *behavior* – as they are left for the individual to decide. It could be argued that the lack of coordination of designer behavior yields different overall results depending on how the effort is allocated. For example, if one designer spends four hours defining a problem at a system level, the overall contribution to the project would be four hours; if, however, four designers spend one hour each defining the same problem at the same level, the lack of coordination might net a contribution to the project of little more than one hour. The differences in effort allocation and the nature of the coordination in the sample makes individual data a

possible interesting perspective to better understand the relation between designer behavior (activities at design levels) and team productivity. Considering the data as individual journals also increases the number of degrees of freedom in the analysis. Classifying the data in three strata reduces the amount of extractable information in the sample when only considering teams and looking at twelve activities at their design level, because few teams spent time on all activities on non-chosen alternatives. Analysis was performed on team aggregated data to determine if there are differences between teams allocating time to a specific activity as opposed to not allocating any effort to it at all. Although some activities have only a few people spending time on them, the assumptions of non-constant variance and lack of normality of the response variable are not violated, as will be discussed in the limitations section of this chapter.

The first analysis on the data is linear regression with each team member representing a data point that consists of the percent of total design time spent at a particular activity and the average productivity score of the team. Productivity scores were not corrected per person to avoid introducing bias in the data.

The second step involves factor analysis to determine each individual's factor scores and rotated factor scores. These scores can then be used as explanatory variables for further linear regression analysis. Also, factor patterns allow us to see which activities vary in the same axis and whether they vary in the same direction. This means that individuals spend effort proportionally on certain activities, and at the same time allocate effort on other activities in an inversely proportional manner – if effort is high on one activity, it is low on another. This also allows us to determine whether the activities that

have coefficients that differ significantly from zero for the first model are linearly dependent or not – if they appear in different factors they do not relate in the way they vary, therefore collinearity should be low. Linear models on the three sets of variables (activities, factors, and rotated factors) explain some of the variance in productivity, as presented in the results section.

Analysis Results

Analysis on data aggregated to the team level did not yield any discernible patterns, so the following results are from analysis using the journal data at the individual level. The initial multivariate linear regression of activity, design level combinations on non-chosen alternatives versus team productivity, resulted in four significant variables at $p < 0.05$, as shown in Table 12. Four variables plus the intercept explain 56% of productivity variance. The variables with coefficients that differ significantly from zero ($p < 0.05$) are system idea generations and engineering analysis, conceptual problem definition, and detail design refinement. Only system idea generation has a positive coefficient, while the other three are negatively related to average productivity. It should be noted that, unlike the analysis on design iteration where design work was not stratified by scope, conceptual problem definition and detail design refinement both have a negative effect on average productivity. Assuming the model is valid, it indicates that work performed only on non-final alternatives explains 56% of the variance in productivity. The model can

predict more than half of the variance in team productivity *without* looking at the effort allocated on developing the final solution.

Table 12. Multivariate Linear Regression Statistics for Behavioral Iteration.

R^2	0.5611				
Adjusted R^2	0.5005				
Standard Error	0.00532				
Observations	33				
ANOVA	Degrees of Freedom	Sum of Squares	Mean Square	F	P-value
Regression	4	0.00105	0.000262	9.27	<0.0001
Residual	29	0.00082	0.000028		
Total	33	0.00187			
	Coefficients	Standard Error	t Stat	P-value	
Intercept	0.02924	0.00304	9.63	<0.0001	
S/IG	0.09711	0.03314	2.93	0.0065	
C/PD	-0.08347	0.02938	-2.84	0.0081	
S/EA	-0.08729	0.03488	-2.50	0.0182	
D/DR	-0.01629	0.00446	-3.65	0.0010	

Factor analysis on the variables concerning only the non-chosen alternatives resulted in six factors that explain an accumulated 84% of the variance in the original variables. From these six factors, two have regression coefficients that differ significantly from zero (Factor 1 and 2), whose model only explains 27% of the variance in productivity (see Appendix B for the scree plot of eigenvalues, the factor pattern matrix, and the scoring coefficient matrix).

Factor 1 varies in the same direction but in opposite sign to detail design refinement. Since the regression coefficient for Factor 1 is positive, the association between detail design refinement and average productivity is negative. Factor 1 presents a relatively strong factor pattern or relation with concept engineering analysis and both concept and

system problem definition, all related to higher productivity. On the other hand, Factor 2 has a negative regression coefficient, then system design refinement for this data stratum relates with higher productivity while concept problem definition and idea generation relate with lower productivity (Fig. 8).

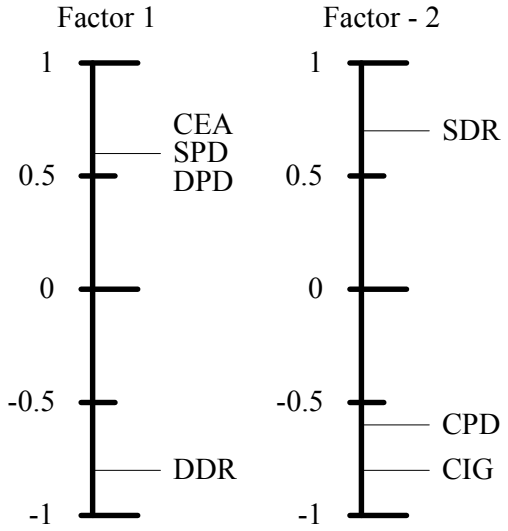


Figure 8. Factors 1 and 2 for the Non-chosen Alternative with Related Activities.

Rotating the six factors alters which have significant non-zero coefficients. Factors 1 and 6 explain only 28% of productivity variance, but the factors still shed light on the collinearity among variables. Factor 1 is related to both system and detail problem definition and system design refinement and explains around 18% of the variance in average productivity. Factor 6 clearly represents system engineering analysis, and relates with lower productivity, accounting for the remaining 10% of the variance (Fig. 9).

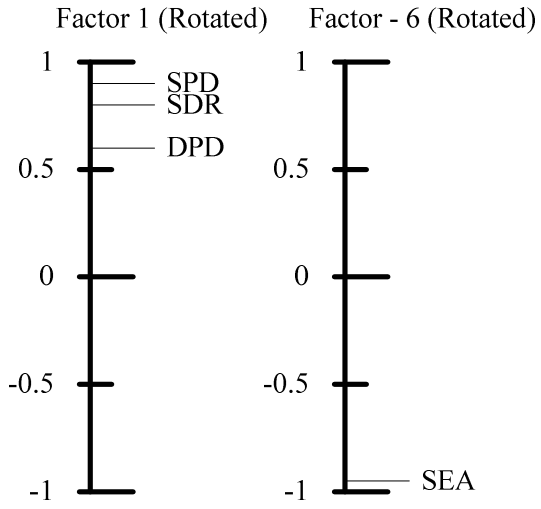


Figure 9. Rotated Factors 1 and 6 for the Non-chosen Alternative with Related Activities.

While Factor Scores and Rotated Factor Scores do not yield models that explain much variance in productivity, they do present an interesting picture of the relations between the activities that appear to associate significantly with productivity. For further detail on the analysis results in this chapter, please refer to Appendix B.

Limitations

The strata reduce the amount of hours spent at each activity, basically dividing it roughly by three. To still be able to analyze the different strata, the data was analyzed at the individual level rather than compiled per team. Teams spending no time on some of the alternatives distort the data, skewing it towards zero. Even with this skew, the assumptions of normality and constant variance do not appear to be violated (Fig. 10, 11).

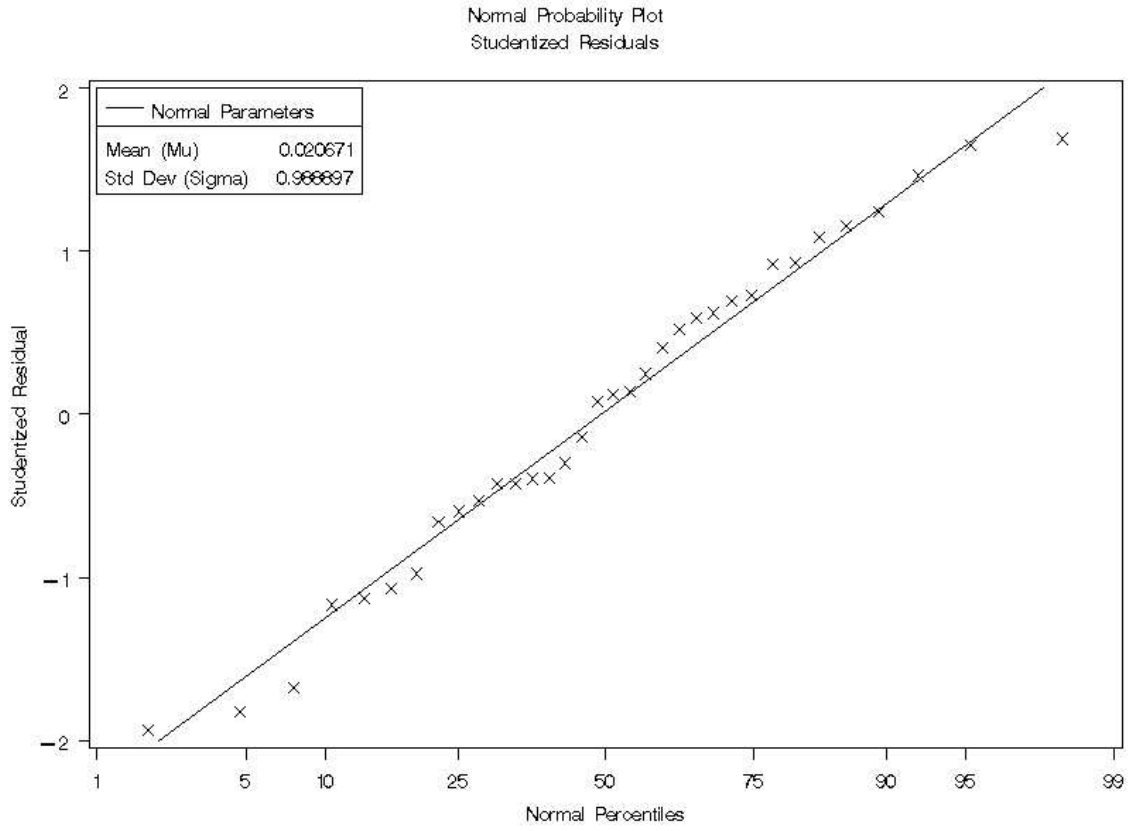


Figure 10. Normal Probability Plot for the Studentized Residuals for the Behavioral Iteration Model.

A correlation test for normality yields a correlation between the theoretical and empirical quantiles of 0.993, which is higher than 0.964, the critical value for a sample of 33 and confidence level of 0.05. As for the variance assumption, the "tulip" pattern observed in the design iterations model appears again, this time a little less obvious.

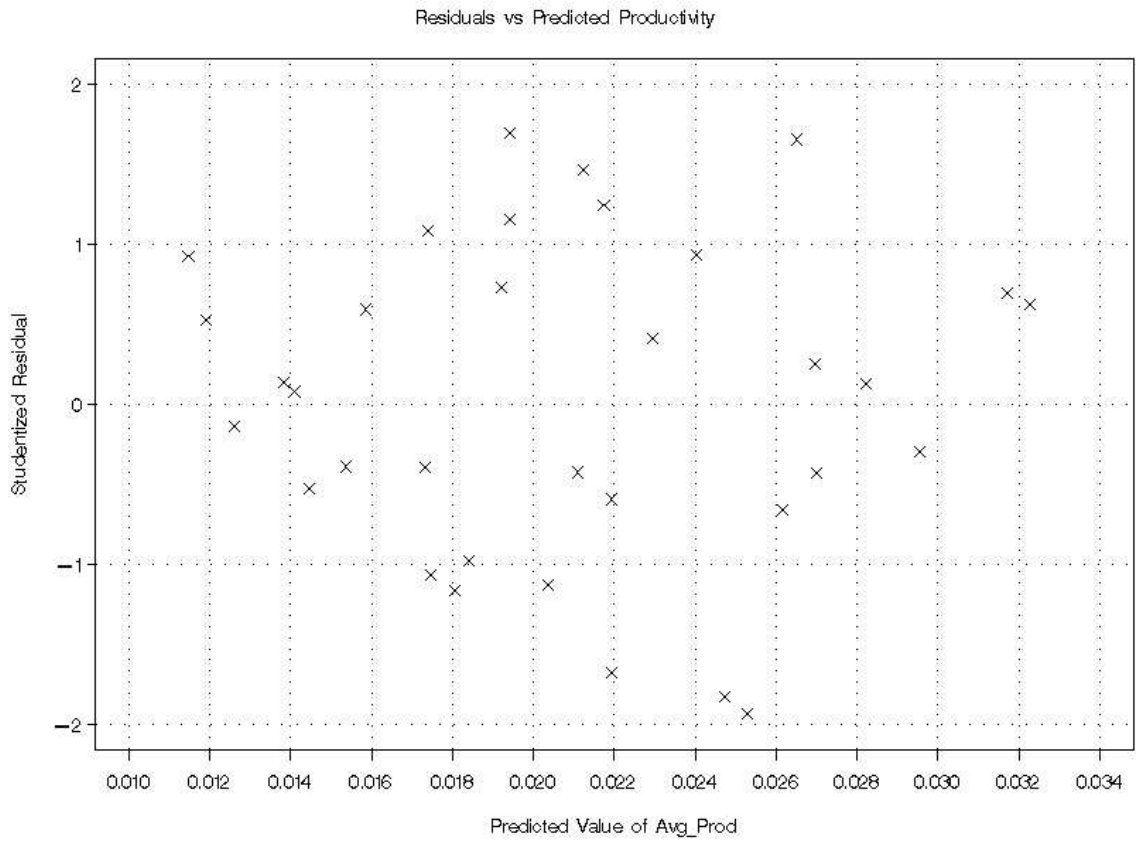


Figure 11. Studentized Residual vs. Predicted Productivity for the Behavioral Iteration Model.

Timing of Productive Activities

So far there seems to be some evidence that the framework's implications are consistent with the performance of the sample projects. Both design and behavioral iteration associate with higher productivity, with behavioral iteration involving only the repetition of system idea generation for higher productivity. Repeating certain activities then relates to higher productivity in the sample. A complementary piece of information is the timing of these activities that relate to higher productivity. Now that we know what

to do, it is important to determine when to do it to be more productive. Most of these activities appear scattered along the project's timeline and differ from team to team, person to person. However, some patterns arise from sample averages, which allow creating a picture of the average design process for the students in the sample. This average process, although not expected to represent any particular project, presents patterns of effort allocation on the activities that are common among most projects in the sample. The next chapter explores these patterns and suggests the division of the project's timeline into three phases based on effort allocation to specific activity groups. These activity groups are based on the affinities the problem solution co-evolution chapter contains, which relate problem definition and idea generation activities in an initial sequence of diagnosis and transformation of design data, and engineering analysis and design refinement in a second.

CHAPTER 7

PROJECT PHASES

Activity timing plays an important role in design processes that should be considered when attempting to improve design productivity because the leverage an activity can have depends on the nature of the information it diagnoses or transforms. For instance, a finite element analysis on a conceptual sketch or on a detailed drawing might not advance the design towards completion in the same manner, although the time to perform them would be similar. To this end, the sample projects were analyzed from a timing perspective to determine when activities occur and which patterns arise that might help explain differences in productivity, and how these patterns relate to iteration types

Iteration Timing and its Relation to Project Phases

Engineering design research often characterizes design as a process composed of phases, with particular emphasis placed on the initial concept creation and selection steps. For example, Ulrich and Eppinger (2000) define problem definition, concept design, system-level design, detail design, and production phases in product development. In a similar fashion, Pahl and Beitz (2001) consider design as a highly iterative process through three stages: concept, embodiment, and detail design. However, as Otto and Wood point out (2001), there is no clear transition between concept and embodiment design. By the same token, Ulrich and Eppinger (2000) discuss system-level design, but

the discussion revolves around degrees of modularity in the design's architecture, and the relationship with concept design is left unstated. In fact, it can be a bit confusing as "modularity" can be a concept-level objective. It seems therefore that there is no clear definition of the design process during the transition from concept to detail design.

Observation of timing patterns in the design journal data has uncovered affinities among certain activities that allow grouping them into phases. Conceptual problem definition and idea generation dominate the first three weeks in the sample, thus defining the first phase – the *front end* (see Figure 12). Other activities at different design levels are present in the first three weeks, but do not show up consistently or in significant amounts in the sample.

Weeks														
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Front End C/PD, C/IG			Transition S, C, D			Back End D/EA, D/DR								

Figure 12. Design Process Phases and their Main Activities.

At the other end, detailed engineering analysis and design refinement dominate the last seven weeks, and can be used to define the *back end* phase of the project. Together, these activities represent the single most important productivity predictor because they correspond heavily with total project hours but not to quality. However, their predictive power is less actionable because they appear at the last stage of the project.

In between the front and back ends is a transition phase, where concept level problem definition and idea generation begin to phase out and detail level engineering analysis

and design refinement begin to pick up. No single activity or design level dominates this phase, although most of the system level activity for the project can be observed during these three weeks.

In comparing these observations with the results reported in the design iteration chapter, an interesting pattern emerges. The amount of time spent in the activities that dominate the front and back ends (concept level problem definition and idea generation, and detail level engineering analysis and design refinement) is not associated with productivity, positively or negatively. On the other hand, those activities most closely associated with positive or negative productivity coefficients are found in greatest abundance in the transition phase. It seems, then, that the transition between concept and detail design is much more critical to designer productivity than either the front or back end design effort, at least in terms of distinguishing among the projects within the sample. This supports the proposition that skipping design levels can be detrimental to project outcomes. In addition, these results suggest that system-level problem definition and idea generation are more fruitful during this transition phase than refining conceptual ideas or generating new ideas at the detail level.

These observations apply to behavioral iteration as well, since the activity we need to repeat at the different alternatives or scopes is, again, system-level idea generation. Also for system-level engineering analysis, which obtained a negative productivity coefficient in the analysis for design iteration, now appears related to lower productivity for the non-chosen alternatives. The role these activities play at the system level indicates that design and behavioral iteration should occur at the transition phase. Note on Figure 13 the

pattern of effort allocation per design levels. The fifth week presents the most similar effort allocation for the three design levels, and the transition phase starts when concept work dips under 60% and ends when detail work rises above 60% - the eighth week falls on spring break, which drops the average effort for most teams. During the first and last weeks many teams do not perform any design work, which lowers the allocation for all design levels, being more noticeable on the dominant one (Fig. 13). Please refer to Appendix C for further information on effort allocation in the sample.

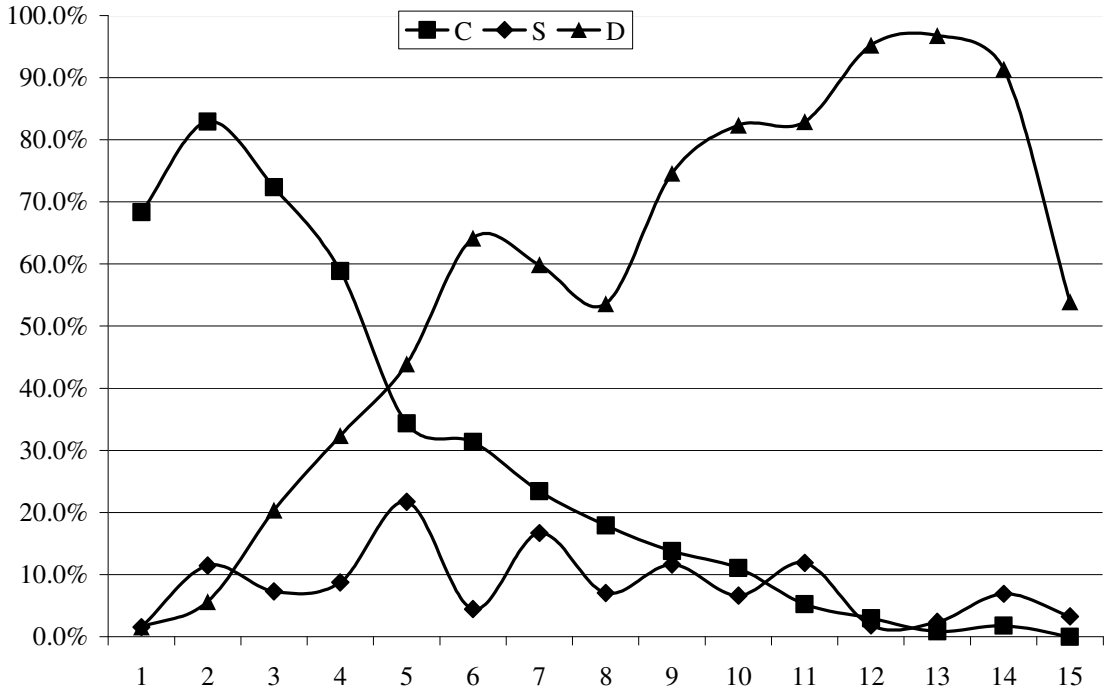


Figure 13. Percent Weekly Effort Allocation Per Design Level.

Two Project Measures that Relate to Productivity

The following sections present two different project measures that relate to differences in productivity among the projects in the sample. These two conditions are evaluated against productivity using t-tests and rank comparisons, and each represent process characteristics present in the sample data. The two measures are sufficient but not necessary for higher productivity, while combined they are necessary and sufficient for lower productivity.

Measure for Front Loading

A possible measure for front loading, or allocating more effort at the initial stages, is to compare the amount of effort allocated to the weeks before selecting an alternative to the total effort for the project. A ratio between the time spent on design activity per week before convergence and the same measure but for the entire project can quantify front loading. This ratio is larger than one if more effort than the weekly average was allocated to work prior to choosing an alternative, and less than one otherwise. The second, third, and fourth most productive teams (out of ten) had a ratio higher than one. A t-test shows that these three projects are significantly more productive than the rest, with a p-value of 0.002, which is less than the confidence level of 0.05 and thus provides evidence to reject the hypothesis of equal productivity sample averages. The variances cannot be considered different because the F-test does not provide evidence to reject the hypothesis of equal variance with a p-value of 0.71, which is higher than any reasonable

level of significance. A linear model using the ratio as explanatory variable explains 60% of the variance in productivity, with the highest productivity project as a possible outlier. Appendix D presents a graph showing the spread of the two groups – without the outlier – in relation to each other, as well as a chart of the linear model that helps clearly identify the outlier. A different measure of front loading is the percent design time allocated before the selection of the final alternative, which was considered but did not present any patterns, as shown in Appendix D.

This measure of front loading alone provides information useful to discern productive from less productive processes, but, as the next section presents, combined with a measure of effort allocation at the end of the project, it allows us to synergize front and back end effort allocation to improve productivity.

Back End Dominated by Engineering Analysis and Design Refinement at a Detail Level

Design teams can also be separated according to the amount of effort spent at D/EA and D/DR at the back end phase (the second half of the project). Teams that allocated more than the sample's average of 75% percent of their weekly effort to these activities were significantly less productive than the rest of the teams, which allocated their effort more evenly. A hypothesis test using the t-distribution shows that the first, third, and fourth most productive teams could not be considered as drawn from the same population as the remaining seven teams with a p-value of 0.0218, which is less than the confidence level of 0.05. This was done considering equal variance of the samples, since, again, an F-test for equal variance did not provide evidence to reject the hypothesis that the

variances are different. Table 13 shows the average weekly effort allocation of the activity groups of interest with the corresponding phase. On average, student teams spent 79% of the front end's design effort on conceptual problem definition and idea generation, and 75% of effort at the back end on detailed engineering analysis and design refinement. Given the variability in the sample, the true means of these effort allocations are greater than zero at a 99% confidence level. Table 13 presents the rest of time allocations to give an idea of the relative magnitudes. Appendix E presents average effort allocation for each of the activities and their design levels.

Table 13. Average Percent Weekly Effort Allocation for the Sample Grouped per Phase.

	Phases (project weeks)		
	Front End (weeks 1 to 3)	Transition (weeks 4 to 6)	Back End (weeks 7 to 15)
C/PD – C/IG	79.2%	31.3%	4.0%
C/EA – C/DR	2.4%	11.5%	2.7%
S	7.4%	11.5%	7.3%
D/PD – D/IG	6.3%	18.2%	11.1%
D/EA – D/DR	4.7%	27.5%	75.0%

Two Sufficient but not Necessary Conditions for a Productive Process

Combining the rankings on the two dimensions shows that the conditions are sufficient but not necessary. All teams that front-loaded are highly productive, but the team with the highest productivity did not front-load. On the other hand, all teams that

did not have the back end phase dominated by D/EA – D/DR had higher productivity, but the second highest did have its back end phase dominated by the activity combination.

The six least productive teams had allocated less time per week before selecting an alternative than after, and during the second half of the project timeline they had mostly evaluated or refined design details. This combination produces one necessary and sufficient condition for lower productivity teams. These teams clearly relied mostly on D/EA and D/DR to achieve an acceptable design, and they were the least productive teams (Table 14).

Table 14. Projects Ranked on Productivity and Classified on Both Measures.

	D/EA – D/DR Dominate Back End Phase?	
	Yes	No
Front Loading Ratio > 1	2nd	3rd, 4th
Front Loading Ratio < 1	5th, 6th, 7th, 8th, 9th,10th	1st

Allocating more effort per week before convergence and extending the front end and transition phases into the back end phase (by delaying the evaluation and refinement of details) are actions to take in order to improve productivity, as they are best-in-class for the sample.

CHAPTER 8

CONCLUSIONS AND RECOMMENDATIONS

The extant literature on iteration provides insightful definitions and categorizations useful to better understanding the phenomenon. However, the taxonomies do not seem very helpful in answering the question of whether iteration is absolutely necessary, and if so under what conditions and with what consequence. In an effort to further explore iteration, this thesis presents a new classification to directly address this question that delineates rework from design and behavioral iteration types.

Rework iteration does not help the design evolve towards the intended goal because it focuses on recovering from previous design errors or poor alternative selection. Design iteration focuses on the evolution of the design through abstraction levels while behavioral iteration explores design space through alterations on scope. The implications derived from this framework are to avoid the causes of rework iteration (identifying errors at the source) by 1) performing design iteration to evolve design definition in intermediate levels, not just concept and detail levels; and 2) performing behavioral iterations to more thoroughly explore alternatives, in parallel to reduce design lead-time. Analysis of actual design data from student projects corroborates the usefulness of these implications.

The first implication of the iteration framework, to perform design iterations without skipping design levels, relates to the importance of the transition phase in the process and specifically to the role that system level work plays. Data analysis provides empirical

support for the implication because generating ideas and defining the problem at system level present the highest positive association with increased productivity.

The second implication also presents the generation of ideas concerning interfaces and configuration of the alternative solutions, even if not selected, as associated with productivity. For these non-chosen alternatives, defining the problem at a conceptual level adds time without improving quality, and the same applies for refining design details. However, evaluating the ideas generated at the system level does not increase time substantially but relates with lower productivity. This seems to indicate that system-level engineering analysis relates to lower quality designs, maybe because it leads to poor alternative selection, and joins the activities that should be avoided for a more productive design process.

The data present patterns that indicate three distinct phases based on how teams allocate design effort. A front end phase practically consists of defining the problem at a conceptual level, which the data show is productive as long as it is not part of alternative exploration, and generating ideas for possible design solutions. A back end phase, which in the data accounts on average for more than half of a project's timeline, is mostly evaluating and refining design details. In the data, the back end phase thus defined is not productive: it associates strongly with total project effort while not associating at all with increased quality. Analysis on both recommendations indicates that the key for a productive process lays on system level work. While the teams in the sample spent an average 7.67% of their effort at this level, activities at this level associate with final productivity, both high (system idea generation) and low (system engineering analysis).

A possible design process model that incorporates these findings would focus mainly on extending the front end and, more substantially, the transition phase further into the timeline. Each of these phases lapses 3 weeks in the data, adding up to only 6 of the 15 weeks of a project's timeline. The pattern should probably reverse, allocating approximately the last 3 weeks of the timeline to the back end or design completion phase. The transition phase and system level design in particular, should increase their current effort allocation both in terms of timeline and weekly effort.

Both recommendations lead to a possible model for a productive design pattern (Fig. 14). The results suggest that allocating effort at defining the problem at a conceptual level, without relating it to specific alternative solutions, is representative of a higher productivity process. It seems then that this activity, which occurs most notably during the front end phase, should be emphasized in models of the design process that attempt to increase productivity.

While this is possibly a commonly accepted result, the present data analysis does show that this activity performed simultaneously with the generation of ideas is common among students and particularly among the more productive teams, as opposed to the assumed need to complete the problem definition first and, when done, generate ideas. This might be a result of representing the process using flow charts, which is not flexible enough to represent simultaneous task execution.

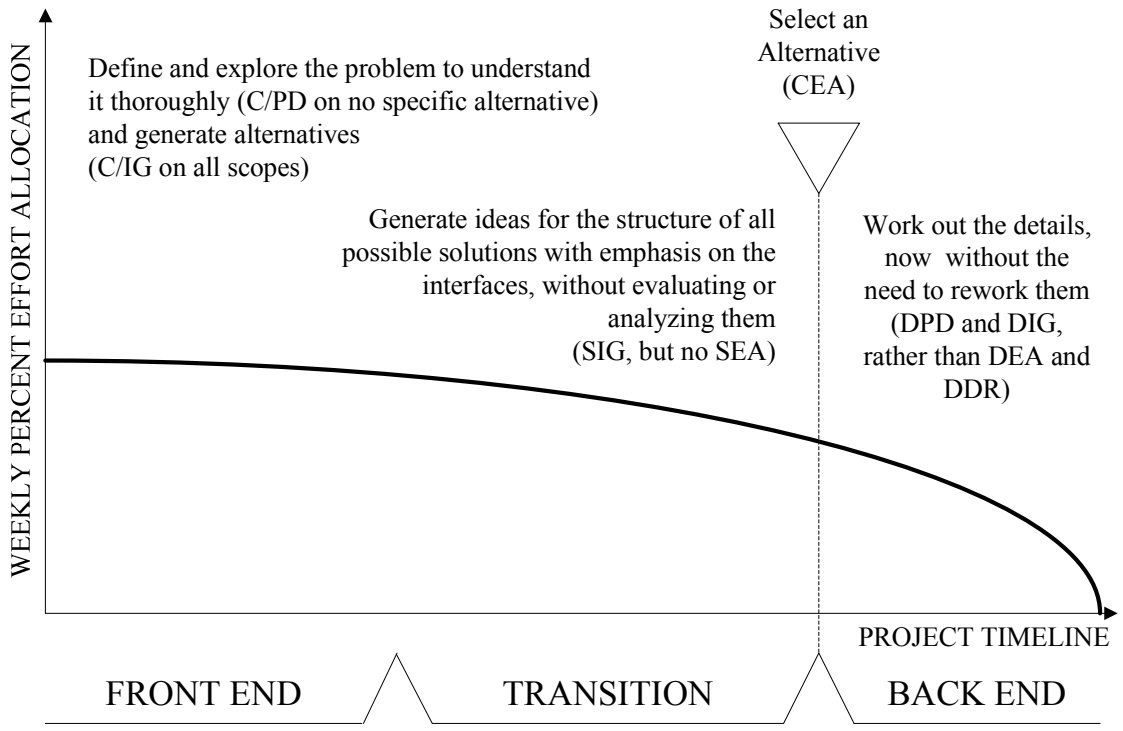


Figure 14. Diagram of the Design Process Proposed to Enhance Productivity.

Generating qualitatively different approaches to a problem when defining subsystems for a particular concept, and defining their configurations and interfaces plays a pivotal role since it is important to repeat this activity at this abstraction when performing both design and behavioral iterations. On the other hand, evaluating existing interfaces or configurations plays the opposite role and associates with lower productivity. Both activities appear mostly during the transition phase, but have also some presence during the front end. The design process models should then consider this phenomena and encourage system idea generation and eliminate system engineering analysis, which only accounts for a few hours at most.

The proposed design model only contains one milestone, which is the selection of an alternative to develop it to completion, as the final design. This milestone is present in most of the teams in the sample projects in a very condensed period of time, although not a milestone in the pure sense of the word, as an event of zero duration; because of its characteristics, it could be considered a fuzzy milestone. The evaluation of all and selection of one alternative is performed at the conceptual level. This indicates that the sequence of design levels is not sequential, following a concept – system – detail format, but rather presents concept and system level work as occurring simultaneously. The selection of a conceptually different final alternative design indicates the fuzzy end of this concept – system tandem. It should be noted that the selection and evaluation of alternatives seems effective when performed at the concept level, taking into account the overall picture and not just each alternative's configuration. At the same time, this decision made at the concept level needs to be informed by activities at the system level, because the greater picture needs to include configuration and interface issues and establish configuration feasibility before committing to one alternative (Sobek et al. 1999). This observation might explain why engineering analysis at a system level is counterproductive.

It is expected that in a process that adheres to these guidelines detail level work will only be necessary to complete the design, and not to address interfaces issues or infeasibilities in concept or configuration. This is suspected to reduce substantially the amount of hours allocated to this level, which for the sample is almost 73% of total effort allocation.

Future Work

The limited sample size on which this analysis rests is the main limitation of the results, which indicates that future work should first increase the sample projects available to provide more degrees of freedom for this type of analysis. I have also received comments on the possible implications of considering the data on an individual versus team aggregated data, which increases degrees of freedom but may result in artificially creating variance. An individual is not an artificial unit, but there is no data available on each individual's productivity since the quality scores were per team. Another important limitation that stems from the nature of the data used is possible bias due to subjective coding of activities, design levels, and scopes. While the analysis does not seem to indicate there is a problem of biased data, it is hard to determine with a relatively small sample. The results should then be evaluated with artificial variance and possible bias stemming from subjective coding in mind.

Whether these results are broadly applicable to mechanical engineering capstone design processes would require additional samples to validate the models and correct them if necessary. As far as extrapolating the results to other fields, such as industrial or electrical engineering, the coding needs to be validated first on samples from these other fields to determine if the activities and design levels represent the same design issues. Once the coding is validated across design disciplines, samples from these disciplines should be used to validate the model – or the applicability to the discipline. If the model does not fit other design disciplines, and the samples from these other disciplines can

form a significantly different model, the applicability of the models presented here will be limited to mechanical capstone courses.

As far as limitations from having only a sample from a specific university, the results can only be considered general to capstone design courses for mechanical engineering at Montana State University. Until samples from other universities are used to validate the model and the model survives, these results are particular to the setting in which the data was collected.

Finally, while the results might be applicable to engineering student designers in an academic setting, they may not directly apply to product development organizations until the guidelines are validated against actual practice because of the inherent inexperience of engineering students. Another approach might be to perform a designed experiment to apply these guidelines to a project in a product development environment and evaluate the results in comparison to other projects developed at the same time, as a control group. In any case, these results are consistent with my professional experience in automotive component product development in an international environment, which was the basis for the framework in the first place.

In the present thesis, system-level work associates both with higher and lower productivity depending on the activity performed and the scope on which it is performed. This seems to indicate that a useful next step is to further the understanding of system level work through continued research. Another area of interest for future research is the role representations play in engineering design and how they may play an important role in the effectiveness and efficiency of system-level design.

REFERENCES

Adams, R.S. Atman, C.J. 1999. Cognitive Processes in Iterative Design Behavior. Proceedings of the 29th ASEE/IEEE Frontiers in Education Conference, Session 11a6. November 10-13, San Juan de Puerto Rico.

Adams, R.S. Atman, C.J. 2000. Characterizing Engineering Student Design Processes – An Illustration of Iteration. Proceedings of the ASEE Annual Conference, Session 2330. June 18-21, St. Louis, MO.

Antonsson, E.K. Otto, K.N. 1995. Imprecision in Engineering Design. ASME Journal of Mechanical Design. 117 B: 25-32.

Ball, L.J. Evans J. St. B. T. Dennis, I. 1994. Cognitive processes in engineering design: a longitudinal study. Ergonomics. 37(11): 1753-1786.

Browning, T.R. 2001. Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Review and New Directions. IEEE Transactions on Engineering Management. August. 48(3): 292-306

Busby, J.A. Lloyd P.A. 1999. Influences on Solution Search Processes in Design Organizations. Research in Engineering Design. 11: 158-171

Costa, R. Sobek II, D.K. 2003. Iteration in Engineering Design: Inherent and Unavoidable or Product of Choices Made? Proceedings of the ASME Design Theory & Methodology Conference (Paper # DETC'03/DTM-48662), Chicago IL, September 2003.

Costa, R. Sobek II, D.K. 2004. How Process Affects Performance: An Analysis of Student Design Productivity. To appear in the Proceedings of the ASME Design Theory & Methodology Conference (Paper # DETC2004-57274), Salt Lake UT, September 2004.

Cusumano, M. 1997. How Microsoft Makes Large Teams Work Like Small Teams. Sloan Management Review. Fall. 39(1): 9-20.

Denker, S. Steward, D.V. Browning, T.R. 2001. Planning Concurrency and Managing Iterations in Projects. Project Management Journal. September. 32(3): 31-38

Hazelrigg, G.A. 2003. Thoughts on Model Validation for Engineering Design. Proceedings of the ASME Design Theory & Methodology Conference (Paper # DETC'03/DTM-48632), Chicago IL, September 2003.

Hoedemaker, G.M. Blackburn, J.D. Van Wassenhove, L.N. 1995. Limits to concurrency. INSEAD R&D Working Papers. January, 95(27): 1-21

Jain, V. Modeling the Relationship between Design Process Variables and Outcomes of Student Engineering Design Projects. Master Thesis, Montana State University, July 2003.

Karlsson, C. Nellore, R. Soderquist, K. 1998. Black Box Engineering: Redefining the Role of Product Specifications. Journal of Product Innovation Management 15: 534-549

Krishnan, V. Eppinger, S.D. Whitney, D.E. 1995. Accelerating Product Development by the Exchange of Preliminary Product Design Information. Journal of Mechanical Design. December. 117: 491-498

Krishnan, V. Eppinger, S.D. Whitney, D.E. 1997. Simplifying Iterations in Cross-Functional Design Decision Making. Journal of Mechanical Design. December. 119: 485-493

March, J.G. Simon, H.A. 1958. Organizations. New York, John Wiley

Neter, J. Kutner, M.H. Nachsheim, C.J. Wasserman W. 1996. Applied Linear Statistical Models. McGraw-Hill

Otto, K.N. Antonsson, E.K. 1994. Modeling Imprecision in Product Development. Proceedings of the Third IEEE International Conference on Fuzzy Systems (FUZZ-IEEE '94). 1: 346-351.

Otto, K. Wood, K. 2001. Product Design: Techniques in Reverse Engineering and New Product Development. Prentice Hall, Inc. Upper Saddle River, New Jersey. 535-550.

Pahl, G. Beitz, W. 2001. Engineering Design: A Systematic Approach. Springer-Verlag. London. 61-70.

European Space Agency Inquiry Board press release: http://www.esa.int/export/esaCP/Pr_20_1996_p_EN.html. Also find the official report in http://ravel.esrin.esa.it/docs/esa-x-1819eng.pdf. Access date: 8 June 2004.

Shingo, Shigeo. 1992. The Shingo Production Management System: Improving Process Functions. Portland, Productivity Press.

Smith, R.P. Eppinger, S.D. 1997a. Identifying Controlling Features of Engineering Design Iteration. Management Science. March. 43(3): 276-293

Smith, R.P. Eppinger, S.D. 1997b. A Predictive Model of Sequential Iteration in Engineering Design. Management Science. August. 43(8): 1104-1120

Sobek II, D.K., Ward, A. Liker, J.K. 1999. Toyota's Principles of Set-Based Concurrent Engineering. Sloan Management Review, Winter, 40(2): 67-82.

Sobek II, D.K. 2002. Preliminary Findings from Coding Student Design Journals. Proceedings of the 2002 American Society for Engineering Education Annual Conference. Montreal, Canada.

Sobek II, D.K. Jain, J.K. 2004. Two Instruments for Assessing Design Outcomes of Capstone Projects. To appear in the Proceedings of the 2004 American Society of Engineering Education Conference, Salt Lake City.

Steward, D.V. 1981a. System Analysis and Management: Structure, Strategy, and Design. Princeton, NJ: Petrocelli Books.

Steward, D.V. 1981b. The design structure system: A method for managing the design of complex systems. IEEE Transactions on Engineering Management. 28(3): 71-74

Terwiesch, C. Loch, C.H. De Meyer, A. 2002. Exchanging Preliminary Information in Concurrent Engineering: Alternative Coordination Strategies. Organization Science. July-August, 13(4): 402-419

Ulrich, K. Eppinger, S. 2000. Product Design and Development. 2nd Ed. Irwin McGraw-Hill, Boston.

Van Eikema Hommes, Q.D. Whitney, J. Zambito, T. 2001. The Predictability of System Interactions at Early Phase of Product Development Process. Proceedings of the ASME Design Theory & Methodology Conference (Paper # DETC'03/DTM-48635), Chicago IL, September 2003.

Ward, A. Liker, J.K. Cristiano, J.J. Sobek II, D.K. 1995. The Second Toyota Paradox: How Delaying Decisions Can Make Better Cars Faster. Sloan Management Review. Spring. 36: 43-61.

Wood, K.L., Antonsson, E.K. 1989. Computations with Imprecise Parameters in Preliminary Engineering Design: Background and Theory. Transactions of the ASME. December. 111: 616-625.

Yassine, A.A. Whitney, D.E. Lavine, J. Zambito, T. 2000. Do-it-Right-First-Time (DRTF) Approach to Design Structure Matrix (DSM) Restructuring. Proceedings of the ASME Design Theory & Methodology Conference (Paper # DETC'00/DTM-14547), Baltimore, MD, September 2000.

Yassine, A.A. Whitney D.E. Zambito, T. 2001. Assessment of Rework Probabilities for Simulating Product Development Processes Using the Design Structure Matrix (DSM). Proceedings of the ASME Design Theory & Methodology Conference (Paper # DETC'01/DTM-21693), Pittsburgh PA, September 2001.

APPENDICES

APPENDIX A

DESIGN ITERATION ANALYSIS

FACTOR ANALYSIS ON TEAM AGGREGATED DATA

The FACTOR Procedure
Initial Factor Method: Principal Components

Prior Communality Estimates: ONE

Eigenvalues of the Correlation Matrix: Total = 12 Average = 1

	Eigenvalue	Difference	Proportion	Cumulative
1	3.41550553	0.23340404	0.2846	0.2846
2	3.18210150	1.14323883	0.2652	0.5498
3	2.03886267	0.59105596	0.1699	0.7197
4	1.44780671	0.48339276	0.1207	0.8404
5	0.96441395	0.57367699	0.0804	0.9207
6	0.39073696	0.05866997	0.0326	0.9533
7	0.33206699	0.18744361	0.0277	0.9810
8	0.14462339	0.08630738	0.0121	0.9930
9	0.05831601	0.03889514	0.0049	0.9979
10	0.01942087	0.01327544	0.0016	0.9995
11	0.00614543	0.00614543	0.0005	1.0000
12	0.00000000		0.0000	1.0000

4 factors will be retained by the MINEIGEN criterion.

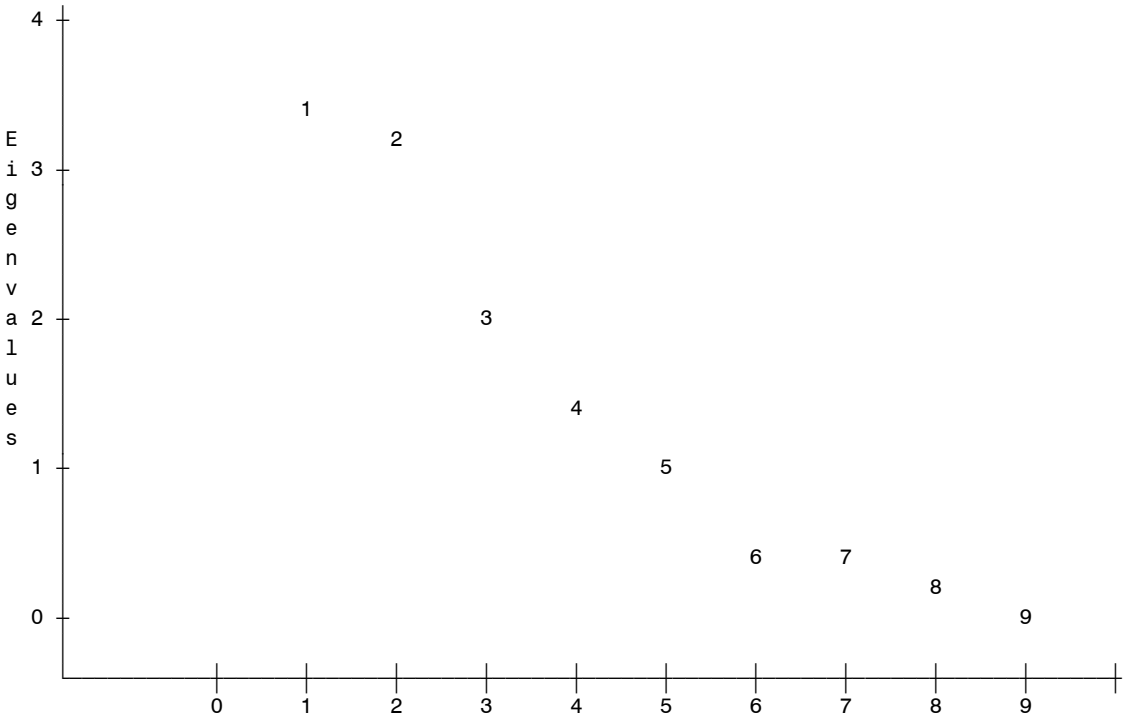
Factor Pattern

	Factor1	Factor2	Factor3	Factor4
C_PD	-0.66831	0.62745	0.24788	0.13608
C_IG	0.49204	0.74254	0.18189	0.13495
C_EA	-0.76442	0.25036	0.44050	0.10041
C_DR	0.29676	0.69164	-0.39081	0.44471
S_PD	-0.34062	0.81634	-0.07035	0.08676
S_IG	-0.61422	0.21741	0.56555	-0.31869
S_EA	0.35823	0.19041	0.78133	0.33967
S_DR	0.32615	0.73413	-0.46620	-0.07074
D_PD	0.59160	0.33694	0.13891	-0.43971
D_IG	0.55786	0.18833	0.32054	-0.68211
D_EA	0.36401	-0.50695	0.24994	0.51893
D_DR	0.72983	0.03315	0.51034	0.21413

Variance Explained by Each Factor

Factor1	Factor2	Factor3	Factor4
3.4155055	3.1821015	2.0388627	1.4478067

Scree Plot of Eigenvalues



Final Communality Estimates: Total = 10.084276

C_PD	C_IG	C_EA	C_DR	S_PD	S_IG
0.92030314	0.84477255	0.85114335	0.91693086	0.79491610	0.84594049
S_EA	**S_DR**	**D_PD**	**D_IG**	**D_EA**	**D_DR**
0.89042795	0.86767554	0.67616705	0.91468766	0.72125829	0.84005342

Scoring Coefficients Estimated by Regression

Squared Multiple Correlations of the Variables with Each Factor

Factor1	Factor2	Factor3	Factor4
1.0000000	1.0000000	1.0000000	1.0000000

Standardized Scoring Coefficients

	Factor1	Factor2	Factor3	Factor4
C_PD	0.10106	0.21165	0.46917	0.29937
C_IG	-0.01951	0.22538	-0.10240	-0.02001
C_EA	-0.00822	0.08919	0.46859	0.21857
C_DR	0.37378	0.23134	0.14439	0.50573
S_PD	-0.36567	0.24358	-0.34603	-0.12415
S_IG	-0.74169	0.04093	-0.38078	-0.60901
S_EA	0.36451	0.07250	0.68734	0.41431
S_DR	-0.14539	0.21896	-0.51083	-0.21559
D_PD	-0.02509	0.09622	-0.16416	-0.44096
D_IG	0.65682	0.08324	0.73529	-0.12956
D_EA	-0.15083	-0.17186	-0.17894	0.18026
D_DR	0.00000	0.00000	0.00000	0.00000

Factor Scores

Factor_1	Factor_2	Factor_3	Factor_4	Avg_Prod
-6.629	4.042	-10.403	-4.428	0.0133
-11.971	0.122	2.132	24.723	0.0120
-9.034	8.054	14.225	16.704	0.0350
-27.038	-20.833	-16.573	34.088	0.0300
-20.370	-15.449	-17.404	23.071	0.0189
-10.420	27.434	-2.732	5.971	0.0281
-6.806	11.682	8.123	9.830	0.0228
-11.796	-6.779	-0.925	20.046	0.0144
-6.719	1.752	-6.247	-7.063	0.0288
0.354	3.851	0.315	-9.545	0.0153
4.396	20.277	15.535	-15.978	0.0250

MULTIVARIATE LINEAR REGRESSION MODEL: TEAM AGGREGATED FACTORS VS. PRODUCTIVITY

Dependent Variable: Avg_Prod
R-Square Selection Method

Number in Model	R-Square	Adjusted R-Square	C(p)	Variables in Model
1	0.0414	-.0651	64.0467	Factor_3
1	0.0391	-.0677	64.2164	Factor_2
1	0.0182	-.0909	65.7661	Factor_1
1	0.0018	-.1091	66.9799	Factor_4
2	0.2073	0.0091	53.7533	Factor_1 Factor_2
2	0.1953	-.0059	54.6397	Factor_1 Factor_3
2	0.0872	-.1410	62.6499	Factor_2 Factor_4
2	0.0574	-.1782	64.8580	Factor_3 Factor_4
2	0.0481	-.1899	65.5497	Factor_2 Factor_3
2	0.0472	-.1910	65.6160	Factor_1 Factor_4
3	0.9186	0.8836	3.0365	Factor_1 Factor_3 Factor_4
3	0.2937	-.0090	49.3478	Factor_1 Factor_2 Factor_3
3	0.2430	-.0814	53.1028	Factor_1 Factor_2 Factor_4
3	0.0923	-.2967	64.2726	Factor_2 Factor_3 Factor_4
4	0.9190	0.8651	5.0000	Factor_1 Factor_2 Factor_3 Factor_4

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	4	0.00056260	0.00014065	17.03	0.0020
Error	6	0.00004956	0.00000826		
Corrected Total	10	0.00061215			

Root MSE	0.00287	R-Square	0.9190
Dependent Mean	0.02220	Adj R-Sq	0.8651
Coeff Var	12.94844		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|	Standardized Estimate
Intercept	1	-0.00148	0.00309	-0.48	0.6490	0
Factor_1	1	-0.00406	0.00051895	-7.83	0.0002	-4.50064
Factor_2	1	-0.00002	0.00010824	-0.19	0.8548	-0.03748
Factor_3	1	0.00158	0.00022259	7.08	0.0004	2.23320
Factor_4	1	-0.00152	0.00022257	-6.81	0.0005	-3.16561

MULTIVARIATE LINEAR REGRESSION MODEL: INDIVIDUAL DATA, DESIGN LEVEL-ACTIVTY CODES VS. PRODUCTIVITY

Dependent Variable: Avg_Prod

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	6	0.00139	0.00023204	12.88	<.0001
Error	27	0.00048624	0.00001801		
Corrected Total	33	0.00188			

Root MSE	0.00424	R-Square	0.7411
Dependent Mean	0.02089	Adj R-Sq	0.6836
Coeff Var	20.31334		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|	Standardized Estimate
Intercept	1	0.04191	0.00386	10.85	<.0001	0
CDR	1	-0.18490	0.07556	-2.45	0.0212	-0.29678
SIG	1	0.08902	0.03242	2.75	0.0106	0.28961
SEA	1	-0.19370	0.05217	-3.71	0.0009	-0.37736
DIG	1	-0.09093	0.02353	-3.86	0.0006	-0.45124
DEA	1	-0.02739	0.00697	-3.93	0.0005	-0.47598
DDR	1	-0.03003	0.00504	-5.96	<.0001	-0.71376

APPENDIX B

BEHAVIORAL ITERATION ANALYSIS

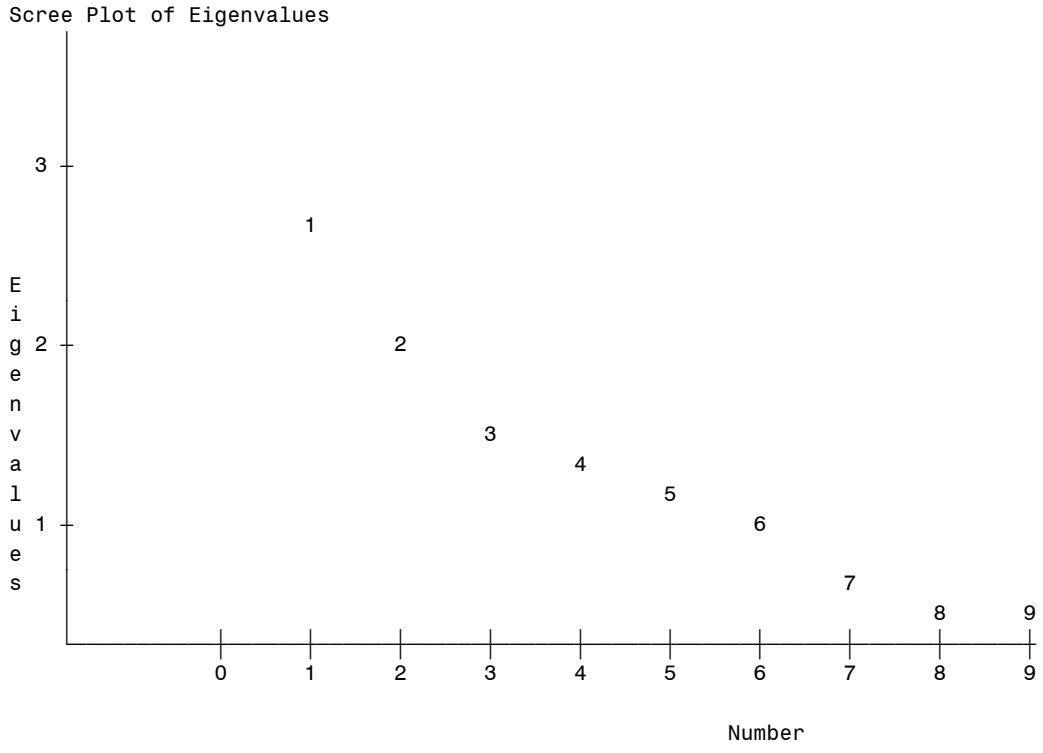
FACTOR ANALYSIS ON INDIVIDUAL DATA

Prior Communality Estimates: ONE

Eigenvalues of the Correlation Matrix: Total = 12 Average = 1

	Eigenvalue	Difference	Proportion	Cumulative
1	2.60716265	0.55473874	0.2173	0.2173
2	2.05242391	0.48540078	0.1710	0.3883
3	1.56702313	0.17539954	0.1306	0.5189
4	1.39162359	0.19177375	0.1160	0.6349
5	1.19984984	0.13276956	0.1000	0.7348
6	1.06708028	0.36298464	0.0889	0.8238
7	0.70409564	0.20037236	0.0587	0.8824
8	0.50372328	0.04473616	0.0420	0.9244
9	0.45898712	0.19229020	0.0382	0.9627
10	0.26669692	0.08536334	0.0222	0.9849
11	0.18133359	0.18133355	0.0151	1.0000
12	0.00000004		0.0000	1.0000

6 factors will be retained by the NFACTOR criterion.



Factor Pattern

	Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
SPD	0.65487	-0.51399	0.13133	-0.07072	-0.26184	0.07499
DPD	0.63936	-0.13679	-0.32411	0.04317	-0.24844	0.06107
CEA	0.60645	0.32737	0.30960	-0.03493	-0.08939	-0.31131
DDR	-0.83488	-0.02368	0.48110	0.19896	-0.14332	-0.03270
CPD	0.32790	0.80557	-0.11033	0.08153	-0.14959	0.04301
CIG	0.46142	0.59405	0.50152	-0.13436	0.18983	0.06756
SDR	0.42366	-0.73517	0.18075	0.11301	-0.12575	-0.17099
DEA	0.00847	0.06619	-0.72641	-0.52021	0.29074	-0.19560
DIG	0.30418	0.20296	-0.18767	0.76482	0.00607	0.13704
SIG	0.11170	-0.18578	-0.19356	0.54025	0.66596	0.17138
CDR	0.31534	-0.19736	0.46977	-0.28718	0.64510	0.08262
SEA	0.01677	-0.02104	0.00199	-0.27672	-0.15029	0.91151

Variance Explained by Each Factor

Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
2.6071627	2.0524239	1.5670231	1.3916236	1.1998498	1.0670803

Final Communality Estimates: Total = 9.885163

CIG	CPD	CEA	CDR	SIG	SPD
0.87598294	0.79951007	0.67692905	0.86452804	0.84920415	0.78947125

SEA	SDR	DIG	DPD	DEA	DDR
0.93074512	0.81045870	0.77270266	0.59985164	0.92553811	0.99024168

Orthogonal Transformation Matrix

	1	2	3	4	5	6
1	0.73371	0.57470	0.23973	0.22153	0.15636	0.01969
2	-0.58601	0.76456	0.09656	0.02153	-0.24488	0.04782
3	0.01092	0.27692	-0.77418	-0.26628	0.50088	-0.04543
4	0.02216	-0.03386	-0.47617	0.78023	-0.32469	-0.23969
5	-0.34185	-0.06601	0.30347	0.46076	0.74569	-0.13542
6	-0.02806	-0.05460	-0.12259	0.24186	0.05688	0.95889

Rotated Factor Pattern

	Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
SPD	0.86896	0.03534	-0.04929	-0.05864	0.12602	0.10667
SDR	0.79393	-0.25474	-0.18037	0.01877	0.19662	-0.20905
DPD	0.62990	0.18471	0.28755	0.15898	-0.22467	0.10263
CIG	-0.07386	0.84658	-0.10698	-0.01956	0.36690	0.08599
CPD	-0.18096	0.77857	0.15231	0.12446	-0.33683	0.09195
CEA	0.29502	0.70864	-0.03502	-0.08478	0.09671	-0.26450
DEA	-0.14594	-0.13658	0.93071	-0.12250	-0.00415	-0.06591
DDR	-0.53911	-0.36018	-0.70911	-0.23228	-0.05710	-0.09907
SIG	-0.03179	-0.20306	0.08253	0.84211	0.29694	-0.05323
DIG	0.11322	0.24424	-0.14133	0.75440	-0.33214	-0.02852
CDR	0.12295	0.12305	0.01524	0.03367	0.91192	0.03613
SEA	0.04432	-0.03637	-0.02514	-0.06196	0.03840	0.95995

Prerotation Method: Varimax

Variance Explained by Each Factor

Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
2.2502689	2.1910127	1.5502293	1.4043216	1.3972875	1.0920434

Final Communality Estimates: Total = 9.885163

CIG	CPD	CEA	CDR	SIG	SPD
0.87598294	0.79951007	0.67692905	0.86452804	0.84920415	0.78947125

SEA	SDR	DIG	DPD	DEA	DDR
0.93074512	0.81045870	0.77270266	0.59985164	0.92553811	0.99024168

Scoring Coefficients Estimated by Regression

Squared Multiple Correlations of the Variables with Each Factor

Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
1.0000000	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000

Standardized Scoring Coefficients

	Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
SPD	0.43572	0.00686	-0.00492	-0.07666	0.00585	0.11052
SDR	0.41436	-0.11164	-0.03902	-0.00198	0.06440	-0.16223
DPD	0.37211	0.09120	0.32560	0.09995	-0.19556	0.11897
CIG	-0.04414	0.42972	0.02472	0.00201	0.27857	0.08423
CPD	-0.06878	0.37235	0.11416	0.06862	-0.21682	0.07615
CEA	0.12419	0.33877	-0.02954	-0.11534	0.03428	-0.25595
DEA	0.01811	-0.02318	0.92732	-0.02699	0.07349	-0.04864
DDR	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
SIG	-0.07942	-0.12411	0.11445	0.65124	0.26791	-0.00332
DIG	0.06611	0.10240	-0.06956	0.54105	-0.22771	0.01568
CDR	-0.03064	0.05288	0.06622	0.05700	0.66691	0.03944
SEA	0.04675	-0.02306	0.00141	0.00706	0.02758	0.89148

Rotation Method: Promax (power = 3)

Target Matrix for Procrustean Transformation

	Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
SPD	1.00000	0.00009	-0.00019	-0.00038	0.00302	0.00176
SDR	0.73325	-0.03061	-0.00888	0.00001	0.01104	-0.01271
DPD	0.57510	0.01833	0.05652	0.01133	-0.02587	0.00236
CIG	-0.00053	1.00000	-0.00165	-0.00001	0.06385	0.00079
CPD	-0.00886	0.89206	0.00546	0.00353	-0.05666	0.00110
CEA	0.04929	0.86338	-0.00009	-0.00143	0.00172	-0.03373
DEA	-0.00373	-0.00387	1.00000	-0.00271	-0.00000	-0.00033
DDR	-0.16999	-0.06408	-0.39965	-0.01667	-0.00020	-0.00100
SIG	-0.00004	-0.01446	0.00079	1.00000	0.03546	-0.00020
DIG	0.00228	0.02898	-0.00459	0.82834	-0.05718	-0.00003
CDR	0.00247	0.00313	0.00000	0.00006	1.00000	0.00006
SEA	0.00010	-0.00007	-0.00002	-0.00035	0.00007	1.00000

Procrustean Transformation Matrix

	1	2	3	4	5	6
1	0.87884599	-0.0744141	-0.0234194	-0.06914	-0.0501501	0.01808937
2	-0.0556013	1.02252143	-0.0496038	-0.0648916	0.03997312	-0.0439485
3	-0.04059	-0.0681094	0.8126913	-0.0526356	0.03622527	-0.0444886
4	-0.0724748	-0.0701654	-0.0564285	1.07333797	0.03631347	-0.0009561
5	-0.0689793	0.08803714	0.05156752	0.06788315	0.72356869	0.02156565
6	0.01614873	-0.088695	-0.0623388	-0.0067952	0.03059368	0.89495012

Normalized Oblique Transformation Matrix

	1	2	3	4	5	6
1	0.67215	0.52024	0.18164	0.14179	0.16620	-0.00213
2	-0.63570	0.80236	0.04741	-0.00770	-0.15886	-0.01255
3	0.01124	0.40739	-0.75895	-0.22042	0.47565	-0.00814
4	0.00310	-0.06681	-0.54648	0.80948	-0.33290	-0.22726
5	-0.46173	-0.01764	0.35298	0.53601	0.82302	-0.13958
6	-0.02637	-0.14436	-0.21112	0.25826	0.10552	0.98863

Inter-Factor Correlations

	Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
Factor1	1.00000	0.14009	0.08823	0.15620	0.10959	-0.01891
Factor2	0.14009	1.00000	0.17064	0.17014	-0.15882	0.15793
Factor3	0.08823	0.17064	1.00000	0.15435	-0.13888	0.15021
Factor4	0.15620	0.17014	0.15435	1.00000	-0.12772	0.04443
Factor5	0.10959	-0.15882	-0.13888	-0.12772	1.00000	-0.10295
Factor6	-0.01891	0.15793	0.15021	0.04443	-0.10295	1.00000

Rotated Factor Pattern (Standardized Regression Coefficients)

	Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
SPD	0.88709	-0.01969	-0.07470	-0.11036	0.06891	0.13075
SDR	0.81707	-0.27648	-0.16513	0.00580	0.11402	-0.17033
DPD	0.62630	0.08350	0.23146	0.08070	-0.23857	0.08823
CIG	-0.15170	0.91688	-0.14249	-0.03926	0.42896	0.05831
CPD	-0.22475	0.76298	0.07505	0.06154	-0.27167	0.03496
CEA	0.25237	0.75314	-0.05603	-0.14136	0.10126	-0.29527
DEA	-0.17525	-0.18056	0.98419	-0.15498	0.03720	-0.11067
DDR	-0.47305	-0.26339	-0.67032	-0.14845	-0.09380	-0.05938
SIG	-0.11933	-0.24239	0.06204	0.89848	0.34235	-0.04263
DIG	0.06928	0.17365	-0.23744	0.74069	-0.30611	-0.04085
CDR	0.04177	0.19296	0.05859	0.07733	0.94246	0.05488
SEA	0.06917	-0.11778	-0.09373	-0.06705	0.07169	0.98523

Reference Axis Correlations

	Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
Factor1	1.00000	-0.13243	-0.07044	-0.14346	-0.15434	0.04166
Factor2	-0.13243	1.00000	-0.10609	-0.11267	0.13301	-0.12959
Factor3	-0.07044	-0.10609	1.00000	-0.10664	0.10366	-0.12088
Factor4	-0.14346	-0.11267	-0.10664	1.00000	0.11024	-0.00175
Factor5	-0.15434	0.13301	0.10366	0.11024	1.00000	0.05912
Factor6	0.04166	-0.12959	-0.12088	-0.00175	0.05912	1.00000

Reference Structure (Semipartial Correlations)

	Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
SPD	0.85761	-0.01872	-0.07191	-0.10629	0.06636	0.12767
SDR	0.78992	-0.26289	-0.15896	0.00559	0.10979	-0.16632
DPD	0.60548	0.07940	0.22282	0.07772	-0.22973	0.08616
CIG	-0.14666	0.87184	-0.13717	-0.03781	0.41306	0.05693
CPD	-0.21728	0.72549	0.07225	0.05927	-0.26161	0.03414
CEA	0.24398	0.71614	-0.05394	-0.13615	0.09751	-0.28833
DEA	-0.16943	-0.17169	0.94746	-0.14926	0.03582	-0.10807
DDR	-0.45733	-0.25045	-0.64530	-0.14297	-0.09032	-0.05798
SIG	-0.11536	-0.23048	0.05972	0.86536	0.32966	-0.04163
DIG	0.06698	0.16511	-0.22858	0.71338	-0.29477	-0.03989
CDR	0.04038	0.18348	0.05640	0.07448	0.90755	0.05359
SEA	0.06687	-0.11200	-0.09023	-0.06457	0.06903	0.96206

Variance Explained by Each Factor Eliminating Other Factors

Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
2.1160840	2.0938595	1.4882485	1.3510432	1.3512126	1.0857160

Factor Structure (Correlations)

	Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
SPD	0.86559	0.08276	-0.00676	0.01034	0.18027	0.08764
SDR	0.78039	-0.23421	-0.18074	0.03878	0.28720	-0.26572
DPD	0.64321	0.27629	0.35981	0.26285	-0.23473	0.15249
CIG	0.00395	0.80572	-0.05629	0.01885	0.28551	0.13867
CPD	-0.13207	0.80344	0.23789	0.20408	-0.43936	0.20168
CEA	0.34754	0.69217	0.01452	-0.00850	0.06553	-0.20623
DEA	-0.13175	-0.08692	0.89221	-0.07083	-0.05883	0.00125
DDR	-0.60143	-0.46378	-0.77580	-0.36127	0.01436	-0.18966
SIG	0.03086	-0.15676	0.09488	0.80255	0.24879	-0.06466
DIG	0.15558	0.31102	-0.05100	0.78169	-0.38351	0.01402
CDR	0.18830	0.08095	-0.01551	0.00780	0.89273	-0.00022
SEA	0.02315	0.00871	0.01996	-0.05613	0.01813	0.94088

Variance Explained by Each Factor Ignoring Other Factors

Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
2.3504308	2.2624090	1.6322958	1.5066465	1.4588580	1.1295787

Final Communality Estimates: Total = 9.885163

CIG	CPD	CEA	CDR	SIG	SPD
0.87598294	0.79951007	0.67692905	0.86452804	0.84920415	0.78947125

SEA	SDR	DIG	DPD	DEA	DDR
0.93074512	0.81045870	0.77270266	0.59985164	0.92553811	0.99024168

Scoring Coefficients Estimated by Regression

Squared Multiple Correlations of the Variables with Each Factor

Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
1.0000000	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000

Standardized Scoring Coefficients

	Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
SPD	0.42636	0.03526	0.01871	-0.03731	0.03459	0.10321
SDR	0.40840	-0.09838	-0.03940	0.01638	0.10866	-0.18265
DPD	0.37939	0.16443	0.37231	0.17604	-0.21294	0.16299
CIG	0.00369	0.41195	0.04709	0.02263	0.22774	0.11276
CPD	-0.04748	0.39419	0.16099	0.11465	-0.26880	0.13482
CEA	0.14545	0.31736	-0.01742	-0.08096	0.02880	-0.22850
DEA	0.04991	0.03903	0.90899	0.04307	0.01092	0.02495
DDR	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
SIG	-0.02485	-0.08879	0.12336	0.62109	0.21867	-0.00644
DIG	0.09242	0.15374	-0.01055	0.55547	-0.26242	0.04113
CDR	0.01907	0.02165	0.03668	0.02868	0.64200	0.01579
SEA	0.03241	0.02604	0.04861	0.01549	0.00200	0.87854

MULTIVARIATE LINEAR REGRESSION MODEL: INDIVIDUAL DATA, DESIGN LEVEL-ACTIVITY CODES VS PRODUCTIVITY

Dependent Variable: Avg_Prod

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	4	0.00105	0.00026232	9.27	<.0001
Error	29	0.00082088	0.00002831		
Corrected Total	33	0.00187			

Root MSE	0.00532	R-Square	0.5611
Dependent Mean	0.02090	Adj R-Sq	0.5005
Coeff Var	25.46211		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|	Standardized Estimate
Intercept	1	0.02924	0.00304	9.63	<.0001	0
SIG	1	0.09711	0.03314	2.93	0.0065	0.36935
CPD	1	-0.08347	0.02938	-2.84	0.0081	-0.37005
SEA	1	-0.08729	0.03488	-2.50	0.0182	-0.31002
DDR	1	-0.01629	0.00446	-3.65	0.0010	-0.48262

APPENDIX C

EFFORT ALLOCATION IN THE SAMPLE

WEEKLY AVERAGE TIME ALLOCATION PER ACTIVITY AND DESIGN LEVEL

The percentages presented are the averages, over the sample of teams, of the percentage effort allocation per activity per week. For a given team, effort allocation adds to 100% across activities, but not in this table as it presents the averages for the sample. The shaded areas indicate percent allocations lower than 10%, to visually indicate which activities dominate a given week. Note that the front and back ends are dominated by C and D activities respectively, while the transition phase (weeks 4 to 6 inclusive) spreads more evenly across design levels.

	Weeks														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CPD	46%	62%	55%	30%	17%	21%	14%	12%	6%	7%	2%	2%	1%	0%	0%
CIG	21%	16%	15%	17%	9%	2%	0%	5%	1%	0%	0%	0%	0%	0%	0%
CEA	1%	3%	2%	9%	6%	4%	8%	0%	5%	4%	2%	1%	0%	1%	0%
CDR	0%	1%	0%	4%	2%	4%	2%	1%	1%	1%	1%	0%	0%	0%	0%
SPD	0%	0%	1%	0%	1%	1%	5%	2%	6%	2%	2%	0%	0%	2%	0%
SIG	1%	8%	2%	5%	13%	2%	8%	1%	2%	1%	3%	1%	1%	0%	3%
SEA	0%	1%	3%	3%	2%	1%	1%	0%	0%	2%	1%	1%	0%	0%	0%
SDR	0%	3%	2%	1%	5%	1%	4%	4%	4%	3%	5%	0%	1%	5%	0%
DPD	1%	1%	11%	13%	14%	21%	11%	17%	14%	10%	7%	6%	4%	5%	0%
DIG	0%	1%	2%	4%	6%	6%	3%	3%	4%	4%	3%	1%	1%	1%	0%
DEA	0%	2%	3%	14%	15%	19%	15%	18%	22%	30%	26%	34%	40%	41%	17%
DDR	0%	2%	5%	2%	9%	18%	30%	15%	34%	38%	47%	54%	52%	45%	37%

WEEKLY MINIMUM TIME ALLOCATION PER ACTIVITY AND DESIGN LEVEL

The objective of this table is to indicate the minimum time allocations per activity per week across the sample to indicate which activities had effort allocated to them by any of the teams in the sample for a given week. This approach was preferred to a standard deviation due to the non-normal distribution of effort allocation. Note that only C/PD and C/IG had effort allocated to during the front end for any of the teams in the sample, with the same happening for D/EA and D/DR at the back end. The non-shaded areas indicate the effort allocated is greater than zero.

	Weeks														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CPD	0%	12%	17%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
CIG	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
CEA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
CDR	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
SPD	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
SIG	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
SEA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
SDR	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
DPD	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
DIG	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
DEA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	4%	0%
DDR	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	3%	6%	3%	0%

WEEKLY MAXIMUM TIME ALLOCATION PER ACTIVITY AND DESIGN LEVEL

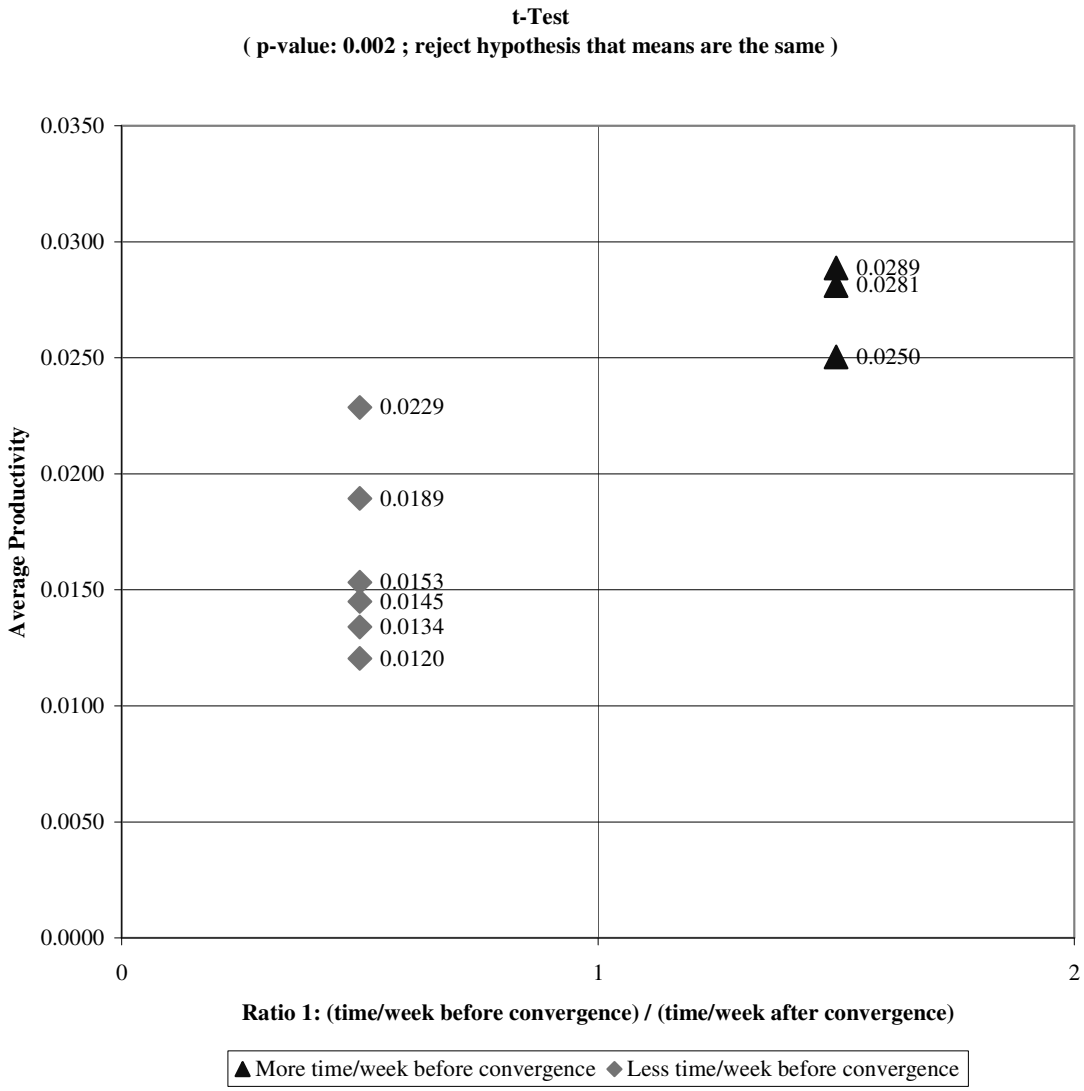
The maximum effort allocation for each activity per week indicates which activities were not allocated any time for a given week. The shaded areas indicate zero effort allocation. Although most activities had some effort allocated to them by at least one of the teams, large effort allocations per week are consistent with phase definitions: C/PD and C/IG dominate the first three weeks or front end, the following three weeks or transition phase presents a more even distribution of effort, and the second half of the project or back end is dominated by D/EA and D/DR. Notice that some teams did not follow the phases, as at least one team was heavily investing time on C/PD as far as week 10. However, these phases help describe the overall effort-allocation patterns in the sample.

	Weeks														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CPD	100%	100%	100%	87%	60%	100%	100%	100%	36%	80%	19%	17%	10%	6%	0%
CIG	100%	74%	52%	47%	38%	20%	2%	72%	14%	2%	2%	0%	0%	2%	0%
CEA	8%	23%	13%	64%	28%	28%	62%	0%	39%	51%	14%	12%	2%	11%	0%
CDR	0%	9%	3%	23%	8%	52%	23%	13%	9%	9%	13%	0%	0%	0%	0%
SPD	0%	1%	10%	3%	9%	7%	44%	22%	65%	17%	27%	1%	5%	25%	0%
SIG	10%	24%	12%	13%	80%	10%	71%	14%	14%	6%	37%	5%	12%	7%	45%
SEA	5%	6%	32%	22%	12%	4%	8%	0%	2%	13%	6%	5%	3%	0%	0%
SDR	0%	17%	14%	8%	38%	6%	23%	50%	14%	28%	71%	3%	3%	64%	0%
DPD	11%	6%	72%	49%	50%	100%	40%	61%	81%	41%	34%	19%	19%	37%	0%
DIG	0%	11%	11%	21%	31%	43%	18%	19%	21%	24%	19%	6%	9%	12%	0%
DEA	5%	15%	24%	56%	75%	51%	88%	100%	80%	99%	88%	86%	93%	69%	76%
DDR	2%	11%	39%	14%	65%	84%	100%	59%	83%	98%	98%	100%	98%	78%	100%

APPENDIX D

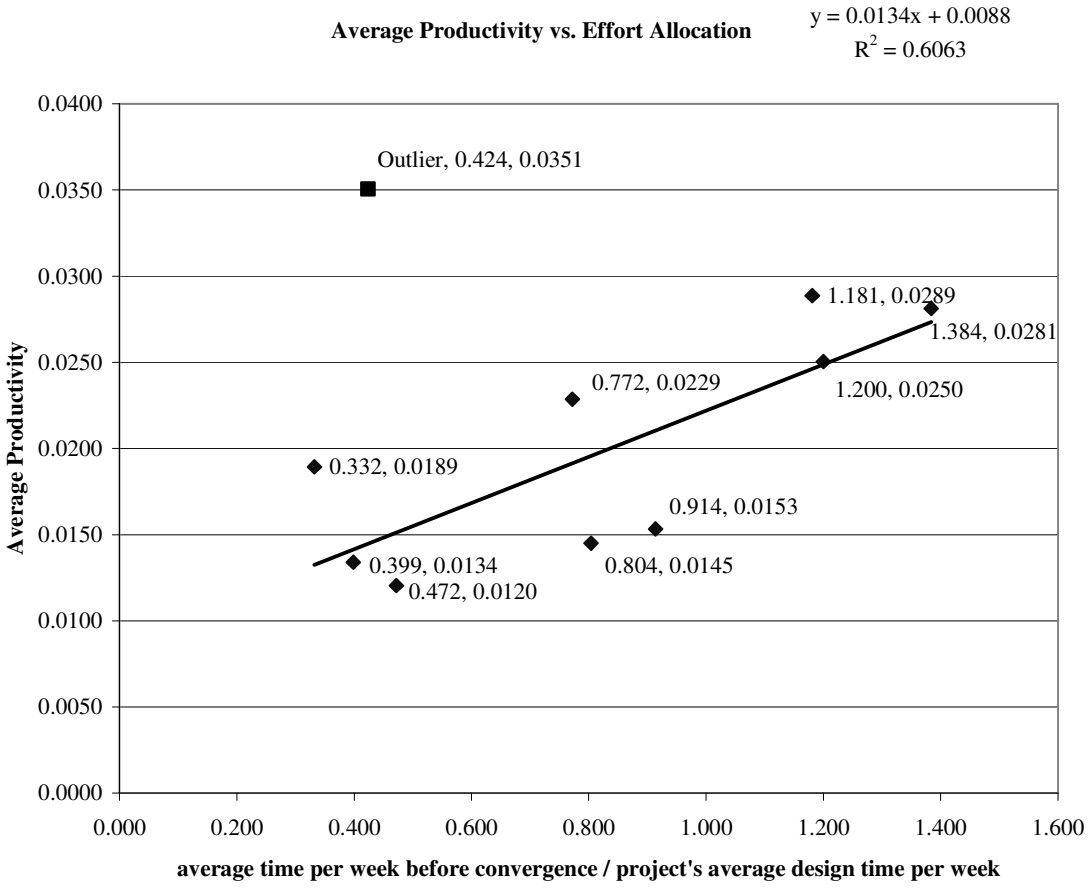
FRONT LOADING ANALYSIS

SAMPLE CLASSIFIED ON VALUE OF RATIO



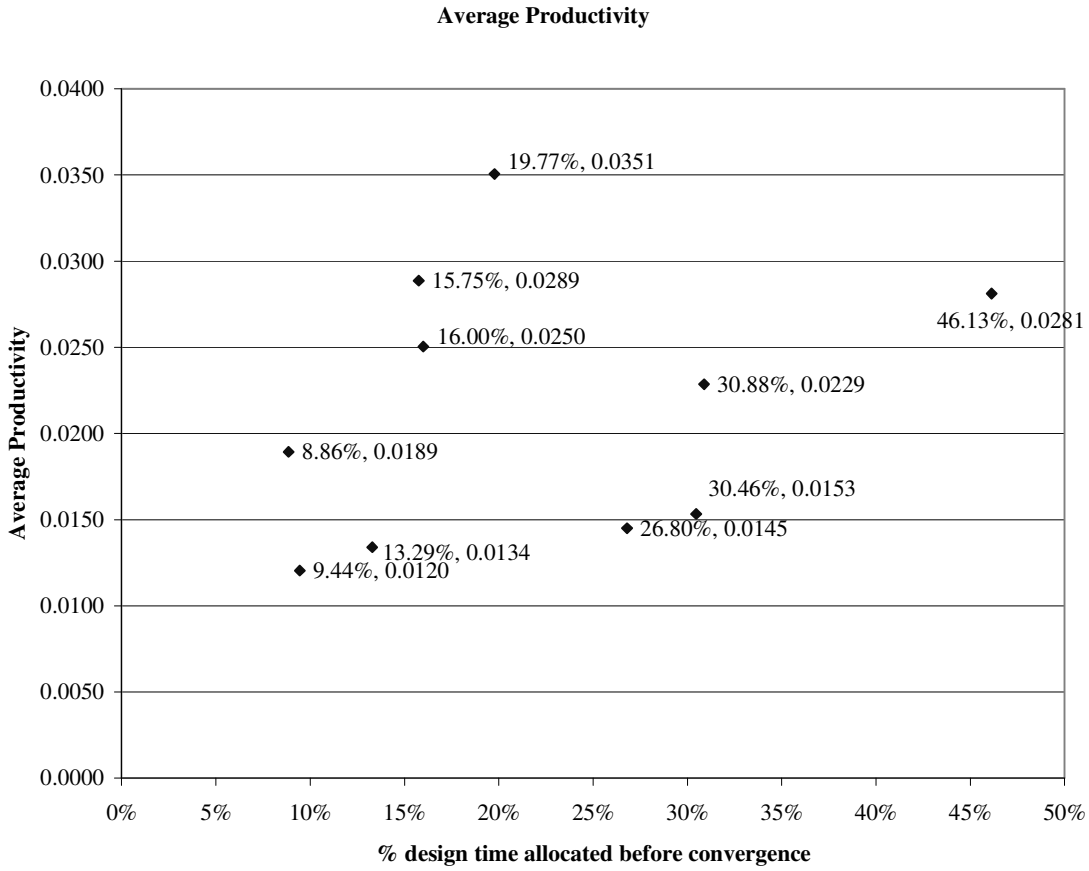
This t-test excludes the most productive team, which is an outlier, as will be apparent in the linear model of the ratio explaining productivity. The averages are significantly different for the two groups.

LINEAR MODEL OF RATIO EXPLAINING PRODUCTIVITY



The most productive team is a clear outlier in the association between the front-loading ratio and average productivity. The rest of the projects in the sample do present a linear pattern that explains 60% of the variance in productivity.

PERCENT DESIGN TIME BEFORE CONVERGENCE AND PRODUCTIVITY



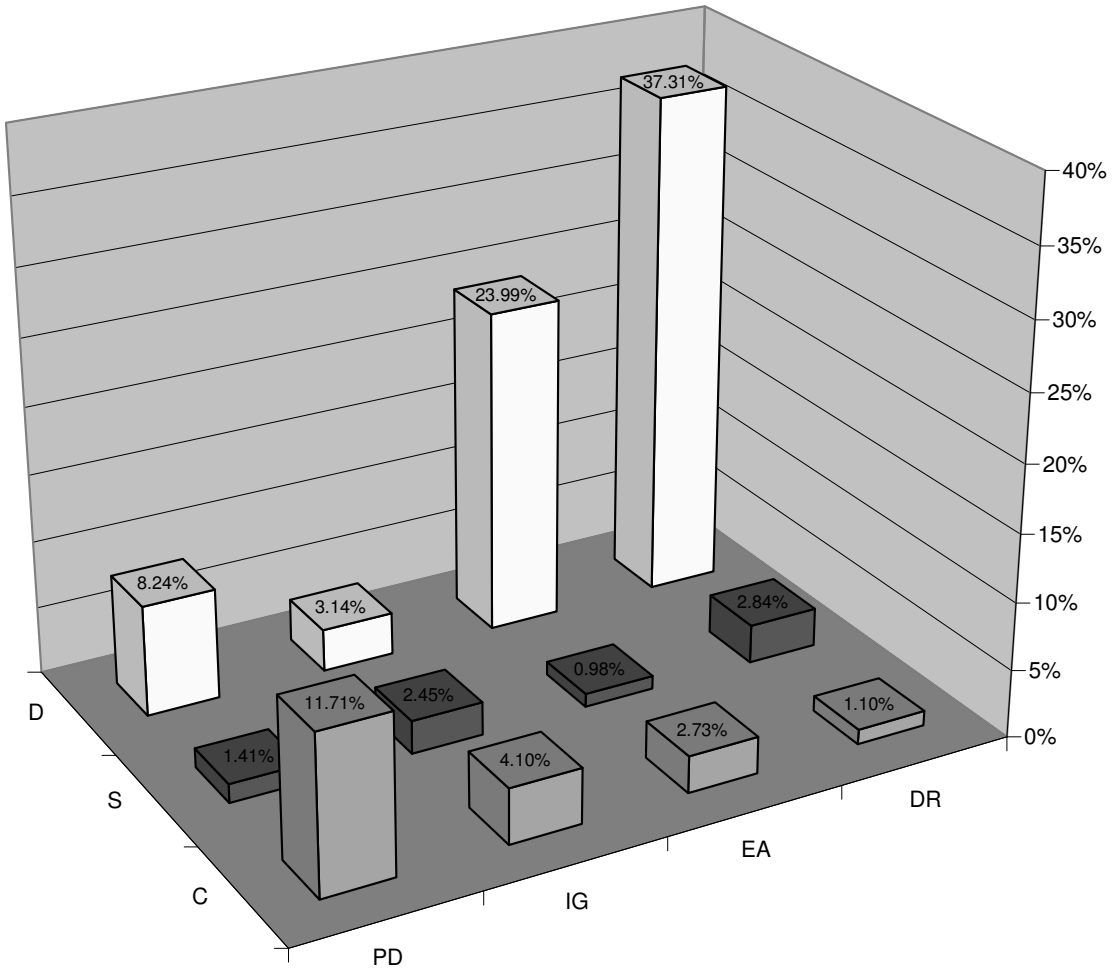
Percent design time spent before convergence does not present any pattern of association with productivity. Scope may play an important role as a discerning factor.

APPENDIX E

DETAIL LEVEL WORK AT BACK END

TIME DISTRIBUTION BY ACTIVITY AND DESIGN LEVEL

The chart visually indicates the percent of total time for the sample that corresponds to each activity. Note that 37.31% of the cumulative design time for the entire sample was allocated to D/DR. The second highest time allocation falls on D/EA. The combined effort allocation for the D/EA – D/DR combination is therefore 61.3% of cumulative design time for the sample. Almost two thirds of the time spent corresponds to only two of the twelve activities, indicating effort is not evenly distributed.



GROUP AVERAGES FOR EFFORT DISTRIBUTION

The sample can be divided in two groups based on their time allocation strategies. The shaded areas in gray indicate effort allocation of more than 80% for the time period, while the areas shaded in green are for effort allocations of more than 20%. Note the averages of percent effort allocation for the two groups differ in the activity that consumes more than 80% of the effort for a given time period. In both groups, the transition phase from weeks 4 to 6 inclusive presents a more spread allocation of effort. The two effort allocation strategies differ significantly on average productivity, as confirmed by a t-test. As seen in the previous chart, most effort is allocated to D/EA – D/DR, which makes the difference in effort allocation more substantial in absolute terms than for other activities.

Back End Dominated by D/EA – D/DR				**Rest**			
	Phases (weeks)				**Phases (weeks)**		
	1 - 3	**4 - 6**	**7 - 15**		**1 - 3**	**4 - 6**	**7 - 15**
C/PD - C/IG	69.4%	21.0%	1.1%	**C/PD - C/IG**	89.8%	46.0%	9.9%
C/EA - C/DR	1.9%	8.5%	0.2%	**C/EA - C/DR**	3.2%	3.9%	1.3%
S	10.6%	10.3%	2.9%	**S**	3.4%	20.1%	11.4%
D/PD - D/IG	9.8%	20.3%	8.0%	**D/PD - D/IG**	2.8%	24.8%	16.4%
D/EA - D/DR	8.3%	39.9%	87.7%	**D/EA - D/DR**	0.8%	5.2%	61.0%
Avg. Prod.	0.0180			**Avg. Prod.**	0.0294		