Aalto University
School of Science
Department of Industrial Engineering and Management

TANELI TUOMAS KANTOMAA

MANAGING ITERATIONS IN PRODUCT DEVELOPMENT

Master's thesis
Espoo, October 10, 2012

Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in engineering, Espoo October 10th 2012.

Supervisor Professor Paul Lillrank Dr.Sc. (Tech.)

Instructor Eero Tihilä M.Sc. (Tech.)

Aalto University School of Science
Department of Industrial Engineering and Management

ABSTRACT OF THE MASTER'S THESIS

Author: Taneli Tuomas Kantomaa

Title: Managing Iterations in Product Development

Title in Finnish: Iterointien hallinta tuotekehityksessä

Degree Programme: Industrial Engineering and Management

Major subject: Industrial Engineering and Management

Minor subject: Power Electronics

Chair (code): TU-22

Supervisor:
Professor Paul Lillrank, Dr.Sc. (Tech.)

Instructor:
Eero Tihilä, M.Sc. (Tech.)

Harmful iterations increase project lead time and make their scheduling difficult. On the other hand, iteration is not always undesired; some iteration is natural to product development and feedback providing iteration in the early phases of develoment may increase development speed. In this master's thesis causes for harmful iterations are recognized and classified, their impacts are diagnosed, and means to diminish their impacts are developed. Also, an iterative development model aiming at early, feedback providing iterations for avoiding later harmful iterations is presented.

This master's thesis starts out with a very comprehensive theoretical part covering theory about product development in general, Lean Product Development (LPD), process modeling and then moves on to cover theory about task level iterations. The theoretical part presents tools for modeling and analyzing the detailed design process in product development through a Design Structure Matrix (DSM) representation, network analysis or a matrix representation, and presents task overlapping principles. It also presents the concurrent engineering model and the spiral development model.

The empirical part analyzes the main causes for iterations in the company through System Dynamics (SD) approach. Causes for iteration are identified, elaborated and classified as well as many other areas needing improvement. The proposed iterative development framework aims at early feedback providing iterations for avoiding later harmful iterations. The process structure should reflect project uncertainties instead of forcing single process structure for all projects.

The achievements of this thesis, besides of classifying the causes of iteration and presenting insights in how to manage them, was describing how product development projects in the company proceed from planning to sales release, documenting events from present and past projects, and pointing out other issues that need addressing. This thesis could serve as a handbook for new project managers for familiarizing them on product development in the company as well as serving as a collected lessons learned document.

Date: 10.10.2012 | Language: English | Number of pages: 136+3

Keywords: iterations in product development, System Dynamics (SD), spiral development model, iterative development model, Lean Product Development (LPD), Concurrent Engineering (CE), Design Structure Matrix (DSM), Network Analysis (NA)

Aalto-yliopisto Perustieteiden korkeakoulu Tuotantotalouden laitos	DIPLOMITYÖN TIIVISTELMÄ
Tekijä: Taneli Tuomas Kantomaa	
Työn nimi: Iterointien hallinta tuotekehityksessä	
Title in English: Managing Iterations in Product Development	
Tutkinto-ohjelma: Tuotantotalouden tutkinto-ohjelma	
Pääaine: Tuotantotalous	Sivuaine: Tehoelektroniikka
Opetusyksikön koodi: TU-22	
Työn valvoja: Professori Paul Lillrank	Työn ohjaaja: DI Eero Tihilä

Haitalliset iteroinnit lisäävät projektien läpimenoaikaa ja tekevät niiden aikatauluttamisesta vaikeaa. Toisaalta jonkinasteinen iterointi on luontaista tuotekehitykselle ja tuotekehitysprosessin alkuvaiheessa tapahtuvat, epävarmuuksia poistavat iteroinnit nopeuttavat tuotekehitystä. Tässä diplomityössä tunnistetaan ja luokitellaan erilaiset haitalliset iteroinnit, arvioidaan niiden vaikutuksia ja esitetään keinoja niiden vaikutusten vähentämiseksi. Lisäksi tämä diplomityö esittää iteratiivisen kehitysmallin, joka tähtää aikaisiin epävarmuuksia poistaviin iterointeihin myöhäisempien haitallisten iterointien välttämiseksi.

Tämä diplomityö alkaa erittäin kattavalla teoriaosuudella. Teoriaosuus alkaa yleisellä tuotekehitysteorialla, kattaa Lean tuotekehityksen, prosessien mallintamisen ja jatkaa sitten tehtävätason iterointien teoreettisella tarkastelulla. Teoriaosuus esittää työkaluja yksityiskohtaisen suunnitteluprosessin mallintamiseen ja analysointiin Design Structure Matrix:in (DSM) avulla, informaatioverkkoanalyysin avulla tai matriisiesityksen avulla, ja esittää tehtävien rinnastuksen periaatteet. Teoriaosuus esittää myös Concurrent Engineering (CE)- mallin ja spiraalimallin.

Empiriaosuudessa iterointien pääsyyt tunnistetaan, niihin syvennytään ja ne luokitellaan systeemidynamiikan avulla. Sen lisäksi monia muita kehityskohteita tuodaan esille. Ehdotettu iteratiivinen kehitysmalli pyrkii aikaisiin epävarmuuksia poistaviin iterointeihin, joiden tarkoitus on estää myöhempiä haitallisia iterointeja. Prosessirakenteen tulisi heijastaa projektin epävarmuustekijöitä sen sijaan, että yksi ja sama prosessirakenne pakotetaan jokaiselle tuotekehitysprojektille.

Edellä mainittujen lisäksi tämän diplomityön saavutuksiin kuuluu tuotekehitysprojektien etenemisen kuvaaminen aina suunnitteluvaiheesta tuotteen myyntiin vapauttamiseen, nykyisten ja menneiden tuotekehitysprojektien tapahtumien dokumentointi, ja muiden kehityskohteiden tunnistaminen. Tämä diplomityö voisi toimia oppaana uusille projektipäälliköille yrityksessä ja tutustuttaa heidät tapaan, jolla tuotekehitysprojekteja viedään yrityksessä läpi, sekä toimia yhteenvetona menneiden projektien opeista yrityksessä.

Päivämäärä: 10.10.2012	Kieli: Englanti	Sivumäärä: 136+3

Avainsanat: iteroinnit tuotekehityksessä, systeemidynamiikka, spiraalimalli, iteratiivinen kehitysmalli, Lean tuotekehitys, Concurrent Engineering (CE), Design Structure Matrix (DSM), verkkoanalyysi

FOREWORD

First of all, I would like to thank my instructor Eero Tihilä for all the support and guidance that he has given me during my master's thesis. Eero has an honorable ABB work history of more than 40 years and relying on this experience he was able to provide valuable information for this master's thesis. Besides of the issues straightly related to my master's thesis Eero shared with me interesting knowledge about important events in the history of ABB and its predecessor Strömberg. This way, as Eero has now retired, he has passed on parts of the corporate knowledge to younger generations of ABB workers.

I would like to thank my supervisor Paul Lillrank who came up with the subject for this master's thesis. Instead of making a study on a subject that has been covered many times before in other master's thesis, the topic of this thesis was a novel one. This added enthusiasm in making this master's thesis.

I am also thankful for all the interviewees for their time and valuable comments concerning this master's thesis.

Most of all I would like to thank my parents. The support from my parents has been overwhelming throughout my studies and throughout my life. They have given me feeling of security by making me know that in every situation in my life, if needed, I can trust and count in their help.

Espoo, October 2012

Taneli Kantomaa

ABBREVIATIONS

R&D	**RESEARCH AND DEVELOPMENT**
NPD	**NEW PRODUCT DEVELOPMENT**
PDP	**PRODUCT DEVELOPMENT PROCESS**
LPD	**LEAN PRODUCT DEVELOPMENT**
CE	**CONCURRENT ENGINEERING**
IPD	**INTEGRATED PRODUCT DEVELOPMENT**
BPI	**BUSINESS PROCESS IMPROVEMENT**
BPR	**BUSINESS PROCESS REENGINEERING**
DSM	**DESIGN STRUCTURE MATRIX**
QDSM	**QUANTIFIED DESIGN STRUCTURE MATRIX**
WTM	**WORK TRANSFORMATION MATRIX**
NA	**NETWORK ANALYSIS**
CPM	**CRITICAL PATH METHOD**
PERT	**PROGRAM EVALUATION AND REVIEW TECHNIQUE**
GERT	**GRAPHICAL EVALUATION AND REVIEW TECHNIQUE**
CCPM	**CRITICAL CHAIN PROJECT MANAGEMENT**
SD	**SYSTEM DYNAMICS**

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

"Most companies die not because they do the wrong things, but because they keep doing what *used* to be the right things for too long"
-Yves Doz and Mikko Kosonen

Heightened international competition, demanding markets, and ever accelerating speed of technological change have forced companies to continuously examine and streamline the way they do business in order to remain competitive. Ever more demanding markets make it more and more important to be able to understand the needs of the customers through active interaction and to be able to communicate these needs within the company, and to be able to answer these needs in a timeliness fashion. Companies also operate with the knowledge that their competitors will inevitably come to the market with a product that changes the basis of competition. The ability to answer customer demands and changes of the business environment quickly and innovatively is one of the separating factors between successful and unsuccessful companies. In technology intensive industries it is the effectiveness and innovativeness of Product Development that makes up this ability. The speed of innovation realization and time-based competition are critical success factors in many dynamic industries.[1] [2]

The importance of a short lead time for product development is well illustrated in a study that if a product suffers from a 50% over expenditure in product development, the loss of total recoverable profit is 4%. However, if the product is late to market by 6 months for a life cycle of 5 years, it can lose one third of its profits. The development project is thus required to be on-time and predictable. The underlying design activities, however, are often interlinked and quite uncertain. For example iterations of a task or a set of tasks may occur when the results of the task(s) fail to meet specified criteria, or when new information is introduced (changes in customer demands, technical changes). Uncertain number of iterations and risks often introduces major uncertainties on the project's timeliness completion. [3] [4]

Iterations are not always undesired, though. In many cases they are inherent in the design process. An iterative process makes it possible to easily accommodate change, to obtain feedback and to factor it into the project, to reduce risk early, and to adjust the process dynamically[5]. Iterations frequently occur during product development to improve the product quality for satisfying customer needs[6]. However, iterations may improve the product performance while increasing development lead-time. Therefore, it is important to determine a proper process structure and to carefully manage iterations for a complex product

[1] Kessler, Eric H., and Paul E. Bierly. 2000. "Internal vs. external learning in new product development: effects on speed, costs and competitive advantage" *R&D Management* 30, no. 3: 213.
[2] Paul Trott (2005) *Innovation management and New Product Development* 3rd ed., Prentice Hall ISBN-10: 0-273-68643-7 ISBN-13: 978-0-273-68643-9
[3] Nichols, G., (1990) Getting engineering changes under control *Journal of Engineering Design* 1 (1)
[4] Peter B. Luh, Feng Liu, Bryan Moser (1997). Scheduling of design projects with uncertain number of iterations *European Journal of Operational Research*, 113, 575-592
[5] IBM, Rational method composer – Rational unified process, version 7.1, 2006
[6] Smith, R.P., Eppinger, S.D. (1997). A Predictive model of sequential iteration in engineering design. *Management Science*, 43(8), 1104-1120

development project. Poor process structure leads to futile iterations that may cause critical delays in getting products to market and great sales losses. [7] [8]

Innovation itself is a very broad concept. Most authors define innovation by suggesting that innovation is concerned with the commercial and practical application of *invention*. Invention, then, is the conception of the idea. Thus Innovation is the sum of theoretical conception, technical invention and commercial exploitation. Innovation depends on inventions but inventions need to be harnessed to commercial activities before they can contribute to the growth of an organization. Thus:

Innovation is the management of all the activities involved in the process of idea generation, technology development, manufacturing, and marketing of a new (or improved) product or manufacturing process or equipment.[2]

Research and Development (R&D) comprises two terms. To many the term research means the systematic approach to the discovery of new knowledge. Universities do not usually develop products. In industry, however, research is much more generic term and can involve both new science and the use of old science to produce a new product. So R&D is the management of scientific *research* and the *development* of new products. Sometimes it is difficult to determine where research ends and development begins. It is probably more realistic to view industrial R&D as a continuum with scientific knowledge and concepts at one end and physical products at the other. As we move right on this continuum we get closer to product development activities and the further we go from research activities, as figure 1.1 depicts.[2]

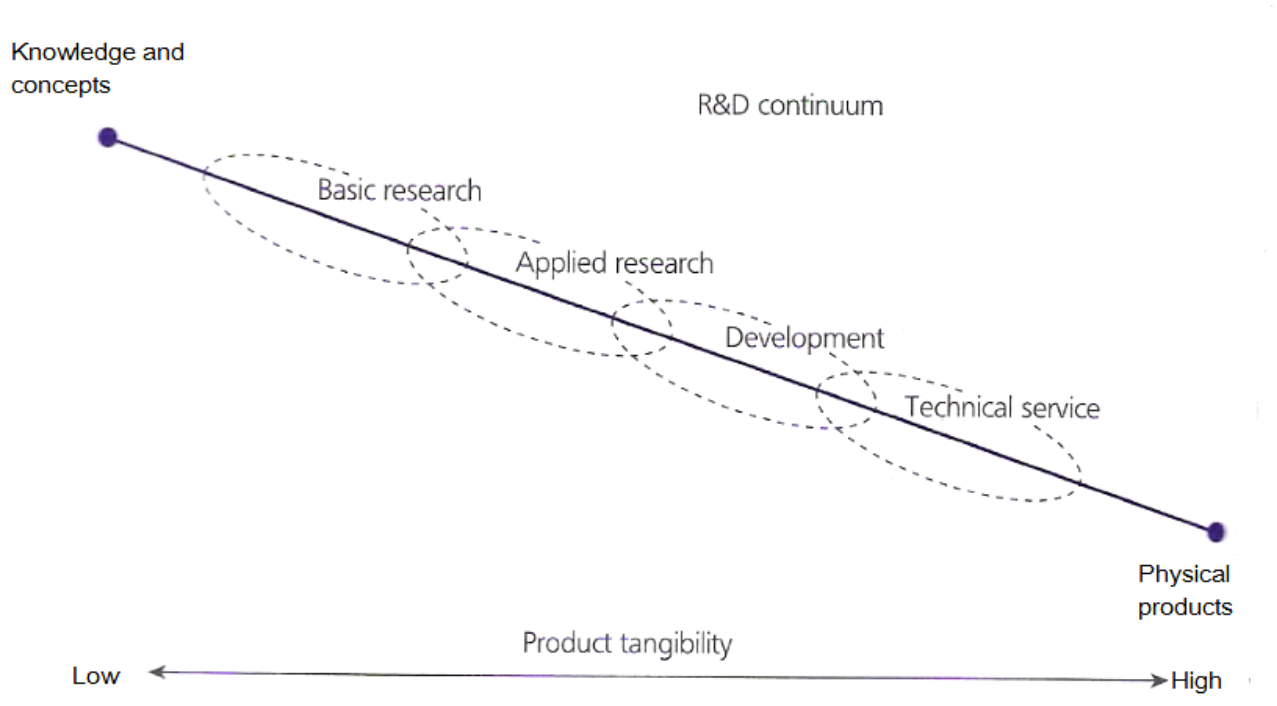


Figure 1.1 R&D continuum[2]

[7] Browning, T.R., Eppinger, S.D. (2002). Modeling impacts of process architecture on cost and schedule risk in product development. *IEEE Transaction on Engineering Management*, 49(4), 428-442

[8] Eppinger, S.D., (2001) Innovation at the speed of information. *Harvard Business Review*, 79(1), 149-158

R&D activities have changed dramatically in the past decades. There are numerous factors that have contributed to these changes of which three are of most importance:

- Technology explosion – most of our technological knowledge has been created during the past decades
- Shortening of technological life cycle – the technology cycle includes scientific and technological developments prior to the traditional product life cycle and these cycles have been continuously shortening
- Globalization of technology – technology transfer in the form of licensing and strategic alliances

Another major change has been the shift in emphasis within industrial R&D from an internal to an external focus. Traditionally R&D management has been management of internal R&D. The focus has shifted making external acquisition of technology (sourcing, acquisition of firms with technology), R&D strategic alliances (joint ventures, technology cooperation) and external R&D contracts (subcontractors, sponsored research at universities) the most important technology management issues in many multi-technology companies.

1.1 Background and motivation

The client of this study is a global actor in power and automation technologies. The products that it makes are targeted to improve the performance of its customers while lowering their environmental impact. The company operates in more than 100 countries and has offices in 87 of them to give its global and local customers the support they need in their daily businesses. The company has been seen as one of the pioneers in R&D internationalization. Small markets and R&D resources at the home country lead to the expansion of activities abroad. The company has a strong presence in Finland.

As is the case in many other product development environments, the lead times of product development projects in the company are hard to predict. The *naturally* unpredictable nature of product development projects is the reason for this. This master's thesis is concerned with improving the controllability (and through that predictability) of product development projects. If these projects could be executed the way they were planned without disturbances they could be managed like any other predictable projects. The fact that complex product development projects are naturally unpredictable and experience disturbances gives them their distinctiveness. Disturbances experienced during the execution of product development projects and deficiencies in the process lead to *iterations*. Iterations can also be natural to development; good examples of this are design iterations. In this master's thesis different kinds of iterations are recognized and classified, their impacts are diagnosed, and means to diminish their negative impact are developed. So in a way this study is all about *risk management*. This master's thesis is a part of a bigger development process that is taking place in the company which aims to improve the efficiency of product development.

1.2 Preliminary study

A study of a related issue was done earlier as a part of a special study course TU-22.1155 at Aalto University's Industrial Engineering and Management department the same year this study was commenced. The name of the preliminary study was "Developing the planning of product development projects in SAC-profit center". This previous study explored the current situation of planning and management of NPD projects in the company and gave improvement suggestions to those areas. However this study was more of a descriptive nature and it was not intended to be very comprehensive. Nevertheless, it gave a good basis for further research of NPD projects in the company.[9]

1.3 Objectives

The main research question and the auxiliary research questions of the study are presented below. A number of steps have to be taken in order to be able to answer to these questions. The methodology part of this study further explains how the study is to be conducted.

Main research question:

- **How to manage iterations in NPD to diminish their impact on project lead time and resource consumption?**

Auxiliary research questions:

- **What different kinds of iterations are there?**
  - **What causes them?**
  - **What are their features?**
  - **Which iterations are beneficial and which iterations are to be avoided?**
  - **What are their impacts on project lead time and resource consumption?**
- **Which tasks/phases should be done through iterations and which through the straight forward way?**

Examining how Lean principles are presently applied in the company is also a part of this study.

[9] Kantomaa, T.T., Halonen, T. *Developing the planning of product development projects in SAC-profit center (2011)*

1.4 Methodology

1.4.1 Planned progress of the study

The planned progress of the study is depicted in figure 1.2.

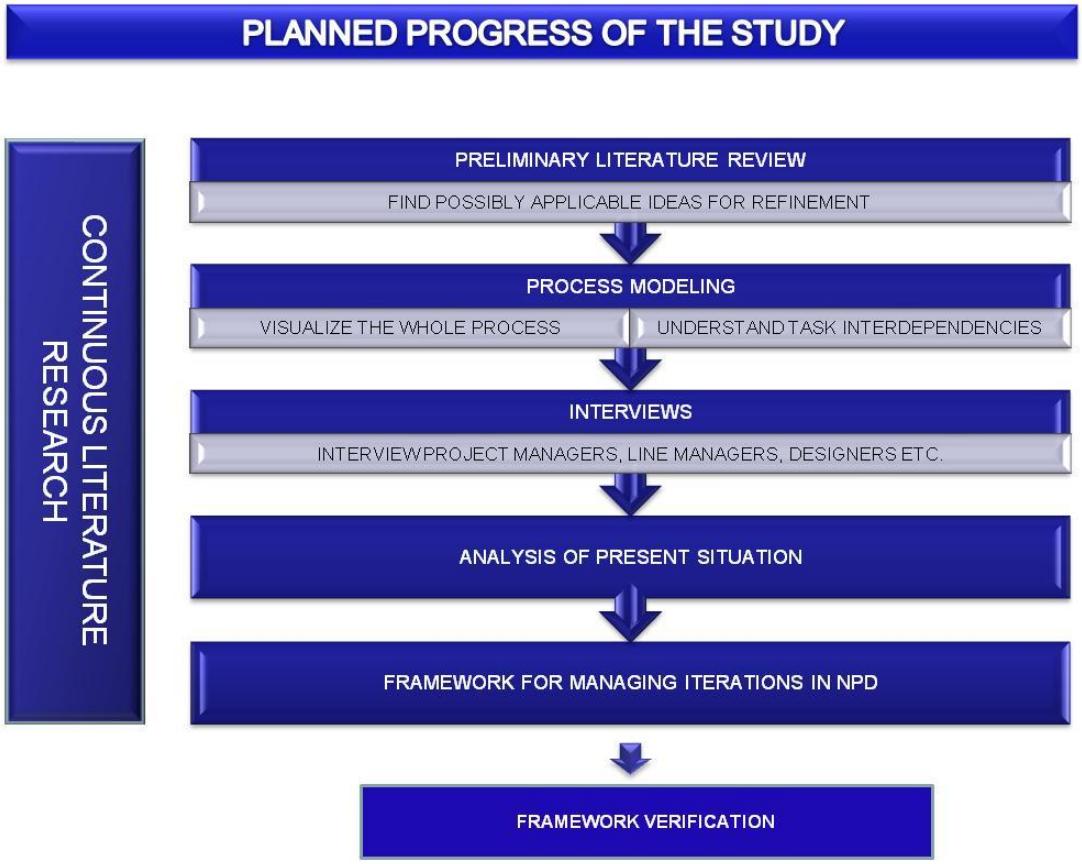


Figure 1.2 Planned progress of the study (Kantomaa, T.T.)

The first part of the study includes a preliminary literature review of the topic. Subjects as New Product Development (NPD), NPD processes, their evolution and contemporary trends will be covered. Also existing literature on iterations in NPD will be gone through although the initial presumption is that such literature is scarce. In the literature review only academic publications (books, journals) are to be reviewed. Academic publications will be searched in libraries, and with search engines utilizing only academic databases. This way the probability, that the material is reliable is enhanced. Also novelty of publications is considered; the aim is to review publications mainly from recent years.

In the next phase the Product Development Process (PDP) is modeled in the subject company. Generating a process map is essential for analysis purposes and the modeling of an entire NPD project has not yet been done in the subject company. There exists process models of different design functions but these are on a really rough level (except for mechanical design process) so they won't be helpful for in-depth analysis. The level of detail to be included in the model is yet unknown; the level of detail needed will be clarified as the work proceeds. The model is not supposed to be fully declarative meaning that not every task is to be

included separately in the model. Tightly connected tasks are represented as entities and the analysis will be based on this kind of process model.

In the process model functions outside the product development unit are also connected to product development tasks. These functions include product management, sourcing and manufacturing operations. There exists a tight connection between these functions and the product development process.

In order to get a reliable process model it will be generated with the help of members from every design team associated with the model. An attempt is also made to place the associated gates of the stage-gate process to the model according to chronological order.

The goal of modeling a NPD project is to create a process model that

- Depicts a NPD project on a sufficient level of detail
- Depicts the interdependencies between tasks
- Shows the chronological order of the tasks
- Is reliable

In the third part interviews are conducted. The interviewees will include at least project managers, line managers and designers of different functions. Also people from different units outside product development are interviewed, i.e. people from product management, sourcing and production. Before the interviews can be conducted enough knowledge of the product development process must be gathered in order to be able to ask productive questions. The structure of the interviews will be decided later.

The fourth part of the study will be to analyze the current situation and find areas of improvement. Here not only the iterations are considered but issues that have arisen during the study that should and could be improved in the product development unit are manifested.

In the fifth part the framework for managing iterations is generated and it is verified to see if the changes suggested by it could be implemented in practice.

1.4.2 Realized progress of the study

The theoretical part starts with general product development theory, covers Lean Product Development (LPD), process modeling and then moves on to cover theory about iterations. As was the initial assumption, there was no one comprehensive theory to be found on iterations in product development. During the theoretical study only academic publications (books, articles) were reviewed.

Search engines Nelli and Google Scholar were used to search through the following databases: Ebrary, EBSCOhost, Ellibs, Emerald Fulltext (Emerald), JSTOR, Knovel, Science Direct (Elsevier) SD, Wiley Interscience, Wiley Online Library. Used search words included: Iterations in product development, iterative development, System Dynamics (SD), Lean Product Development (LPD), agile product development, Graphical Evaluation and Review Technique (GERT), dynamic planning model, Network Analysis (NA), set-based approach, product development speed, process improvement etc.

When the empirical part of the study was commenced it became clear very soon that modeling the detailed design process even on a rough level would require so much time and effort that it was out of the reach of this study. Another approach was then chosen; System Dynamics (SD) was used for analysis purposes.

In the third part interviews were conducted. A total of 37 people were interviewed; 15 designers, 3 main designers, 9 project managers, 2 product managers, 1 line manager, 1 program manager, 1 product development manager and 5 people from other functions (sourcing, production, manual team). These interviews lasted from 1 hour to as long as 5 hours. The average time was somewhere around 2 hours. There is no complete interview template in the appendices, since all the interviews were individualized according to the work function and personal history of the interviewee. The interviews were semi-structured, which means the interview themes were known, but not all of the exact questions were known in advance of interviews. If all the questions and topics asked in different interview sessions were to be listed the list would go on for tens of pages so there is no sense in doing so. Appendix A lists some of the topics that were covered in the interviews.

Also a questionnaire on how designers spend their work time was conducted. A total of 17 designers from High Power Drives (4 from each team, except 5 from main circuit design team) responded to the questionnaire. The questionnaire form can be found in appendix B.

Documents found in corporate databases were also studied: specifications, product determination table documents, project descriptions, project schedules, project risk identification documents, project steering group meeting documents etc. All and all a comprehensive empirical research was made.

The fourth part of the study was to analyze the current situation and find areas of improvement.

In the fifth part the framework for managing iterations was developed. The main parts of the framework were verified by discussing them through with project managers and seasoned designers, and a document containing events from past projects was also made to give examples of situations where the framework would have proved to be useful in retrospective investigation. This is a separate document made in Finnish.

This thesis was based on theoretical, conceptual and qualitative analysis. The next step would be to try to quantify different iteration related phenomena. This would allow, for example, cost-worth analysis; what improvements would be expected from different development efforts and what would these development efforts require. Key Performance Indicators (KPI's) would reveal the actual improvements. They could then be compared to the expected ones to analyze any inconsistencies.

2 PRODUCT DEVELOPMENT

Product development is the set of activities beginning with the perception of a market opportunity and ending in the production, sale and delivery of a product.[10]

The above mentioned statement defines product development as a concept ranging broader than just design activities, which define the physical form of the product, consisting also from functions like marketing and manufacturing. These latter mentioned functions are actually present already in the design phase. When creating a new product the product development project starts with defining the product attributes, a process where marketing is strongly present. Broadly defined manufacturing function is concerned in designing, operating and/or coordinating the production system and also includes purchasing, distribution and installation. The production system design affects the product design and vice versa. The sourcing department is responsible for selecting the right suppliers taking into consideration cost, quality and availability of the sub-parts and these decisions also affect the design of the product. Trott (2005) suggests in his simultaneous coupling model that it is the simultaneous coupling of knowledge within all these three main functions that fosters innovation. Furthermore, the point of commencement for innovation is not known in advance. Several other functions, including sales and finance are also involved in the development of a new product. [2] [10]

Project team is the collection of individuals developing the product. This team usually has a single team leader; project manager. The team can be seen consisting of a core team and an extended team as figure 2.1 illustrates. In order to work effectively the core team is kept small enough, while the extended team may consist of dozens, hundreds or even thousands of other members. In most cases, a team within the firm will be supported by individuals or teams at partner companies, suppliers, and consulting firms.[10]

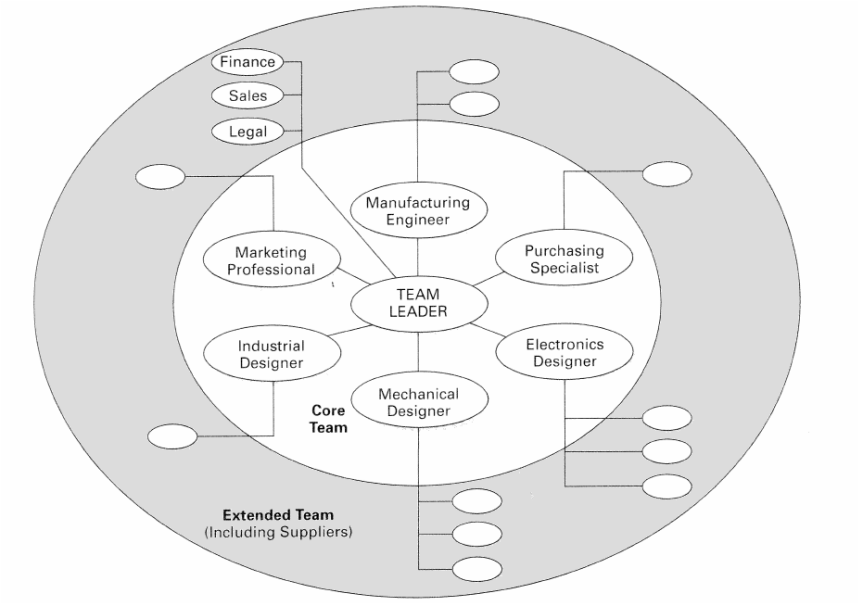


Figure 2.1 Composition of product development team for an electromechanical product of modest complexity[10]

[10] Ulrich, K.T., Eppinger, S.D. *Product Design and Development (2008) 4th ed.* McGraw-Hill ISBN: 978-007-125947-7

Product Development is challenging for many reasons. These include:

Trade-offs: An airplane can be made lighter but this action will probably increase manufacturing costs. To maximize the success of a product trade-offs must be recognized, understood, and managed properly.

Dynamics: Technologies improve, customer preferences evolve, competitors introduce new products, and the macroeconomic environment shifts.

Details: Developing a new product may require thousands of decisions on details of modest complexity.

Time pressure: Product development decisions must usually be made quickly and without complete information.

Economics: Developing, producing, and marketing a new product requires a large investment. To earn a reasonable return on this investment, the resulting product must be both appealing to customers and relatively inexpensive to produce.[10]

Every product development project is also distinct. It is because of the above mentioned attributes and of the uncertainties that distinctiveness implies that product development projects are so unpredictable in terms of time, costs and quality. This makes the success rate of NPD projects relatively low and the planning of product development activities difficult.

2.1 A Generic development process

A process is a sequence of steps that transforms a set of inputs into a set of outputs. A product development process is the sequence of steps or activities which an enterprise employs to conceive, design, and commercialize a product.[10]

Many of the steps and activities in the product development process are intellectual and organizational rather than physical. Some organizations define and follow a precise and detailed development process, while others may not even be able to describe their processes. A well defined development process is useful for the following reasons:

Quality assurance: A development process specifies the phases a development project will pass through and the checkpoints along the way. When these phases and checkpoints are chosen wisely, following the development process is one way of assuring the quality of the resulting product.

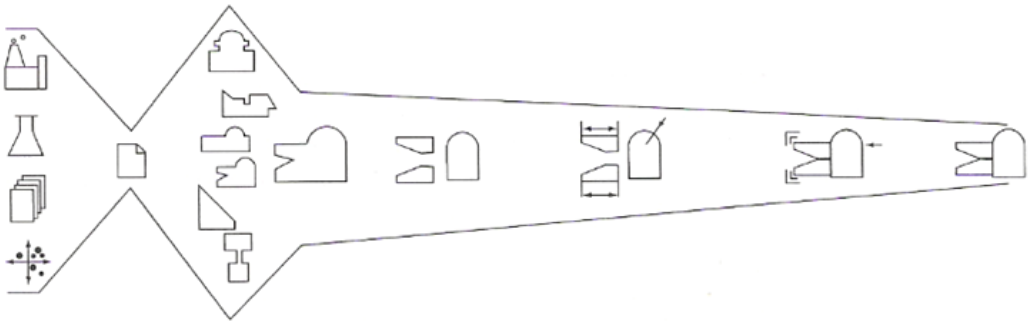
Coordination: A clearly articulated development process acts as a master plan which defines the roles of each of the players on the development team. This plan informs the members of the team when their contributions will be needed and with whom they will need to exchange information and materials.

Planning: A development process contains natural milestones corresponding to the completion of each phase. The timing of these milestones anchors the schedule of the overall development project.

Management: A development process is a benchmark for assessing the performance of an ongoing development effort. By comparing the actual events to the established process, a manager can identify possible problem areas.

Improvement: The careful documentation of an organization's development process often helps to identify opportunities for improvement.[10]

The generic product development process consists of six phases as illustrated in figure 2.2.



Phase 0: Planning	Phase 1: Concept Development	Phase 2: System-Level Design	Phase 3: Detail Design	Phase 4: Testing and Refinement	Phase 5: Production Ramp-Up
Marketing • Articulate market opportunity. • Define market segments.	• Collect customer needs. • Identify lead users. • Identify competitive products.	• Develop plan for product options and extended product family. • Set target sales price point(s).	• Develop marketing plan.	• Develop promotion and launch materials. • Facilitate field testing.	• Place early production with key customers.
Design • Consider product platform and architecture. • Assess new technologies.	• Investigate feasibility of product concepts. • Develop industrial design concepts. • Build and test experimental prototypes.	• Generate alternative product architectures. • Define major subsystems and interfaces. • Refine industrial design.	• Define part geometry. • Choose materials. • Assign tolerances. • Complete industrial design control documentation.	• Reliability testing. • Life testing. • Performance testing. • Obtain regulatory approvals. • Implement design changes.	• Evaluate early production output.
Manufacturing • Identify production constraints. • Set supply chain strategy.	• Estimate manufacturing cost. • Assess production feasibility.	• Identify suppliers for key components. • Perform make-buy analysis. • Define final assembly scheme. • Set target costs.	• Define piece-part production processes. • Design tooling. • Define quality assurance processes. • Begin procurement of long-lead tooling.	• Facilitate supplier ramp-up. • Refine fabrication and assembly processes. • Train work force. • Refine quality assurance processes.	• Begin operation of entire production system.
Other Functions • Research: Demonstrate available technologies. • Finance: Provide planning goals. • General Management: Allocate project resources.	• Finance: Facilitate economic analysis. • Legal: Investigate patent issues.	• Finance: Facilitate make-buy analysis. • Service: Identify service issues.		• Sales: Develop sales plan.	

Figure 2.2 Generic product development process[10]

The process begins with the planning phase, which is the link to advanced research and technology development activities. The output of the planning phase is the project's mission and statement, which is the input required to begin the concept development phase and which serves as a guide to the

development team. The conclusion of the product development process is the product launch, at which time the product becomes available for the markets.

One way to think about the development process is as the initial creation of a wide set of alternative product concepts and then the subsequent narrowing of alternatives and increasing specification of the product until the product can be reliably and repeatedly produced by the production system.

Another way to think about the development process is as an information-processing system. The process begins with inputs such as the corporate objectives and the capabilities of available technologies, product platforms, and production systems. Various activities process the development information, formulating specifications, concepts, and design details. The process concludes when all the information required to support production and sales has been created and communicated. [10]

A third way to think about the development process is as a risk management system. In the early phases of product development, various risks are identified and prioritized. As the process progresses, risks are reduced as the key uncertainties are eliminated and the functions of the product are validated.

Figure 2.2 also shows the activities and responsibilities of the different functions of the organization during each development phase. Because of the continuous involvement of marketing, design, and manufacturing in the product development process their roles are articulated here further.

0. Planning: This phase begins with corporate strategy and includes assessment of technology developments and market objectives. The output of the planning phase is the project mission statement, which specifies the target market for the product, business goals, key assumptions, and constraints.

1. Concept Development: In the concept development phase, the needs of the target market are identified, alternative product concepts are generated and evaluated, and one or more concepts are selected for further development and testing. A concept is a description of the form, function, and features of a product and is usually accompanied by a set of specifications, an analysis of competitive products, and an economic justification of the project.

2. System-level design: The system-level design phase includes the definition of the product architecture and the decomposition of the product into subsystems and components. The final assembly scheme for the production system is usually defined during this phase. The output of this phase includes a geometric layout of the product, a functional specification of each of the product's subsystems, and a preliminary process flow diagram for the final assembly process.

3. Detail design: The detail design phase includes the complete specification of the geometry, materials, and tolerances of all of the unique parts in the product and the identification of all of the standard parts to be purchased from suppliers. A process plan is established and tooling is designed for each part to be fabricated within the production system. The output of this phase is the control documentation - the drawings or computer files describing the geometry of each part and its production tooling, the specifications of the purchased parts, and the process plans for the fabrication and assembly of the product. Two critical issues addressed in the detail design phase are production cost and robust performance.

4. Testing and refinement: The testing and refinement phase involves the construction and evaluation of multiple preproduction versions of the product. Early (alpha) prototypes are usually built with **production-**

intent parts – parts with the same geometry and material properties as intended for the production version of the product but not necessarily fabricated with the actual processes to be used in the production. Alpha prototypes are tested to determine whether the product will work as designed and whether the product satisfies the key customer needs. Later (beta) prototypes are usually built with parts supplied by the intended production processes but may not be assembled using the intended final assembly process. Beta prototypes are extensively evaluated internally and are also typically tested by customers in their own use environment. The goal of beta-prototyping is usually to answer questions about performance and reliability in order to identify necessary engineering changes for the final product.

5. Production ramp-up: In the production ramp-up phase, the product is made using the intended production system. The purpose of the ramp-up is to train the work force and to work out any remaining problems in the production processes. Products produced during the ramp-up are sometimes supplied to preferred customers and evaluated carefully to identify any remaining flaws. The transition from production ramp-up to ongoing production is usually gradual. At some point the product is *launched* and becomes available for widespread distribution.[10]

The general development process described above is most likely to be used in a market-pull situation: a firm begins product development with a market opportunity and then uses whatever available technologies are required to satisfy the market need. There are other process types as well including: *technology-push* products, *platform* products, *process-intensive* products, *customized* products, *high-risk* products, *quick-build* products and *complex systems*. Figure 2.3 shows the key features of these process types.

Process Type	Description	Distinct Features	Examples
Generic (Market-Pull) Products	The team begins with a market opportunity and selects appropriate technologies to meet customer needs.	Process generally includes distinct planning, concept development, system-level design, detail design, testing and refinement, and production ramp-up phases.	Sporting goods, furniture, tools.
Technology-Push Products	The team begins with a new technology, then finds an appropriate market.	Planning phase involves matching technology and market. Concept development assumes a given technology.	Gore-Tex rainwear, Tyvek envelopes.
Platform Products	The team assumes that the new product will be built around an established technological subsystem.	Concept development assumes a proven technology platform.	Consumer electronics, computers, printers.
Process-Intensive Products	Characteristics of the product are highly constrained by the production process.	Either an existing production process must be specified from the start, or both product and process must be developed together from the start.	Snack foods, breakfast cereals, chemicals, semiconductors.
Customized Products	New products are slight variations of existing configurations.	Similarity of projects allows for a streamlined and highly structured development process.	Motors, switches, batteries, containers.
High-Risk Products	Technical or market uncertainties create high risks of failure.	Risks are identified early and tracked throughout the process. Analysis and testing activities take place as early as possible.	Pharmaceuticals, space systems.
Quick-Build Products	Rapid modeling and prototyping enables many design-build-test cycles.	Detail design and testing phases are repeated a number of times until the product is completed or time/budget runs out.	Software, cellular phones.
Complex Systems	System must be decomposed into several subsystems and many components.	Subsystems and components are developed by many teams working in parallel, followed by system integration and validation.	Airplanes, jet engines, automobiles.

Figure 2.3 Summary of variants of generic product development process[10]

2.2 Concept development

In this chapter special attention is paid to the concept development phase because of three reasons; it demands more coordination among functions than any other, it represents the *front-end process*, and it is in this phase of the product development process where the orientation of the whole project is made.

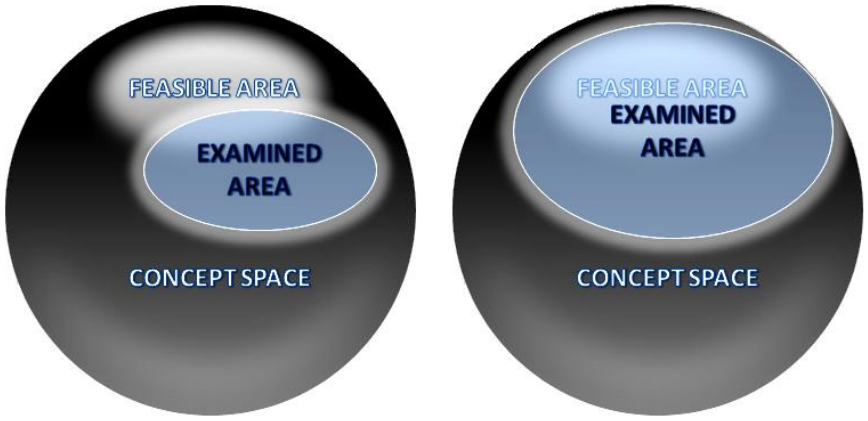


Figure 2.4 Concept space, feasible area and the examined area (Kantomaa, T.T.) *

* *It can be thought that the concepts under examination resemble one another on some level, i.e. they form a cluster in the concept space. The left side of figure 2.4 represents a situation where the cluster of examined concepts overlaps the feasible area only slightly which probably leads to suboptimal solutions. On the right examination of concepts has been done properly with enough time and resources and the whole feasible solution space has been covered.*

Concept development is the phase of the product development process where the orientation of the whole project is made. If this phase is not done properly and with enough time and capable resources, the project will not be successful no matter how well the subsequent phases will be done. The project will not be successful either in terms of quality (the solutions are probably suboptimal since the examination of the concept space hasn't been done properly), or in respect to cost and timeliness, since if the concept development is done poorly in the beginning of the project and the quality aspect is to be achieved then the concepts must be revised later on, and this will naturally increase the lead time and costs of the project. Figure 2.4 depicts one of the most important parts of concept development: the examination of the solution space.

The front-end process generally contains many interrelated activities, ordered roughly as presented in figure 2.5. Rarely does the entire process proceed in purely sequential fashion, completing each activity before beginning the next. In practice, the front-end activities may be overlapped in time and *iteration* is often necessary. At almost any stage new information may become available resulting in back-steps and repetitions of activities. This repetition of nominally complete activities is known as *development iteration*.

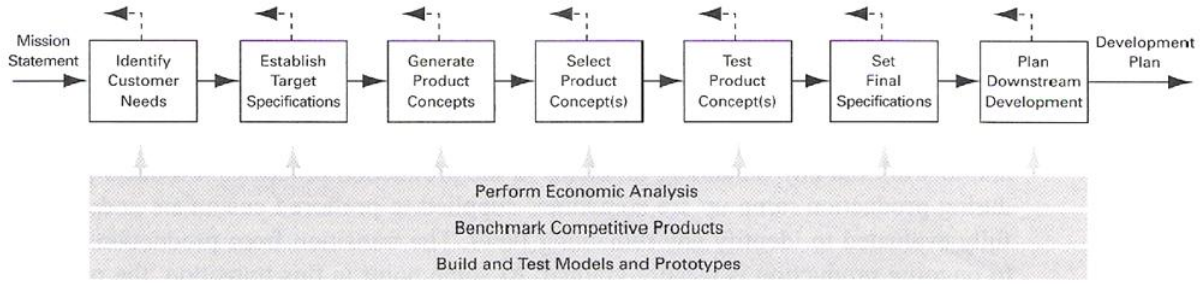


Figure 2.5 Concept development process[10]

The concept development process includes the following activities:

Identifying customer needs: The goal of this activity is to identify customer needs and to effectively communicate them to the development team. The output of this step is a set of customer need statements organized in a hierarchical list, with importance weightings for many or all of the needs.

Establishing target specifications: Specifications provide a precise description of what a product has to do. They are a translation of customer needs into technical terms.

Concept generation: The goal of concept generation is to thoroughly explore the space of product concepts that may address the customer needs. Concept generation includes a mix of external search, creative problem solving within the team, and systematic exploration of the various solution fragments the team generates. The result of this activity is usually a set of 10 to 20 concepts, each typically represented by a sketch and brief descriptive text.

Concept selection: Concept selection is the activity in which various product concepts are analyzed and sequentially eliminated to identify the most promising concept(s). The process usually requires several *iterations* and may initiate additional concept generation and refinement.

Concept testing: One or more concepts are then tested to verify that the customer needs have been met, assess the market potential for the product, and identify any shortcomings which must be remedied during further development.

Setting final specifications: The target specifications set earlier in the process are revisited after a concept has been selected and tested and the final specifications are set. Finalizing specifications is difficult because of trade-offs - inverse relationships between two specifications that are inherent in the selected product concept.

Project planning: At this stage a detailed development schedule is created, a strategy that minimizes the development time is formulated and the resources required to complete the project are identified.

Economic analysis: The team, often with a help of financial analyst, builds an economic model for the new product. This model is used to justify continuation of the overall development program and to resolve specific trade-offs among, for example, development costs and production costs.

Benchmarking competitive products: Understanding competitive products is critical for being able to successfully position new products. Benchmarking also provides a rich source of ideas for the product

design and production process design. Competitive *benchmarking* is performed in support of many of the front-end activities.

Modeling and prototyping: Every stage of the concept development process involves various forms of models and prototypes. These may include, among others: early "proof-of-concept" models, which help the development team to demonstrate feasibility; "form-only" models, which can be shown to customers to evaluate ergonomics and style; spreadsheet models of technical trade-offs; and experimental test models, which can be used to set design parameters for robust performance.

2.2.1 The activity of concept generation

A product concept is an approximate description of the technology, working principles, and form of the product. It is a concise description of how the product will satisfy the customer needs. The phase of concept generation is very important; a good concept is sometimes poorly implemented in subsequent development phases but a poor concept can rarely be manipulated to achieve commercial success. Concept generation is relatively cheap and can be done quickly compared to the rest of development process, so there is no excuse for lack of diligence and care in executing a sound concept generation method.

The concept generation process begins with a set of customer needs and target specifications and results in a set of product concepts of which typically 10 to 20 are selected for further scrutiny. Each concept is typically represented by a sketch and brief descriptive text. Good concept generation leaves the team with confidence that the full space of alternatives has been explored. Thorough exploration of alternatives early in the development process greatly reduces the likelihood that the team will stumble upon a superior concept late in the development process or that a competitor will introduce a product with dramatically better performance than the product under development.

STRUCTURED APPROACHES REDUCE THE LIKELIHOOD OF COSTLY PROBLEMS

Common dysfunctions exhibited by development teams during concept generation include:

- Consideration of only one or two alternatives, often proposed by the most assertive members of the team.
- Failure to consider carefully the usefulness of concepts employed by other firms in related and unrelated products.
- Involvement of only one or two people in the process, resulting in lack of confidence and commitment by the rest of the team.
- Ineffective integration of promising partial solutions.
- Failure to consider entire categories of solutions.

A way to reduce the incidents of these problems is a *structured* method approach. This method encourages the gathering of information from many disparate information sources, by guiding the team in the thorough exploration of alternatives, and by providing a mechanism for integrating partial solutions.

In figure 2.6 a five-step method is presented. The method breaks a complex problem into simpler sub-problems. Solution concepts are then identified for the sub-problems by external and internal search procedures. Classification trees and concept combination tables are then used to systematically explore the space of solution concepts to integrate the sub-problem solutions into a total solution. Finally, the team takes a step back to reflect on the validity and applicability of the results, as well as on the process used. Although the concept generation is presented as a linear sequence it is in fact almost always *iterative*.

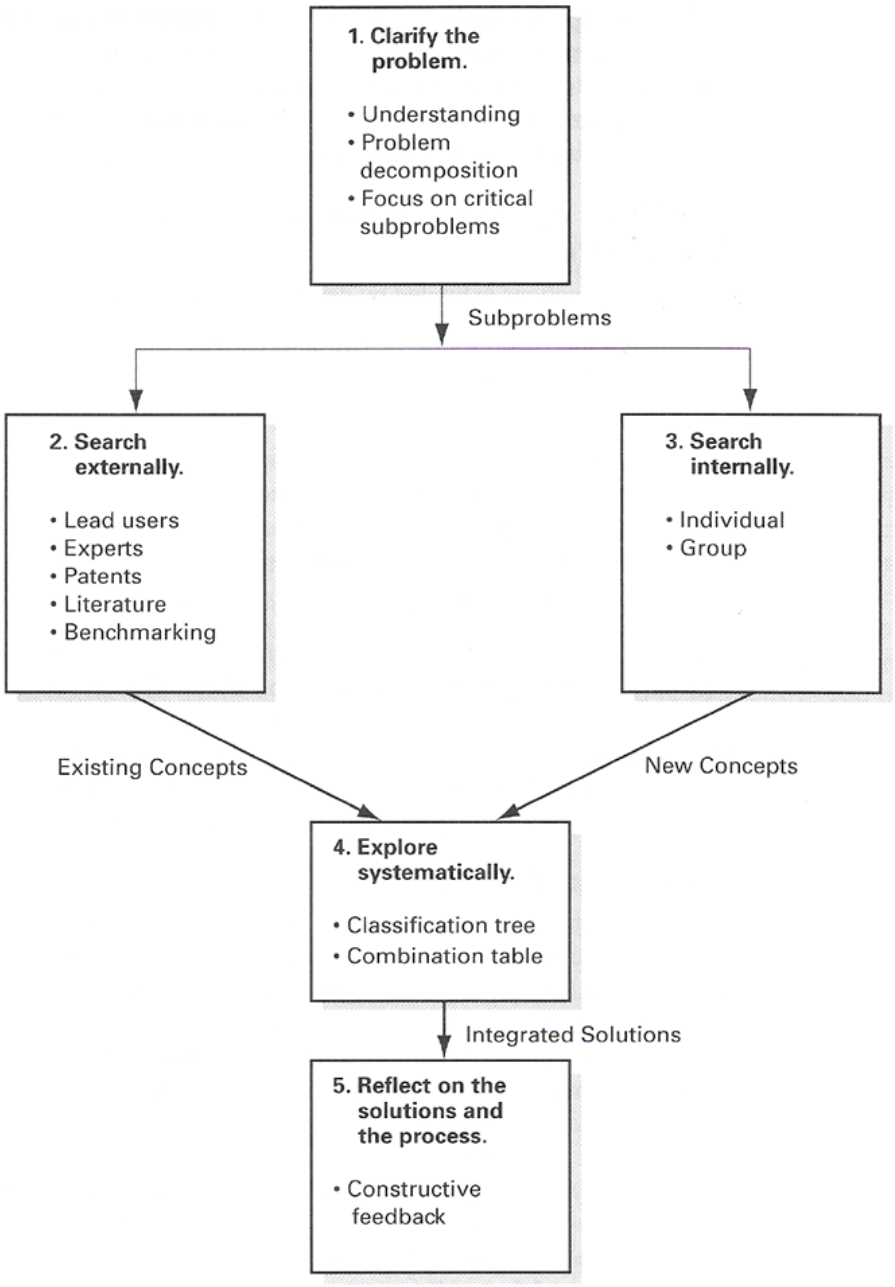


Figure 2.6 Five-step concept generation method[10]

Step 1*:* The problem is clarified by developing general understanding and breaking the problem down to sub-problems if necessary. Inputs for this step are mission statement for the project, the customer needs list, and the preliminary product specifications although these might still be under refinement as the concept generation begins. The problem might be divided into sub-problems according to functionalities, sequence of user actions or key customer needs. After the problem is divided into sub-problems the next thing to do is to focus on the ones that are critical to the success of the product and that are most likely to benefit from novel or creative solutions.

Step 2: External search is focused at finding existing solutions to both the overall problem and the sub-problems identified during the problem clarification step. Implementing an existing solution is usually quicker and cheaper than developing a new solution. This lets the development team to focus its creative energy on issues for which no satisfactory prior solutions exist. The external search for solutions is essentially an information-gathering process. Available time end resources can be optimized by using an expand-and-focus strategy; first *expand* the scope of the search to broad information gathering that might be related to the problem and then *focus* the scope of the search to by exploring the promising directions in more detail. Sources of external information include lead users, expert consultants, patents, literature, and competitive benchmarking. The main disadvantage of patent searches it that concepts found in recent patents are protected (generally for 20 years), so there may be a royalty involved in using them. However, patents are also useful to see what concepts are already protected and must be avoided or licensed. Concepts contained in foreign patents without global coverage and expired patents can be used without payment of royalties. There are several databases containing the actual text of patents that can be searched electronically by key words. Key word searches can be conducted efficiently with only modest practice and are effective in finding patents relevant to a particular product. In many instances a development team knows the competitors offerings quite well but the benchmarking of existing products that have different markets but related functionalities is neglected, just because they might be difficult to find. A conventional solution to one sub-problem can frequently be combined with a novel solution to another sub-problem to yield a superior overall design. For this reason external search includes detailed evaluation not only of directly competitive products but also of technologies used in products with related sub-functions.

Step 3: Internal search is the use of personal and team knowledge and creativity to generate solution concepts. The knowledge already exists and is in the possession of the team and it is just a matter of getting ideas to emerge from this knowledge base. This activity may be the most open-ended and creative of any in New Product Development (NPD). This process of retrieving potentially useful information to the problem at hand can be done either by individuals working in isolation or by a group of people working together. There are four guidelines for improving internal search:

1. Suspend judgment: Product concept decisions have consequences reaching out for many years so they must not be made hastily. Suspending evaluation for the days or weeks required is critical to success.

2. Generate a lot of ideas: Explore the whole solution space by generating a lot of ideas. When the quantity of ideas is large the expectations of quality for any particular idea is lowered, which leads to people sharing ideas they would otherwise not view as worth mentioning. Furthermore, each idea acts as a stimulus for other ideas.

3. Welcome ideas that may seem infeasible: Ideas which initially appear infeasible can often be improved, "debugged", or "repaired" by other members of team. In many instances the most creative ideas are originally seen as infeasible.

4. Use graphical and physical media: Text and verbal language are inherently inefficient vehicles for describing physical entities so graphical presentations and physical forms should be employed.

Step 4: As a result of the external and internal search activities, the team will have collected tens or hundreds of concept *fragments* – solutions to the sub-problems. Systematic exploration is aimed at navigating the space of possibilities by organizing and synthesizing these solution fragments. One alternative would be to go through all the combinations of these concept fragments, but little arithmetic reveals the impossibility of this idea; if there are three sub-problems each having 15 fragments there are 3375 (15x15x15) possible combinations. There are two specific tools for managing this complexity and organizing the thinking of the team: the *concept classification tree* and the *concept combination table*. The classification tree divides the possible solutions into independent categories while the combination table guides in selectively considering combinations of fragments.

Figure 2.7 shows a classification tree for a product under development. The branches correspond to different energy sources.

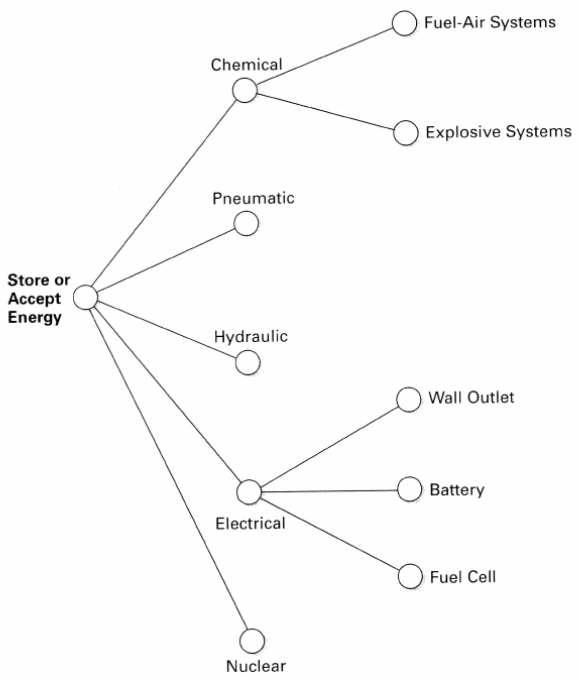


Figure 2.7 Classification tree according to energy source concept fragments[10]

The classification tree provides at least four important benefits:

1. **Pruning of less promising branches** – identify solutions that do not appear to have much merit, eliminate them and focus on the more promising branches of the tree.
2. **Identification of approaches to the problem** – Each branch of the tree is a different approach to the problem. Some of these approaches may be almost completely independent of each other. The division effort helps in organizing the concept generation activities among individuals or task forces.
3. **Exposure of inappropriate emphasis on certain branches** – Once the tree is constructed it can be seen very quickly whether the effort applied to each branch has been appropriately allocated i.e. have all solution branches been thought as well as should have been or has some branch been neglected.
4. **Refinement of the problem decomposition for a particular branch** – Sometimes problem decomposition can be usefully tailored to a particular approach to the problem. Consider the

branch of the tree corresponding to electrical energy source. What if none of the proposed solutions including wall outlet, battery or fuel cell are applicable (in terms of size, cost, and mass) for making enough *instantaneous* driving power for the product. Then the development team may conclude that energy must be accumulated over a substantial period. This analysis leads the team to add a sub-function ("accumulate translational energy") to their function diagram. They chose to add the sub-function after the conversion of electrical energy to mechanical energy, but briefly considered the possibility of accumulating energy in the electrical domain with a capacitor.

The classification tree can be constructed with branches corresponding to the solution fragments of any of the sub-problems, but in general a good candidate for a classification tree is a sub-problem whose solution highly constrains the possible solutions to the remaining sub-problems.

The concept combination table provides a way to consider combinations of solution fragments systematically. Figure 2.8 shows an example of combination table that can be used to consider the combinations of fragments for the electrical branch of the classification tree. There are 24 (4x2x3) possible combinations. One solution is to use a solenoid to compress a spring which is then released repeatedly to produce multiple impacts.

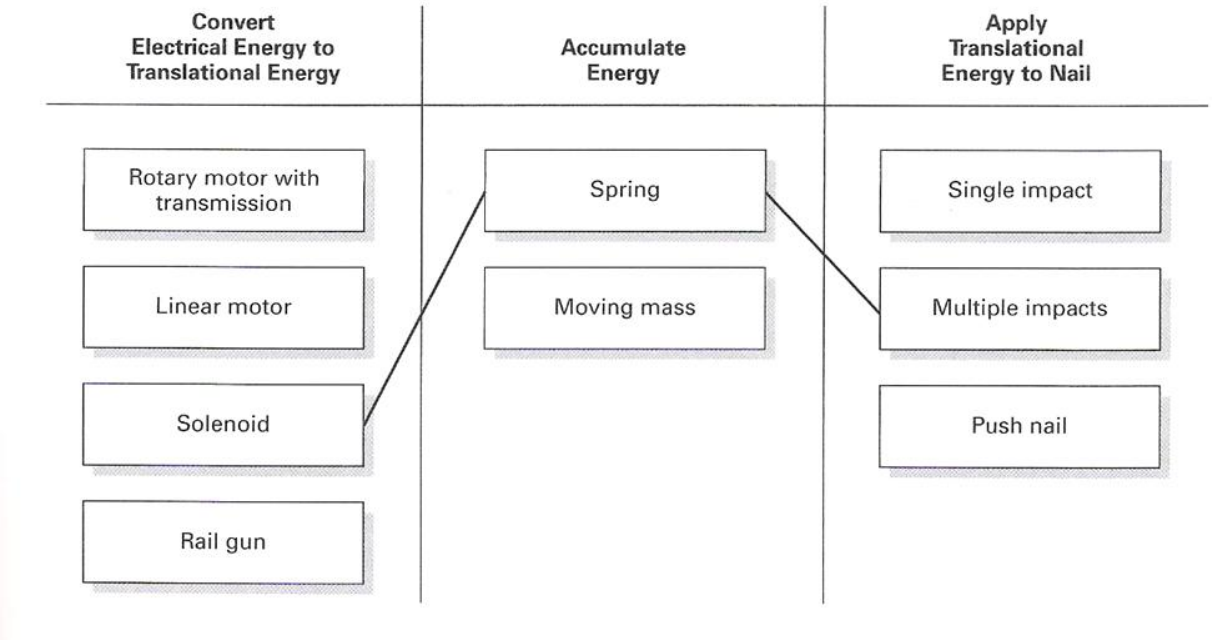


Figure 2.8 Combination table[10]

Step 5: Although the reflection step is placed here at the end the team should be performing reflection throughout the whole process. Questions to ask include:

- Is the team developing confidence that the solution space has been fully explored?
- Are there alternative function diagrams?
- Are there alternative ways to decompose the problem?
- Have external sources been thoroughly pursued?
- Have ideas from everyone been accepted and integrated in the process?

2.3 Prototyping

One definition of *prototype* is "an approximation of the product along one or more dimensions of interest". According to this definition, any entity exhibiting at least one aspect of the product that is of interest to the development team can be viewed as a prototype. This definition deviates from standard usage in that it includes such diverse forms of prototypes as concept sketches, mathematical models, simulations, test components, and fully functional preproduction versions of the product. *Prototyping* is the process of developing such an approximation of the product.

Prototypes can usually be classified along two dimensions; the degree to which the prototype is physical as opposed to analytical, and the degree to which a prototype is comprehensive as opposed to focused. Physical prototypes are tangible artifacts created to approximate the product. Analytical prototypes represent the product in a non-tangible, usually mathematical or visual, manner. Comprehensive prototypes implement most, if not all, of the attributes of a product. Focused prototypes implement one, or a few, of the attributes of a product.

A PROTOTYPE MAY REDUCE THE RISK OF COSTLY ITERATIONS

Figure 2.9 illustrates the role of risk and iteration in product development. In many situations the outcome of a test may dictate whether a development task will have to be repeated. In figure 2.9 an example of injection mold building is given. The probability of building a mold successfully without prototyping is 70%. When prototyping is introduced the probability for building a mold successfully has risen to 95%. If building and testing a prototype substantially increases the likelihood that the subsequent activities will continue without iteration then prototyping is justified.

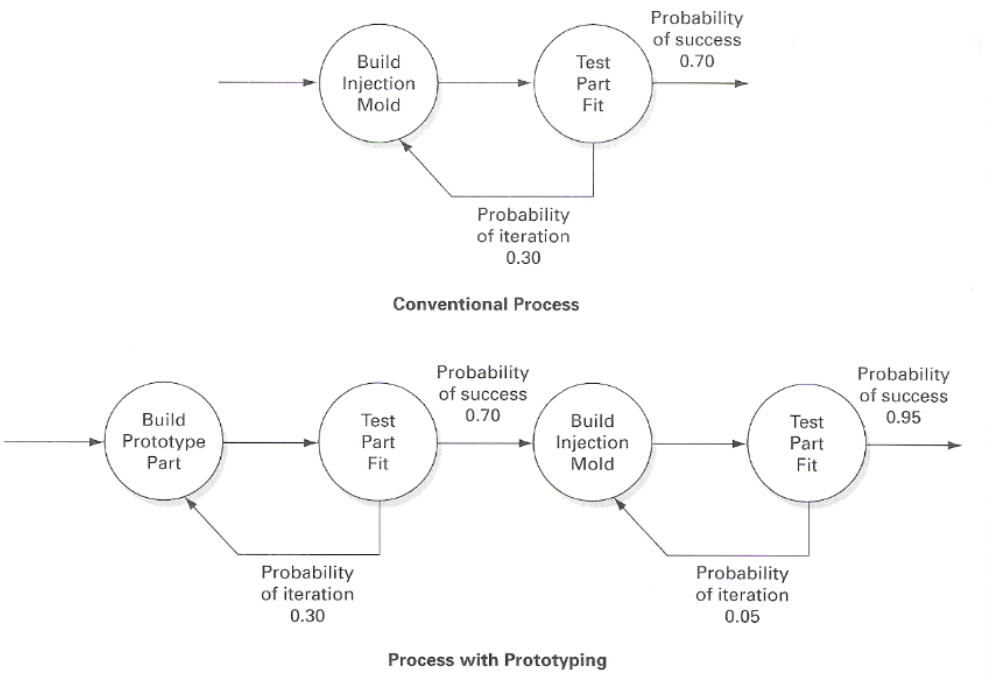


Figure 2.9 Probabilities of success with the conventional and the with the prototyping process[10]

The anticipated benefits of a prototype in reducing risk must be weighed against the time and money required to built and evaluate the prototype. Products that have high risks or uncertainties, because of the high costs of failure, new technology, or the revolutionary nature of the product, will benefit from such prototypes. On the other hand, products for which failure costs are low and the technology is well known do not derive as much risk reduction benefit from prototyping. Most products fall between these two extremes.

Sometimes the addition of a prototype may allow a subsequent activity to be completed more quickly than if the prototype were not built as figure 2.10 depicts.

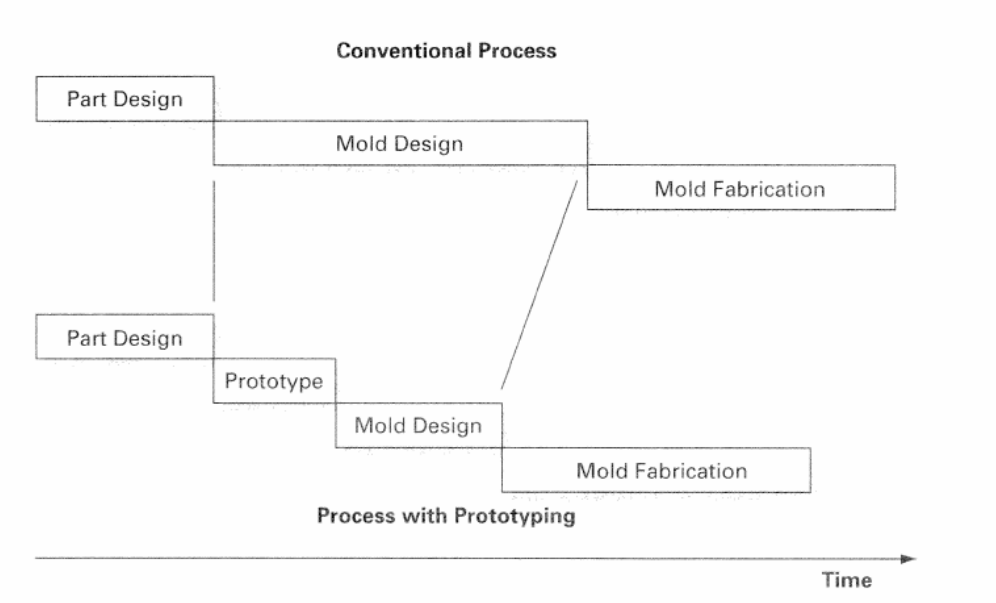


Figure 2.10 Building a prototype may enable more rapid completion of a subsequent step[10]

A prototype may restructure task dependencies as shown in figure 2.11. Here figure 2.11 illustrates tasks that are completed sequentially. It may be possible to complete some of the tasks concurrently by building a prototype. For example, a software test may be dependent of a physical circuit. Rather than waiting for the production version of the Printed Circuit Board (PCB) to use in a test, the team may built a prototype and use it for the test.

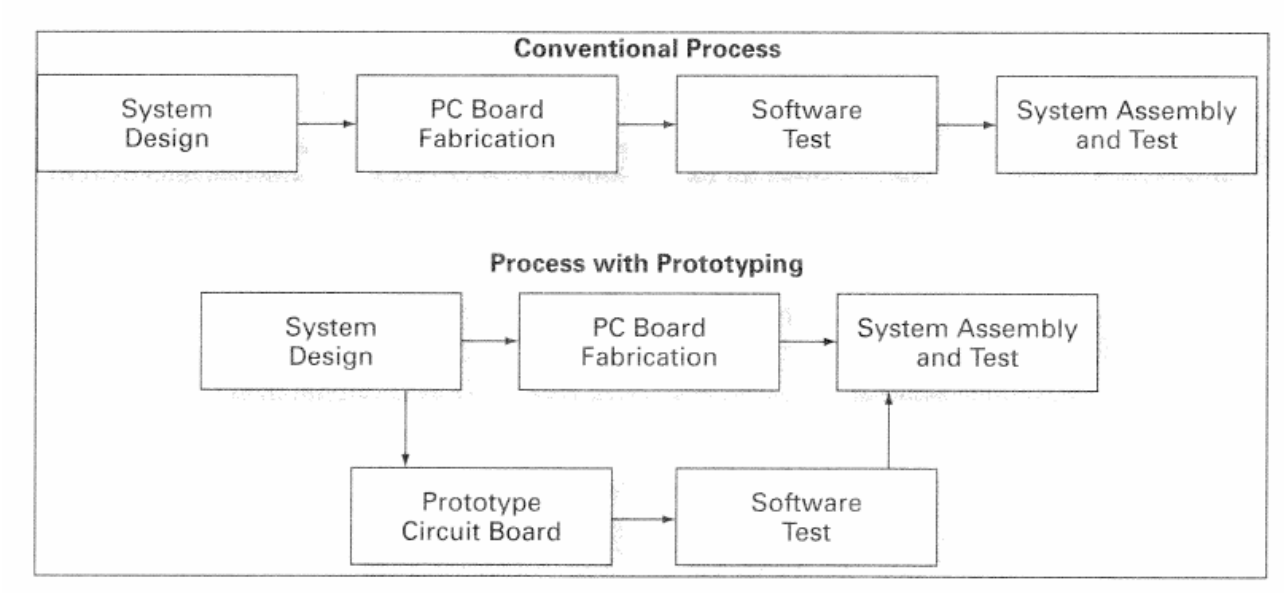


Figure 2.11 Use of a prototype to remove a task from the critical path[10]

In the past 20 years two technologies have become ever more important in prototyping activities: three-dimensional computer modeling (3D CAD) and free-form fabrication. The advantages of computer-aided design (CAD) include the ability to easily visualize the three dimensional form of the design. Through the use of computer-aided engineering (CAE) tools, 3D CAD models have begun to serve as analytical prototypes. In some settings this can eliminate one or more physical prototypes. In 1984 3D systems developed a technology used in free-from fabrication called stereolithography. This technology, and other technologies that followed it, create physical objects directly from 3D CAD models, and can be thought of as "three-dimensional printing". This collection of technologies is often called *rapid prototyping*. Most of the technologies work by constructing an object, one cross-sectional layer at a time, by depositing material or by using a laser to selectively solidify a liquid or powder. The resulting parts are most often made from plastics, but other materials are available, including wax, paper, ceramics, and metals. In some cases the parts are used directly for visualization or in working prototypes. However, the parts are often used as patterns to make molds or patterns from which parts with particular material properties can then be molded or cast. When used appropriately, prototypes created using free-from fabrication can reduce product development time and/or improve the resulting product quality.

2.4 Product development project types

There are four types of product development projects:

- ***New product platforms:*** This type of project involves a major development effort to create a new family of products based on a new, common platform.
- ***Derivatives of existing product platforms:*** These projects extend an existing product platform to better address familiar markets with one or more new products.
- ***Incremental improvements to existing products:*** These projects may involve adding or modifying some features of existing products in order to keep the product line current and competitive. A slight change to remedy minor flaws in an existing product would be an example of this type of project.
- ***Fundamentally new products:*** These projects involve radically different product or production technologies and may help to address new and unfamiliar markets. Such projects inherently involve more risk; however, the long-term success of the enterprise may depend on what is learned through these important projects.

A platform product is built around a preexisting technological subsystem (a technology platform). Product Platform has been defined by McGrath (1995) as "a set of subsystems and interfaces that form a common structure from which a stream of related products can be efficiently developed and produced"[11]. By using platforms a company can develop a set of differentiated products[12]. Example of such platform is the Apple Macintosh operating system or the tape transport mechanism in the Sony Walkman. Actually Sony walkmans 160 variations and four major technical innovations between 1980 and 1990 were all based on its initial platform. Huge investments were made in developing these platforms, and therefore every attempt is made to incorporate them into several different products. In some sense, platform products are very similar to technology-push products in that the team begins the development effort with an assumption that the product concept will embody a particular technology. The primary difference is that a technology platform has already demonstrated its usefulness in the market place in meeting customer demands. The firm can in many cases assume that the technology will also be useful in related markets. Products built on technology platforms are much simpler to develop than if the technology were developed from scratch. For this reason, and because of the possible sharing of costs across several products, a firm may be able to offer a platform product in markets that could not justify the development of a unique technology. [2] [10]

[11] McGrath, M.E. *Product Strategy for High-Technology Companies*, New York, Irwin Professional Publishing

[12] Wheelwright, S.C. and Clark, K.B. (1992) *Revolutionizing Product Development: quantum leaps in speed, efficiency and quality*, Free Press, New York

HOW APPLE DEVELOPS PRODUCTS

Apple has released many new products over the last decade over which only a few have been the start of a new platform. The rest were *iterations.* Apple develops its products through continuous iterative improvement. Good examples of these are the iPhone, iPod and one of Apple's oldest products Mac OS X. The iPhone debuted 2007 with no third-party apps, no 3G networking, and a maximum storage capacity of 8GB. One year later, Apple had doubled storage, added 3G and GPS, and opened the App Store. The year after that, it swapped in a faster processor, added a compass and an improved camera, and doubled storage again. The pattern repeats. There are never any major changes, but small changes that occur on a constant basis. [13]

This example of Apple's incremental development approach represents one component of Integrated Product Development (IPD).

An effective platform can allow a variety of derivative products to be created more rapidly and easily, with each product providing the features and functions desired by a particular market segment. Since platform development projects can take from 2 to 10 times as much time and money as derivative product development projects, a firm cannot afford to make every project a new platform. Figure 2.12 illustrates the leverage of an effective product platform. The critical strategic decision is whether the firm decides to develop a derivative product from an existing platform or develop an entirely new platform.

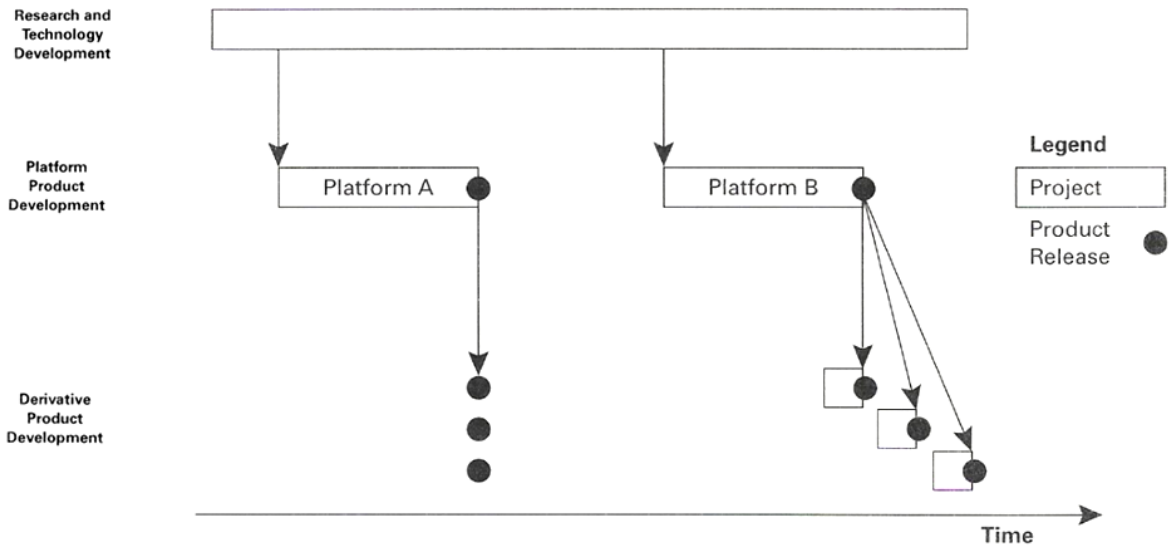


Figure 2.12 A platform development project creates the architecture of a family of products[10]

One technique for coordinating technology development with product planning is the *technology roadmap*. It presents the expected availability and future use of various technologies relevant to the product being considered. This method is especially useful in situations where the critical functional elements of products are well known in advance. The technological roadmap can be augmented with the timing of projects that would utilize these technological developments. The result is a *product-technology roadmap*.

[13] John Gruber *Apple's constant iterations* Macworld Apr2010, Vol. 27 Issue 4, p100-100 ISSN: 07418647

2.5 Stage-gate® system

There has been a variety of studies examining the relationship between formalization and innovation. There is some evidence of an inverse relationship between formalization and innovation, meaning that an increase in formalization may decrease innovative activity. It is unclear, however, whether a decrease in procedures and rules would lead to an increase in innovation. More over organizational planning and routines are necessary for achieving efficiencies. [2]

Using a formal process for product development has previously been demonstrated as increasing the probability that a product will be successful in the market place. Formal processes have also been discovered to reduce cycle times once the physical design of the product begins and reduces it even more for more complex products. As products grow more complex, with more functions designed into the product and designed to work together, the task of organizing the interfaces and interactions between different functions may grow geometrically or exponentially rather than linearly, thereby increasing the need for formal process. Processes do not help projects proceed efficiently through the fuzzy front-end of product development. [14]

Also, comparative studies of process performance indicate that having a formal new product process does not in itself guarantee performance, rather that it is the quality of this process in respect of several key activities which counts[15].

Stage-Gate® System was developed by Robert G. Cooper in 1988, and it is a widely used system by companies developing new products[16]. It is a conceptual and operational road map for moving a new-product project from idea to launch. Stage-Gate® divides the effort into distinct stages separated by management decision gates (gate-keeping). In each stage, a variety of activities (i.e. marketing, technical, and financial) are undertaken concurrently. The primary purpose of the stages is to solve problems and to generate information that will facilitate decision making about the new product (e.g., Is there a market for this new product? How should we design and produce it? Is it financially viable to produce this product?). A typical stage-gate® system is depicted in figure 2.13. [17] [18]

[14] Abbie Griffin, The Effect of Project and Process Characteristics on Product Development Cycle, *Time Journal of Marketing Research,* Vol XXXIV (February 1997), 24-35

[15] S. Minderhoud, P. Fraser, "Shifting paradigms of product development in fast and dynamic markets", *Reliability Engineering & System Safety,* 88 (2005) 127-135 doi: 10.1016/j.ress.2004.07.002

[16] New Product Dynamics website http://www.europa.com/~preston/stage_gate.htm cited 9.1.2012 and 10.1.2012

[17] Robert G. Cooper, *Perspective: The Stage-gate® Idea-To-Launch Process – Update, What's new, and NexGen Systems,* Product Innovation Management 2008;25;213-232

[18] Jeffrey B. Schmidt, Kumar R. Sarangee, and Mitzi M.Montoya, "Exploring New Product Development Project Review Practices", *J Prod Innov Manag 2009;26;520-535 ©PDMA*

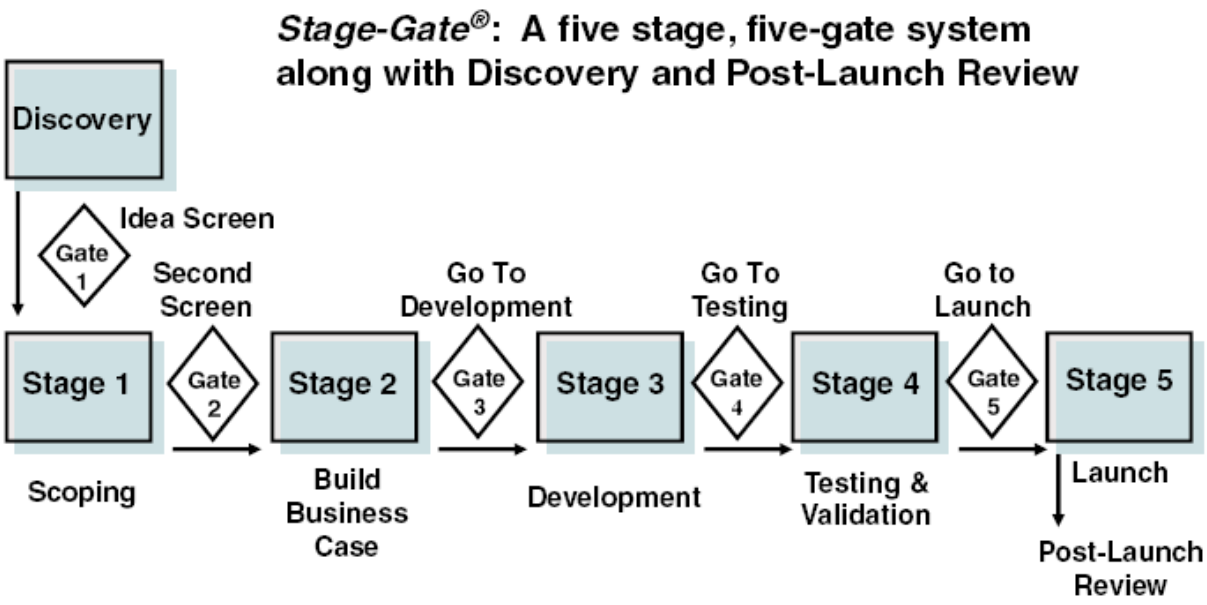


Figure 2.13 Overview of a typical Stage-gate® system [17]

The key stages are:

Stage 0 Discovery: Activities designed to discover opportunities and to generate new product ideas. After a decade of development focused on product extensions and quick hits, the quest for the super-idea – the "home-run", breakthrough idea or major innovation – has become a vital management issue. What some companies are doing is replacing the traditional ideation stage with a much more proactive "Discovery stage".[19]

Stage 1 Scoping: A quick and inexpensive assessment of the technical merits of the project and its market prospects.

Stage 2 Build Business Case: This is the critical homework stage - the one that makes or breaks the project. Technical, marketing and business feasibility are accessed resulting in a business case which has three main components:

- product and project definition
- project justification
- project plan

Stage 3 Development: Plans are translated into concrete deliverables. The actual design and development of the new product occurs, the manufacturing or operations plan is mapped out, the marketing launch and operating plans are developed, and the test plans for the next stage are defined.

Stage 4 Testing and Validation: The purpose of this stage is to validate the entire project:

- the product itself
- production/manufacturing process

[19] Dr. Robert G. Cooper, Dr. Scott J. Edgett, Dr. E.J Kleinschmidt, Optimizing the Stage-gate® process: What Best Practice Companies Are Doing, Part 1, Working paper No. 14 ©2006 The Product Development Institute

- customer acceptance
- economics of the project

Stage 5 Launch: Full commercialization of the product - the beginning of full production and commercial launch. [20]

At each gate go/kill decisions are made by a steering committee either to continue to invest in the project or to kill the project. Gate meetings include going through *must-meet criteria* (checklist) designed to weed misfit projects quickly, and *should-meet criteria* that are scored and added (a point-count system), which are used to prioritize projects[17]. These include strategic, technical, market, financial and operational criterion.

According to Robert G. Cooper (the developer of the system) Stage-gate system brings the following benefits for companies using it: Reduced time-to-market (by about 30 percent), improved product success rates, improved discipline over a complex and potentially chaotic process, and reduced re-work efforts[21].

The stage-gate® system is not a functional process nor is it a project control mechanism. It is a business process and a set of formal procedures for ensuring no critical steps are omitted in project execution. [17]

DEFICIENCIES AND CRITICISM ABOUT THE STAGE-GATE SYSTEM

A lot of criticism has been made in the contemporary literature concerning the rigidity of stage-gate system. Stages-and-gates break work up into sequential phases, thus preventing parallel, overlapping activities, when they cross the decision points. Such processes do not encourage completing tasks in earlier phases to keep them off of the critical path. [16]

At some instances the Stage-gate system has also been accused of being too bureaucratic, which encourages a lot of non-value adding activity. The Stage-gate system has become an end itself in some businesses, as teams go great lengths preparing for gate meetings. In other situations there are too many gates, and too many stages; more than the necessary ones and so the bureaucracy sets in. [22]

Another shortcoming of stages-and-gates processes is that they inherently force fundamental project decisions to be made earlier than necessary, thereby restricting your flexibility to respond to change and raising your cost of change. If change in markets, customer desires, technology, or management direction is characteristic of your development projects, more flexible development techniques should be considered, rather than phased approaches. [16]

[20] Product Development Institute, website http://www.prod-dev.com/stage-gate.php cited 9.1.2012
[21] Dr. Robert G. Cooper, Doing it Right: Winning with New Products, working paper No. 10, The Product Development Institute Inc. ©2006
[22] Cooper, R.G., "The invisible success factors in product innovation", *Journal of Product Innovation Management,* 16, 2, April 1999, 115-133 ©2006 The Product Development Institute

abstract of an article published by Don Reinertsen in June 2002

Don't ask, don't tell

Among the traditional rituals of product development, one of the strangest is the checkpoint review. The stated purpose of this review is to determine whether a project is ready to pass into the next phase. The checkpoint is manned by management, playing the role of astute border guards with unlimited power… Travelers, played by project team members, produce colorful documents for the border police. If they are worthy they are permitted to cross the border.

Officially, the checkpoint stops unworthy projects from entering the next phase. In practice, in most companies, all travelers eventually cross the border, but some are delayed more than others. Or at least they appear to be delayed. The secret of this ritual is that work almost always begins on the next phase before the checkpoint review is completed. Team members pretend to wait for the approval of management. Management pretends not to notice that work has already begun on the next phase. The border guards threaten to stop unworthy projects, but never use this power. The travelers pretend to wait at the border, but actually do not.

What is really going on? It is important to understand the real purpose of the ritual. It has nothing to do with decision-making and everything to do with power. The ritual is designed to reinforce the belief that management is in control. Travelers respect the border guards. As long as everyone follows the unwritten rules, out-of-phase activity is tolerated.

What are the benefits of this ritual? My surveys of product developers suggest that at least 75 percent of the companies that claim to work in one phase at a time actually permit out-of-phase work. They do so for good reason. When time-to-market is valuable, the benefit of overlapping phases far exceeds its cost.

Is this ritual simply a harmless fantasy, or does it have a dark side? I believe it is dangerous for two reasons. When actual behavior in an organization differs from its rules, people are confused about the importance of following the rules. New members of the organization become productive slowly, since they must de-code which rules count and which can be safely ignored. They tread this minefield with care, assuming there are more mines than there really are. This undermines initiative, which is so vital to success in product development.

Second, any distortion of reality is dysfunctional. If we attribute product development success to the wrong cause, we may optimize the wrong thing… If we believe that management makes all important decisions, it follows that only management needs access to quality information. If we recognize that all team members make important decisions on when to engage in out-of-phase activity, it follows that they need the right decision support information to make these decisions. Out-of-phase activity works when team members understand what is on the critical path, and when it makes economic sense to move forward. They must be able to quantify the costs and benefits of out-of-phase activity. Unfortunately, in many organizations low-level team members have no knowledge of these economics… You need to put the information where the decision-making is and if you delude yourself as to who makes decisions you will inevitably put the information in the wrong place… Ask yourself whether this path (invisible path around the checkpoint) should remain dangerously unmarked or whether it should be legitimized as one more tool in our toolkit. [23]

Don Reinertsen, President of Reinertsen & Associates, co-author in "Developing Products in Half the Time: New Rules, New Tools" and author of Managing the Design Factory.

[23] Don Reinertsen, *Product Development – Best Practices Report* (ISSN 1049-8400) ©2002 Management Roundtable Inc.

NEXGEN STAGE-GATE® SYSTEM FOR COUNTERING THE EMERGED DEFICIENCIES

Robert G. Cooper has introduced the NexGen Stage-gate® system for countering the deficiencies that have emerged from practice. The typical proficient new product Stage-gate® system contains about four or five stages and gates (not counting for the ideation stage and post launch review). According to Cooper the Stage-gate system must be modified according to project attributes. Projects that contain less risk deserve a system that has less stages (stages collapsed and gates combined) and has detours designed to speed up the process. This way the Stage-gate® system becomes scalable. According to Cooper, flexibility is obtained by allowing the project team considerable latitude in deciding what actions are really needed and what deliverables are appropriate for each gate. Cooper also encourages overlapping stages to employ simultaneous execution: not waiting for a completion of a previous step and perfect information before moving ahead. [22] [24] [25]

[24] Dr. Robert G. Cooper, "The seven principles of the latest Stage-gate® method add up to a streamlined, new-product idea-to-launch process", MM March/April 2006

[25] Brombacher AC, de Graef M., den Ouden E., Minderhoud S., Lu Y., "Reliability of technical systems and changes in organizational processes", Bedrijfkunde 2001;73(2):49-58 (in Dutch)

2.6 Lean product development

Applying lean to product development environment results in a different set of guiding principles with respect to the original lean manufacturing principles. Product development is different from manufacturing. Here, value may be defined differently and the value stream consists of flows of information and knowledge that are less easily tracked than the material flows of a factory. In manufacturing the ability to mass customize means building the product out of modules which can be combined in different ways to achieve differentiation. In product development by using a platform approach, a company can develop a set of differentiated products[12]. Instead of physical modules, modularity can also mean knowledge modules. Through the reuse of knowledge modules efficiency and increased NPD speed can be achieved. The basic idea is to capture and re-use the *engineering intent* during a product development process. Using modularity in architecture is one example. Product architecture is the arrangement of functional elements of a product into physical blocks. The purpose of product architecture is to define the basic physical building blocks of the product in terms of both what they do and what their interfaces are with the rest of the devices. At one extreme, modular product architecture allows each functional change independently by changing one component. At the other extreme, a fully integrated product requires change in all components to bring a desired change on functional element.[26] Design teams reunite the modules to set the plans for the next iteration of the product. It is critical to designate a creative manager to orchestrate this part of the process, and ensure that all contingencies are being discussed. The most innovative companies such as Apple and Google, assign this role to their most talented product managers and system experts.[27]

Literature provides many different sets of guiding principles for Lean product development. This study combines principles from different sources[28] [29] [30] [31] [32] [33] to come out with the following set of 9 guiding principles of Lean Product Development (LPD):

- **Eliminate waste:** waste is anything that does not add value to the product, value perceived by the customer, excluding necessary but non-value adding activities.
- **Modularity:** this incorporates platform approach as well as other kind of modularity in design; designing the product to be flexible to changes. This flexibility may be useful during the PD project or it may become useful for making later changes to the product when it has already been introduced to markets and the need for changes arise in the form of quality or cost issues. This kind of flexibility in design might raise the initial cost of the product, but in longer run it might prove to be a profitable way of designing products.
- **Workload leveling:** different product development projects with timely overlap compete for the same financial, technical, and human resources. When trying to maximize the overall product

[26] Naveem Gautam, Dissertation: *A Design reuse based framework on Lean Product Development*, doctoral thesis (philosophy), Wayne State University, Detroit, Michigan

[27] Barry Jaruzelski, Richard Holman, and Omar Daud, "Next-Generation Product Development", *Strategy+business,* May 30, 2011 © Booz & Company Inc. (a global management and strategy firm)

[28] Hoppmann J., Rebentisch E., Dombrowski U., Zahn T., "A Framework for organizing Lean Product Development", Engineering *Management journal,* Vol. 23, No. 1 March 2011

[29] "Thinking Tools for Agile Software Development Leaders" © Poppendieck. LLC Last updated February 2, 2003

development performance of an enterprise it is of major importance that their resources are planned on a cross-project basis, a methodology referred to as multi-project management[30].

- **Concurrent engineering (CE):** Concurrent Engineering offers potential to significantly reduce lead-time of the product development. CE is elaborated later in this study.
- **Cross-project knowledge transfer/Amplify learning:** it has been shown that even highly innovative products strongly depend and build upon knowledge of older products. This knowledge, if not appropriately captured, has to be continuously regenerated. [31] [32] This incorporates the aforementioned knowledge modules.
- **Process standardization:** while product development projects naturally differ from case to case, it has been found that many tasks required for planning and executing product development are quite consistent across different projects[31] [3331]. To increase product development performance, it is widely recommended to identify these reoccurring tasks and standardize them. Standardization:
    - reduces variability
    - increases efficiency
    - minimizes errors
    - helps to capture and manage knowledge
    - serves as a basis for continuous improvement [31] [34] [35]
- **Deliver as fast as possible:** Rapid development has many advantages. Speed assures the customer gets what they need now, not what they wanted yesterday.
- **Empower the team:** Top-notch execution lies in getting the details right, and no one understands the details better than the people who actually do the work. When equipped with necessary expertise and guided by a leader, they will make better technical decisions and better process decisions than anyone can make for them.
- **Build integrity in:** A system is perceived to have integrity when a user thinks – "Yes" That is exactly what I want". Market share is a rough measure of perceived integrity for products, because it measures customer perception over time[36]. Conceptual integrity means that the system's central concepts work together as a smooth, cohesive whole, and it is a critical factor in creating perceived integrity[37].

Activities in product development can be categorized as either value creating, necessary but non-value-creating (e.g.. performing a necessary hand-over), or waste. Waste in LPD includes the following:

- **overproducing information**
    - two different groups creating the same deliverable
- **over-processing information**

[30] Cusumano, Michael A., and Kentaro Nobeoka, *Thinking Beyond Lean: How Multi-Project Management is Transforming Product Development at Toyota and Other Companies,* Free Press (1998)
[31] Morgan, James M,. and Jeffery K. Liker, *The Toyota Product Development System: Integrating People, Process, and Technology,* Productivity Press (2006).
[32] Thomke, Stefan, and Takahiro Fujimoto, "The Effect of 'Front-loading' Problem-Solving on Product Development Performance", *Journal of Product Innovation Management,* 17:2 (2000), pp. 128-142
[33] Fiore, Clifford, *Accelerated Product Development: Combining Lean and Six Sigma for Peak Performance,* Productivity Press (2004).
[34] Ballé, Freddy, and Michael Ballé, "Lean Development", *Business Strategy Review,* 16:3 (2005), pp. 17-22.
[35] Sobek, Durward K., Allen C. Ward, and Jeffrey K. Liker, "Toyota's Principles of Set-Based Concurrent Engineering", *Sloan Management Review,* 40:2 (1999), pp. 67-83.
[36] Clark, "The Power of Product Integrity" (1994) pp. 278
[37] Brooks, "Mythical Man Month", (1995), pp. 255

- over-engineering components and systems
- working on different IT systems, converting data back and forth
- re-inventing processes/solutions/components/products when simple modifications of the already existing would suffice

➢ **miscommunication of information**
- unnecessary hand-offs
- multitasking
- structural barriers of communication
- knowledge barriers

➢ **generating defective information**
➢ **correcting information**
➢ **waiting of people**
➢ **unnecessary movement of people**[38]

As can be seen from figure 2.14 in product development projects core project activities are often idle, as engineers are waiting for necessary input, according to research from Lean Advancement Initiative (LAI) at MIT.

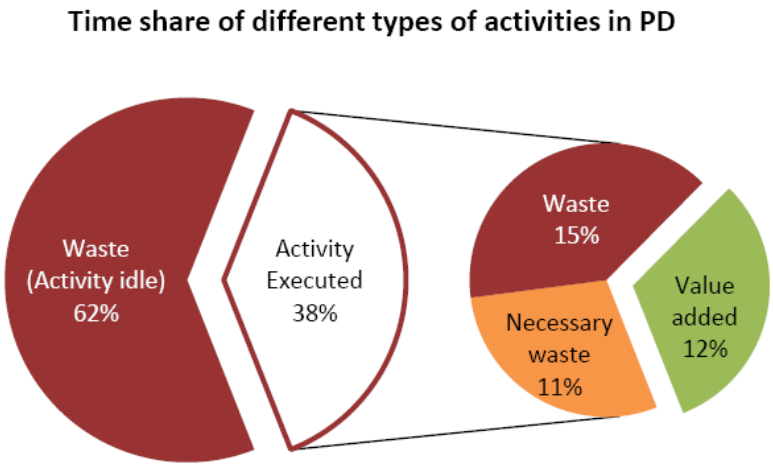


Figure 2.14 Time share of different types of activities in product development[38]

The central value-creating engineering tasks are idle for 62% of the time, on average. Only 12% of the time during the execution of a project are spend on value adding activities, 11% on necessary, but ultimately non-value adding activities, and 77% of the time is wasted (the project either being idle, or the engineers working on waste). [38] [39] [40]

Value stream mapping (VSM) is a Lean technique which aims at identifying waste in a given process. This technique places value adding and non-value adding activities on a process flowchart. After this, measures are taken to eliminate identified sources of waste.

[38] Dr. Josef Oehmen, Dr. Eric Rebentisch, "Waste in Lean Product Development", *LAI Paper Series "Lean Product Development for Practitioners",* July 2010 © Massachusetts Institute of Technology, 2010
[39] McManus, H. (2005) Product Development Value Stream Mapping (PDVSM) Manual, Cambridge, MA, Lean Advancement Initiative (LAI) at MIT
[40] Oppenheim, B. W., (2004) Lean Product Development Flow. *Systems Engineering,* 7, 362-376

abstract of an article published Barry Jaruzelski, Richard Holman, and Omar Daud, "Next-Generation Product Development", Strategy+business, May 30, 2011 © Booz & Company Inc. (a global management and strategy firm)

Combining agile with lean in product development[27]

The traditional gated product design process is rigid and linear, locking in customer preferences, potential risks, and other features at the beginning of the process. The gated model is a carefully choreographed approach that assumes almost perfect information and analysis at the beginning of the process. All too often, however, by the time the product is introduced, customer needs have evolved (or it becomes clear that they weren't fully understood in the first place). Further, when design and technology decisions are made early, so much complexity and risk may be introduced that turning back and reworking aspects of development triggers substantial cost overruns and delays in the final stages. Orderly but frequently ineffective, the gated approach has lost some of its luster in recent years.

Many companies have replaced it with lean product development techniques which focus on eliminating waste and improving speed-to-market. Companies applying lean techniques add continuous touch points with customers so they can test product concepts, prototypes, and features along the development and launch cycle. In so doing, they have reduced cycle time as much as 30 percent compared to the gated approach, as well as lowered development costs by as much as 40 percent and achieved dramatic gains in first-time quality.

But lean techniques fall short at the front end of the process. The enhanced efficiency of lean product development is (like the gated model) still highly dependent on early stabilization of requirements, rather than iterating, optimizing, and trading off requirements to get to winning product design. As a result, whatever innovation there is in this approach tends to be based on safeguarding the status quo rather than being creative – leaving companies exposed to disruptive changes in the market later on.

A next generation approach has been presented: one that applies agile product development techniques at the front end and lean approaches at the back end. An agile front end of product development system that is capable of addressing frequent iterations of multiple design options early in the process, based on continuous testing and highly sophisticated customer-driven design changes. This method, which both encourages flexibility and recognizes the unpredictability of the early stages of product development ensures that the latter part of the cycle is much less uncertain, enabling companies to bring more popular products to market at lower cost, and with fewer delays. Software companies have been the earliest adopters of this process, because they must routinely iterate numerous versions of their programs, and must assess them against customer needs and preferences well before the software is ready for mass release.

The goal of agile product development is to achieve rapid and frequent iterations with multiple design options up front – driven by continuous testing and granular customer analyses – in order to optimize, balance, and prioritize requirements and identify risks earlier. This early stage of the process has four characteristics:

- Rapid, iterative development model employing concurrent engineering
- modular architecture
- early risk identification
- intensive stakeholder and supplier involvement

2.7 Success factors in NPD

In this chapter the findings of different studies about NPD speed and new product performance are presented. In overall these two attributes are positively correlated, but there are individual factors that affect them adversely; for example, Marketing-Manufacturing Integration (MMI) in NPD have been shown to slightly increase time-to-market, but on the other hand having strong positive effect on product performance[41]. Two terms relate to these presented attributes; efficiently means *doing the things right*, whereas effectively means *doing the right things*.

The studies presented are cross-industrial. Factors affecting both speed and product performance appear to be contextually based, and thus cannot be perfectly generalized. Another thing to keep in mind is that these studies have been made using regression analysis. Regression analysis is used when the set of explaining factors is made up of multiple independent variables. The results are highly dependent on the way which the analysis is made and so different analysis approaches can reveal different results from the same sample.

2.7.1 Factors affecting NPD speed

Eisenhardt and Tabrizi (1995)[42] developed two alternative models of NPD speed; compression and experiential. Whereas the compression model relies on rational and overlapping development processes and is more suitable in stable environment, the experiential model relies on iteration, testing, and flexibility and thus is more suitable in uncertain environments. Eisenhardt and Tabrizi (1995) and Terwiesch and Loch (1999)[43] found that process overlap or concurrency, the key element of the compression model is significantly related to development speed under low rather than high environmental uncertainty. Eisenhardt and Tabrizi (1995) also found that iteration has a significant association with development speed under conditions of high rather than low uncertainty.

Recent research from Chen et al. (2010)[44], based on a large sample of empirical studies reporting associations between NPD speed and its antecedents, concluded that four process factors significantly increase NPD speed; process formalization, process concurrency, team experience, and iteration. These factors were significant in a broad range of NPD projects. Meta-analysis combining results from different studies by Gerwin and Barrowman (2002)[45] stated that overlapping and/or interaction and using formal methods appears to be very efficacious character in relation to NPD speed in a broad range of situations. Another study from Menon et. al (2001)[46] using published research findings and business experiences to determine factors related to NPD speed concluded that amongst others team-based incentive systems,

[41] Morgan Swink, Michael Song, "Effects of marketing-manufacturing integration on new product development time and competitive advantage", *Journal of Operations Management,* 25 (2007) 203-217

[42] Eisenhardt, K.M., Tabrizi, B.N., 1995. Accelerating adaptive processes: product innovation in the global computer industry. Administrative Science Quarterly 40, 84-110

[43] Terwiesch, C., Loch, C.H., 1999. Measuring the effectiveness of overlapping development activities. Management Science 45 (4), 455-465

[44] Jiyao Chen, Fariborz Damanpour, Richard R. Reilly, "Understanding antecedents of new product development speed: A meta-analysis, *Journal of Operations Management,* 28 (2010), pp. 17-33 doi:10.1016/j.jom.2009.07.001

[45] Donald Gerwin, Nicholas G., Barrowman, "An evaluation of Research on Integrated Product Development", *Management Science*, Vol. 48, No. 7, July 2002 pp. 938-953

[46] Ajay Menon, Jhinuk Chowdbury, Bryan A. Lukas, "Antecedents and outcomes of new product development speed; An interdisciplinary conceptual framework", *Industrial Marketing Management,* 31 (2002), pp. 317-328

empowerment, risk-tolerant culture, non-bureaucratic organizations, and flat organizational structure layers correlate with increasing NPD speed.

A study from 1999 of 75 NPD projects from ten large firms in USA in several industries reported that (a) clear time-goals, longer tenure among team members and parallel development increased NPD speed, whereas (b) design for manufacturability, frequent product testing, and CAD systems decreased NPD speed. However, when incremental and radical projects were sorted, different factors were found to influence time-to-market. For radical projects it is in most cases faster to co-locate teams, test products frequently and clearly specify the product concept and for incremental projects it is faster to spread out teams, limit testing and allow some vagueness in the product concept.[47]

This latter study reported decreased NPD speed as a result of more frequent testing and over-relying on CAD systems, both of which were not expected in the hypothesis part of this study. The conductors of this study give one potential explanation to why more frequent testing may result in reducing speed; speed is replaced with quality. Members of a project team may become overly concerned with the quality of their product and less so with speed. They go on by stating that using Simon's[48] concept of "satisfactory", whereby the product is brought to market when it is seen as "good enough" rather than when it is seen as "perfect", might able to reduce the number of design-build-test iterations and innovate faster when design improvements are carried out in the marketplace as successive generations rather than in the laboratory as successive iterations[49]. The way Apple develops products is a prime example of this, as was pointed out earlier in this study (pp. 31). Using CAD systems may result in decreased speed when libraries of accumulated knowledge/data don't exist or are not employed in a productive manner in the design and/or simulation of a product, but in occasions where this accumulated knowledge has been gathered and put to use properly, using CAD and simulations are expected to reduce prototyping and lead to increased NPD speed. Simulation solutions are being increasingly integrated with CAD, CAM, CAE etc. solutions and processes.

The aforementioned studies specify factors that affect NPD speed and some of the studies report slightly contrary results. As mentioned earlier this can be the result of different regression analysis approaches but these findings also support contingency theory. Contingency theory argues that there is not one "best answer" to a particular problem; instead, some of the factors affecting NPD speed are context specific and cannot be generalized.

Table 2.1 summarizes ***process factors*** that have been observed in increasing speed in NPD.

Table 2.1 Process factors increasing NPD speed (Kantomaa, T.T.)

PROCESS FACTOR	CLARIFICATION
formalized process	any type of formalization (not necessarily stage-gate)
team experience	longer tenure among team members
concurrent engineering	increases NPD speed, especially when technical uncertainty is low
intentional iteration	increases NPD speed, especially when technical uncertainty is high

[47] Eric H. Kessler, and Alok K. Chakrabarti, "Speeding up the Pace of New Product Development", *J Prod Innov Manag,* 1999;16:231-247

[48] Simon, H.A. Administrative Behaviour. New York: Free Press. 1976

[49] Wheelwright, S.C., and Clark, K.B. Accelerating the design-build-test cycle for effective product development. In: Strategic Management of Technology and Innovation. R.A. Burgelman, M.A. Maidique and S.C. Wheelwright (eds.). Chicago: Irwin, 1996, pp. 859-868

2.7.2 Factors affecting new product performance

A study[18] exploring the NPD project review practices of 425 Product Development & Management Association (PDMA) members revealed that:

- Only review proficiency is significantly associated with product performance (not the number of review points, the number of review criteria, the number of review team size etc.)
  - Furthermore, only the coefficient for proficiently using marketing criteria was significantly related to new product performance (proficiently using technical or financial criteria has no significant association with performance)
  - Finally, only the coefficient for proficiency at the first review point (i.e., the initial screen) is significant
- More review points are used for radical NPD projects than incremental ones, and this is related to a relatively lower rate of survival for radical projects

Key characteristics of effective processes are found to include an emphasis on upfront homework, tough go/kill decision points, sharp early product definition and flexibility[15].

A research using meta-analysis of relating literature conducted by Pattikawa et. al (2006)[50] reported the following variables having sizable relationship with new product project performance. In order of importance these factors are:

- the degree of organizational interaction – coordination and cooperation within the firm and between firms (cross-functionality)
- R&D and marketing interface
- general product development proficiency
- product advantage
- financial/business analysis
- technical proficiency
- management skill
- marketing proficiency
- market orientation - generation of market intelligence and dissemination of that intelligence across departments
- technology synergy - fit between the needs of the project and the firm's R&D or product development, engineering, and production resources and skills
- project manager competency
- launch activities

Another meta-analysis of new product performance literature made by Henard and Szymanski (2001)[51] stated that there are five factors that have the most significant impact on new product performance. These are:

- product advantage
- market potential

[50] Pattikawa, L.H., Verwaal, E., Commandeur, H.R., 2006. Understanding new product project performance. European Journal of Marketing 40 (11/12), 1178

[51] Henard, D.H., Szymanski, D.M., 2001. Why some new products are more successful than others, *Journal of Marketing Research,* 38 (3), 362-373

- meeting customer needs
- predevelopment task proficiencies
- dedicated resources

According to Cooper there are eight critical success factors for new product development projects:

1. **Up-front homework pays off** – Too many projects move straight from the idea stage into development with little or no up-front homework. More time and resources must be devoted to the activities that precede the design and development of the product. Up-front homework means undertaking thorough market and competitive analyses, research on the customer's needs and wants, concept testing, and technical and operations feasibility assessments.
2. **Build in the voice of the customer** – New product projects that feature high-quality marketing actions are blessed with more than double the success rates and 70-percent higher market shares than those projects with poor marketing actions[52]. In many cases the lack of marketing resources is the reason for inadequate marketing activities in NPD projects. For example, one study conducted in the U.S found that market research was not done at all in about 75 percent of NPD projects[53].
3. **Seek differentiated, superior products** – Superior products, with differentiation and unique customer benefits and value to the user, have five times the success rate, over four times the market share, and four times the profitability as products lacking in this ingredient[54]. In many instances the preoccupation with cycle time reduction and the tendency to favor simple, inexpensive projects penalize projects that lead to product superiority[55]. The lack of "game-changers" can also be devoted to shortage of resources and thinking profitability in the short-term.
4. **Demand sharp, stable and early product definition** – A failure to define the product – its target market; the concept, benefits and positioning; and its requirements, features and specs – before development begins is a major cause of both new product failure and serious delays in time-to-market[54] [56] [57].
5. **Plan and resource the launch… early in the game!** – A strong market launch underlies successful products. The need for a quality launch – well planned, properly resourced, and well executed – should be obvious. For example, new product winners devote more than twice as many person-days and dollars to the launch as do failure teams.
6. **Build tough go/kill decision points into your process – a funnel, not a tunnel** – In many companies, projects move far into development without serious scrutiny; once a project begins there is very little chance that it will ever be killed. The result is many marginal projects approved, and a misallocation of scarce resources.

[52] Cooper, R.G., Edgett, S.J. and Kleinschmidt, E.J, *Portfolio Management for new products.* Reading, Mass; Addison-Wesley, 1998

[53] Cooper, R.G., *Winning at New Products: Accelerating the Process from Idea to Launch, 3rd edition.* Reading, Mass: Perseus Books, 2001

[54] Cooper, R.G. & Kleinschmidt, E.J, *New Products: The key Factors in Success.* Chicago: American Marketing Assoc., 1990, monograph

[55] Crawford, C.M., "The hidden costs of accelerated product development", *Journal of Product Innovation Management,* vol. 9, no. 3, Sept. 1992, pp. 188-199

[56] Cooper, R.G. & Kleinschmidt, E.J, "Major New Products: What distinguishes the winners in the chemical industry", *Journal of Product Innovation Management* 2, 10, March 1993, 90-111

[57] Montoya-Weiss, M.M. & Calantone, R.J., "Determinants of new product performance: a review and meta-analysis", *Journal of Product Innovation Management* 11, 5, Nov. 1994, 397-417

7. **Organize around true cross-functional project teams** – "Rip apart a badly developed project and you will unfailingly find 75 percent of slippage attributable to (1) 'siloing' or sending memos up and down vertical organizational 'silos' or 'stovepipes' for decisions, and (2) sequential problem solving", according to Peters[58]. Numerous studies concur; good organizational design is strongly linked with success[57]. Good organizational design means projects that are organized as a cross-functional team, led by a strong project leader, accountable for the entire project from beginning to end, dedicated and focused (as opposed to spread over many projects), and where top management is committed to the project.
8. **Build an international orientation into your new product process** – New products aimed at international markets (as opposed to domestic) and with international requirements built in from the outset fare better[54]. The result is either a *global* product (one version for the entire world) or *glocal* product (one product concept, one development effort, but perhaps several variants to satisfy different international markets). An international orientation also means adopting a transnational new product process, utilizing cross-functional teams with members from different countries, and gathering market information from multiple international markets as input to the new product's design. [21] [59]

The turn of the century witnessed the dismantling of many corporate central research laboratories. In many businesses fundamental research was no longer fashionable. The results were predictable; the research shifted to a much shorter term focus, and industry leaders complained that there were nothing great coming down the pipe.[19]

One study[60] reported that the perspective on managing R&D has changed over the years, moving from technology-centered model to a more interaction-focused view. Figure 2.15 depicts the five generations of R&D according to this study.

[58] Peters, T.J., *Thriving on Chaos.* New York; Harper & Row, 1988.
[59] Dr Robert G Cooper, Dr. Scott J. Edgett, "Overcoming the Current Crunch in NPD resources", Working paper No. 17 ©2006 Product Development Institute
[60] D. Nobelius, "Towards the sixth generation of R&D management", International Journal of *Project Management,* 22 (2004), pp. 369-375

R&D Generations	Context	Process Characteristics
First generation	Black hole demand (1950 to mid- 1960s)	R&D as ivory tower, technology-push oriented, seen as an overhead cost, having little or no interaction with the rest of the company or overall strategy. Focus on scientific breakthroughs.
Second generation	Market shares battle (mid-1960s to early 1970s)	R&D as business, market-pull oriented, and strategy-driven from the business side, all under the umbrella of project management and the internal customer concept.
Third generation	Rationalization efforts (mid-1970s to mid-1980s)	R&D as portfolio, moving away from individual projects view, and with linkages to both business and corporate strategies. Risk-reward and similar methods guide the overall investments.
Fourth generation	Time-based struggle (early 1980s to mid-1990s)	R&D as integrative activity, learning from and with customers, moving away from a product focus to a total concept focus, where activities are conducted in parallel by cross-functional teams.
Fifth generation	Systems integration (mid-1990s onward)	R&D as network, focusing on collaboration within a wider system – involving competitors, suppliers, distributors, etc. The ability to control product development speed is imperative, separating R from D.

Figure 2.15 Description of five generations of R&D processes[60]

Worth noticing is that these five models of R&D generations, though presented on a time scale, hold components or ideas still valid or sought for by many companies, and hence do not represent a map of where companies today are to be placed.

THE SIXTH GENERATION OF R&D

The sixth generation of R&D is re-focusing on the research part, having ever increasing product and technology complexity connecting to loosely-tied multi-technology research networks. The bases for this new set of approaches are a broader multi-technology base for high-tech products and more distributed technology-sourcing structure. The Bluetooth case is a good example of the roots and ideas of the 6th generation of R&D. Originating from Ericsson, it represents a joint cross-industrial, open intellectual property-based effort in developing and bringing a new technology to the market by utilizing the resources from more than one thousand companies. This shift towards the 6th generations of R&D is predicted to return to the roots, i.e. back to the purpose of the first generations corporate research labs, one pursuing more radical innovations. One could see this as a re-focus on the research part of the research and development. The corporate research labs as such are not predicted to resurface, instead the re-focus is taking on other approaches. Much of the breakthrough research will not be a result of one company's lab efforts, instead breakthroughs will be based on joint efforts from loosely tied networks of smaller players driven more of pure interest than profits.[60]

3 MODELING THE PRODUCT DEVELOPMENT PROCESS

Over the past years a number of various ways to improve business performance have been highlighted: time-based competition, matrix structure, lean principles, concurrent engineering and Business Process Reengineering (BPR) to name a few recent "management innovations". Central to many performance improvement efforts is an emphasis on processes[61]. This reflects well the contemporary trend of encouraging companies to think in processes instead of functions and procedures[62]. Improving business processes was "number one priority" among the top ten business priorities in 2009 in a Gartner survey covering more than 1,526 CIO's[63].

The activity of improving business processes relates to many terms; business process improvement (BPI), business process redesign, business process reengineering (BPR), core process redesign, business restructuring, continuous improvement processes, Six Sigma, to name the prominent ones.

A contrast to the incremental ideas of Business Process Improvement (BPI) is that of the radical break-through approaches of business process re-engineering (BPR). The technique is a blend of a number of ideas found within operations (process flow-charting, network management), and the need for customer focus. These were brought together to define BPR as:

The fundamental rethinking and radical redesign of business processes to achieve dramatic improvements in critical, contemporary measures of performance, such as cost, quality, service and speed[64]

The IDEF9 Business Constraint Discovery method was designed to assist in the discovery and analysis of constraints in a business system. A primary motivation driving the development of IDEF9 was the realization that the collection of constraints that forge an enterprise system is generally poorly defined. That, is to say, the knowledge of what constraints exist and how those constrains interact is at best incomplete, disjoint, distributed, and often times completely unknown. If the desire exists to modify the business to improve its performance, then knowledge of these constraints is critical. Knowledge about business constraints can help efforts like BPR, Total Quality Management (TQM), and so forth. [65]

These above mentioned business improvement approaches have been criticized for concentrating in many instances on what needs to be done before and after the improvement act, but the act of improving itself seems to be a black box[66]. Sharp and McDermott compare the mysterious step from the as-is to the to-be with the famous, And Then, A Miracle Occurs procedure[67]. These approaches don't precisely state how the

[61] Soumitra Dutta, Jean-Francois Manzoni (1999), *Process Re-engineering, Organizational Change and Performance Improvement*, McGraw-Hill

[62] Asbjørn Rolstadås (1995). Business process modeling and reengineering. *Performance Management: A Business Process Benchmarking Approach*. p. 148-150

[63] Auringer, A. (2009), Meeting the challenge: The 2009 Higher Education CIO agenda, *Gartner*, Stamford, CT.

[64] Slack, N., Chambers, S., Harland, C., Harrison, A. and Johnson, R. (2004) *Operations Management*, 4th edition, Pitman, London

[65] R. Mayer, M. Painter and M. Lingineni, Information Integration for Concurrent Engineering (IICE): toward a method for business constraint discovery (IDEF9), Knowledge Base Systems, INC (1995)

[66] Gregor Zellner, A structured evaluation of business process improvement approaches, *Business Process Management Journal*, Vol. 17, No. 2, 2011, pp. 203-237

[67] Sharp, A. and McDermott, P. (2001), *Workflow modeling: Tools for process improvement and application development*, Artech House, Boston, MA.

changes needed are to be accomplished and the act of business process improvement seems to be rather art than science. [66]

A method is goal oriented and provides a systematic approach[68]. This means that a method specifies how to achieve defined goals. A technique is a procedure that performs a development activity[69]. For that reason a technique is an essential part of a method. Techniques can be supported by tools (for example software), which are the means for creating the results[70].[66]

Improving processes begins with mapping the present situation (the AS-IS), and by defining what is wanted from the change (the TO-BE). The main purpose behind business process mapping is to assist organizations in attaining full understanding of all the steps of the process, and through interviews to understand how the process is actually done, not just how it is defined to be done. After that an analysis can be made utilizing techniques and tools to make the process run more effectively.

3.1 Traditional techniques for process modeling and project planning

Graph based techniques, such as CPM and PERT, and flowcharts, and matrix based techniques, such as DSM, have been in wide use to capture the relationships between the tasks in product development process and in scheduling these tasks.[71] However, all of these techniques have deficiencies in some aspect. Figure 3.1 illustrates some of these techniques.

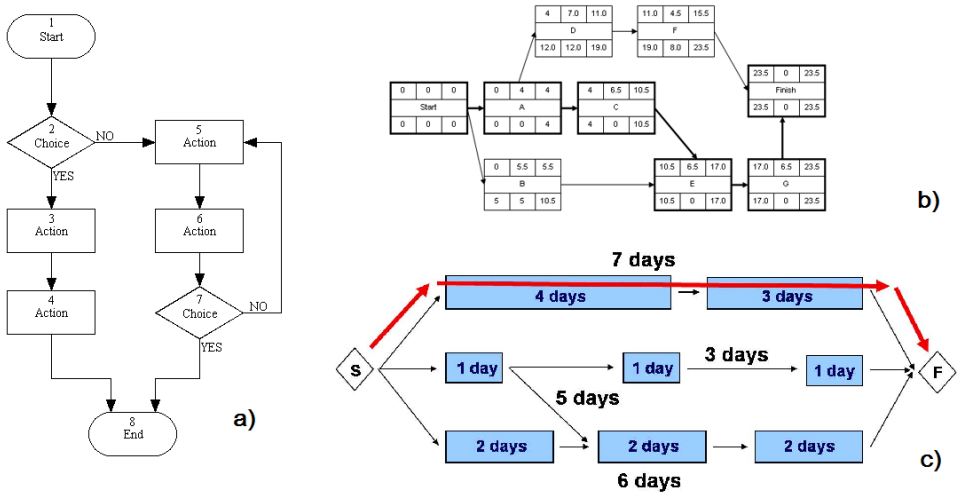


Figure 3.1 Traditional techniques for process modeling and project planning; a) flowchart b) PERT c) CPM (Kantomaa, T.T.)

[68] Braun, C., Wortmann, F., Hafner, M. and Winter, R. (2005) "Method construction – a core approach to organizational engineering", in Liebrock, L.M. (Ed.) *Proceedings of the 2005 ACM Symposium on Applied Computing, SAC '05, Santa Fe, NM,*ACM, New York, NY, pp. 1295-1299

[69] Brinkkemper, S. (1996), "Method engineering: engineering of information systems development methods and tools", *Journal of Information and Software Technology*, Vol. 38, No. 4, pp. 275-280

[70] Baumöl, U. (2005), "Strategic agility through situational method construction", in Reichwald, R. and Huff, A.S. (Eds), *European Academy of Management (EURAM) Annual Conference, Munich.*

[71] Shawn T.Collins, Ali A. Yassine, and Stephen P.Borgatti, Evaluating Product Development using Network Analysis, *Systems Engineering*, DOI 10.1002/sys.20108

Figure 3.1 (a) illustrates a flow chart. Data flows are not typically represented in a flowchart; rather, they are implied by the sequencing of tasks. Flow charts are good for visualizing a process, but they do not contain necessary information for complex analysis and planning purposes.

The Project Evaluation and review technique (PERT) (figure 3.1 (b)) and critical path method (CPM) (figure 3.1 (c)) have long been established in industry as tools for planning and managing projects. Both of these, however, have shortcomings in their rigid analysis and specification for the project structure. CPM predicts the time required to complete a project and identifies the critical path. For less routine projects there is more uncertainty in the completion times, and this uncertainty limits the usefulness of the deterministic CPM model. An alternative to CPM is the PERT project planning model, which uses probabilistic task durations. Many studies have been conducted on extending the classical concept of PERT to deal with more complex problems, but they still cannot represent complex interaction patterns (i.e. iterations) among activities involved in the product development projects.[72] In Pert analysis tasks that are coupled would be collapsed into one task but this is approach hides the real design problems and forbids any opportunity to further improve the design procedure by applying other techniques[73].

Another downfall of PERT and CPM are that they do not take into account the resources used to accomplish tasks, i.e. they do not consider resource constraints (although leveled CPM does). Critical Chain Project Management (CCPM) is a method based on Theory of Constraints (TOC) - thinking. It is a method and a tool (software) that can be applied in multi-project environment to schedule projects optimally according to constraints. CCPM scheduling includes safety buffers, but is still deterministic in the way that it does not take into consideration other dynamics in product development that lead to iterations, and may not cope well in an unpredictable and complex environment. One case study showed how CCPM can become very heavy to maintain in a complex product development environment leading to constant rescheduling[74]. However, the same study reported a benefit of increased visibility between interdependent projects in the multi-project environment in the case company. There has not been a comprehensive case research on the successfulness of implementing CCPM in complex product development environments.

Graphical Evaluation Review Technique (GERT) is an extension of PERT. It is one of the network-based tools proposed to deal with iterations under uncertain conditions. GERT is a procedure for stochastic network analysis in which activities have a probability of occurrence and their duration is treated as a random variable. GERT supports serial, parallel, and iterative structures. [75] [76] [77]

This technique has not gained compelling acceptance in the NPD management and community. [75] Given that GERT requires an up-front definition of the process structure for its analysis, it may not be convenient in exploring process structure alternatives[7].

---

[72] Elmaghraby, S.E.. (1995). Activity Nets: A guided tour through some recent developments. *European Journal of Operations Research*, 82(3), 383-408

[73] Eppinger, S.D., Whitney, D.E., Smith, R.P., and Gebala, D.A. (1994) *A Model-Based Method for Organizing Tasks in Product Development,* Research in Engineering Design, **6**: 1-13

[74] Farhad Dilmaghani (2008). Critical Chain Project Management (CCPM) at Bosch Security Systems (CCTV) Eindhoven, Master's Thesis, University of Twente, School of Management and Governance.

[75] Martinez, C., (2010) "A Coordination Mechanism for Lean Product Development", *Department of Industrial Engineering*, Texas Tech University, Lubbock, Texas

[76] Pritsker, A.A.B., and Happ, W.W., 1966, "GERT: Graphical Evaluation and Review Technique Part I Fundamentals", *The Journal of Industrial Engineering*, 17(5), 267-274

[77] Whitehouse, G.E., 1970, "GERT, A Useful Technique for Analyzing Reliability Problems", Technometrics, 12(1), 33-48

3.2 Design Structure Matrix

Product development projects include the completion of hundreds or even thousands of tasks. These tasks can be either sequential, parallel or coupled tasks. Figure 3.2 presents these three different schemes.

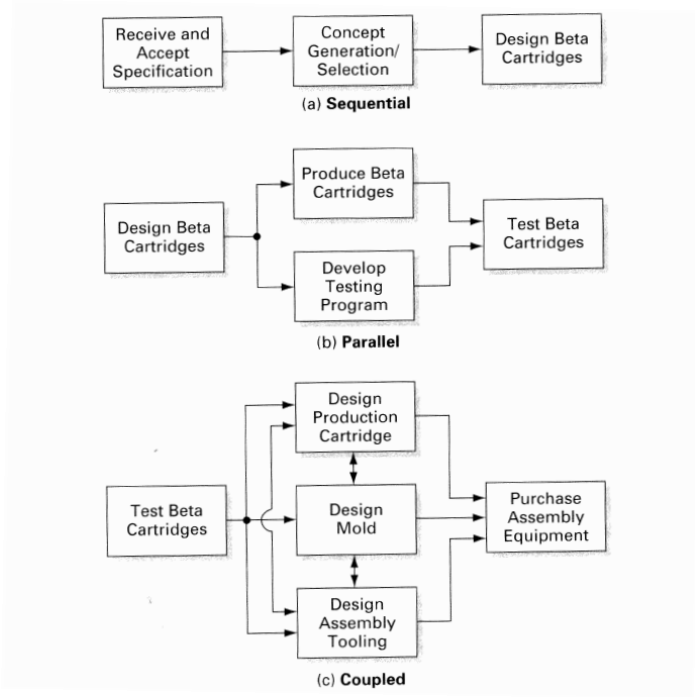


Figure 3.2 Sequential, parallel, and coupled tasks[10]

The representation in figure 3.2 is *information-processing view* or *data-driven perspective* of product development since most of the dependencies involve the transfer of information (data) between the tasks. Task B is dependent on task A if an output of task A is required to complete task B. This dependency is denoted by an arrow from task A to task B. In figure 3.2 (a) tasks are in sequential order because the dependencies impose it. Figure (b) shows four development tasks of which middle two tasks are dependent on the output from the one in the left but these two tasks are not dependent on each other and can be performed in parallel. The task on the right depends on the middle two tasks. Figure (c) shows five development tasks, three of which are coupled. Coupled tasks are mutually dependent; each task requires the result of the other tasks in order to be completed. Coupled tasks either must be executed simultaneously with continual exchanges of information or must be carried out in an iterative fashion. When coupled tasks are completed iteratively, the tasks are performed either sequentially or simultaneously with the understanding that the results are tentative and that each task will most likely be repeated one or more times until the team converges on a solution.[10]

A useful tool for representing and analyzing task dependencies is the Design Structure Matrix (DSM). In a DSM model, a project task is assigned to a row and a corresponding column. The rows and columns are named and ordered identically, although generally only the rows list the names of the tasks completely. Each task is defined by a row in the matrix. Reading across a row reveals the tasks whose output is required to perform the task corresponding to the row. Reading down a column reveals which tasks receive

information from the task corresponding to the column. The diagonal cells are usually filled with dots simply to separate the upper and lower triangles of the matrix and to facilitate tracing dependencies.

The collected data in a DSM captures the structure of interactions, interdependencies, interfaces between system elements and information flow between project tasks. There are two main categories of DSM's; static and time-based. Static DSM analysis uses clustering algorithms to identify groups of coupled tasks to place in subsystem of frequently interacting tasks. In time-based DSMs, the ordering of the rows and columns indicates a flow through time. A mark above the diagonal has special significance; it indicates that an earlier task is dependent on a later task. This can mean that two sequential tasks are ordered backwards and that the sequence of tasks should be changed, or that the tasks are coupled. Terms like "feedforward" and "feedback" become meaningful when referring to interfaces. Time-based DSM analysis uses *partitioning* algorithms that optimally sequence project tasks to minimize the impact of rework and unexpected feedback. [10 84]

Changing task orders is called *sequencing (static)* or *partitioning (time-based)* the DSM, and there are simple algorithms for this use. The structure of the DSM reveals which tasks are sequential, parallel, and coupled as figure 3.3 shows.

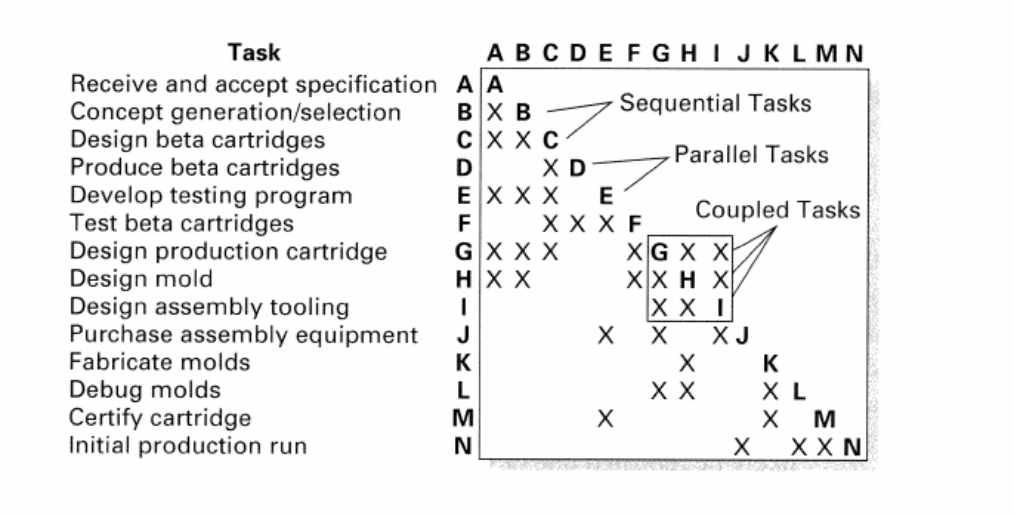


Figure 3.3 DSM comprised of 14 development tasks[10]

Coupled tasks lead to iterations. Project lead time can be diminished through:

- Performing iterations more quickly
- decoupling tasks to avoid iterations
- Considering sets of solutions

Much of the delay in completing coupled tasks is in passing information from one person to another and in waiting for a response. The way to perform the iterations more quickly is to form multi-functional teams where team members work closely together; the mechanical engineer, electrical engineer and testing personnel work closely together and exchange information rapidly. Another way is to decouple tasks where possible. For example, by clearly defining an interface between two interacting components early in the design process, the subsequent design of the two components can proceed independently and in parallel. A

third way to reduce project lead time is to consider sets of solutions instead of point-value estimates of design parameters. In some cases the use of sets or ranges may facilitate faster convergence of coupled tasks. Figure 3.4 illustrates a DSM where iterations have been indicated.

Figure 3.4 Design structure matrix illustrating the iterations needed to complete a particular development phase[71]

Smith end Eppinger extended DSM representation with repetition probabilities and task duration and developed a stochastic model to predict the expected duration of iterative development process. They also developed a Work Transformation Matrix (WTM) model based on DSM to predict slow and rapid convergence of parallel iterations. [6]

Criticism of DSM generally includes the fact that it doesn't support stochastic modeling (excluding the extended DSM representation from Smith and Eppinger) and assumes that all activities in the matrix must be undertaken[7]. Furthermore, overlapping activities cannot be explicitly visualized[78] [79].

Others criticize the fact that in using DSM the task interaction matrix assumes that feedback to earlier tasks is undesirable. Sometimes iterations are intentional. An alternative approach is to cluster the DSM into task subsets that are mutually exclusive or minimally interacting. Static DSM uses this approach for data that typically don't contain time-dependent information; by doing this clusters of intentionally iterative or concurrently scheduled tasks can be identified. A weakness of this approach for rigorous understanding of information flow is that DSM techniques do not include methods to evaluate the within and between

[78] 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*, 48(3), 292-306
[79] Browning, T.R., 2002. "Process Integration Using the Design Structure Matrix", *System Engineering*, 5, 180-193

cluster task relationships once clusters are identified. However, a number of diagnostics measures are available from Network Analysis (NA). [84]

3.3 Network Analysis

In product development each development activity is responsible for generating the required product or process knowledge based on the received input information from other activities. The information evolution of a development activity is considered to as a gradual refinement of the product or process knowledge from its preliminary form to a final form over the period of activity execution. Faster evolution represents an early resolution of technical uncertainty.[80]

It has been noted that information flows in organizations are essential indicators for the quality of development processes[81]. Also, according to Amabile (1998), managers often destroy creativity in project business by failing to make appropriate connections between actors and activities. It is important to identify the interdependencies between particular actors and specific activities, and to identify the resources they require to perform their tasks. However, it is often difficult to determine the "appropriate" connections.[82]

Using Network Analysis (NA) metrics to measure properties of information flow (such as degree centrality, influence, and brokerage) helps to identify important product development tasks and interactions that constrain product development process execution. NA aims at demonstrating several insights to manage the product development process both as a second (i.e., effectively executing) and third order (i.e., highlighting underlying premises and assumptions) form of organizational control[83]. Project managers can use these data to structure team integration mechanisms or to identify coordinating mechanisms for groups of concurrently scheduled tasks. From mapping and analyzing information flows an evaluation of tasks like stage-gates and design reviews can be made; are they acting as effective information flow regulators in the product development process. NA integrates quantitative, qualitative, and graphical data, thus allowing more thorough and in-depth analysis. [84]

Network Analysis

Using Network Analysis (NA) is an important analytical shift from viewing a task as having individually-determined characteristics (as is the case with the traditional tools) to viewing it as representing emergent properties – emerging from patterns of information flows between tasks[84]. Network analysis has its roots on social sciences.

[80] Juite Wang, Yung-I Lin, An overlapping process model to assess schedule risk for new product development *Computers & Industrial Engineering 57 (2009) 460-474*

[81] Sander PC, Brombacher AC. "Analysis of quality information flows in the product creation process of high volume consumer products", *J Prod Econom,* 2000;67;37-52

[82] Amabile, T. (1998). How to kill creativity, *Harvard Business Review*, Vol. 76, No. 5,pp. 76-87.

[83] Collins, Shawn T., Bradley, Joe A., Yassine Ali A. Analyzing Product Development Task Networks to Examine Organizational Change, *IEEE Transactions on Engineering Management* Aug2010, Vol. 57 Issue 3, p513-525

[84] Collins, Shawn T., Yassine, Ali A., Borgatti, Stephen P. (2008). Evaluating Product Development Systems using Network Analysis, *Systems Engineering*, DOI 10.1002/sys

Network analysis has been used in various situations, mostly in social sciences. One research made in the University of Vaasa used NA in the client company (ABB) to analyze agile supply-demand networks (ASDN's)[85]. This research concluded that a network-based framework provided a solid basis for a rich description and analysis of a multi-actor project business, as well as assisting in understanding important interpersonal relationship factors that affect project success; such as trust, commitment and adaption. The University of Vaasa developed agile supply-demand network (ASDN) analysis software for ABB. This software could be useful for analyzing product development processes.

The basic element of NA is an adjacency matrix that defines relationships between each row and column element. For a matrix of product development tasks collected in DSM format, directed data indicates the input/output flow of *information* between each task. This matrix is derived, for example from self-reporting questionnaires distributed to actors in the network asking about relationships with other actors. Personal interviews can be conducted after this to validate the results from the questionnaires and to get additional data difficult to access through impersonal surveys.

Once the data is collected in a matrix form, NA methods contain a variety of analytic techniques. Network position analysis looks at attributes emerging of ties between nodes. Network categories are thus inherently relational roles rather than individual categories. Table 3.1 summarizes a set of key NA metrics for evaluating interacting product development tasks.

Table 3.1 Network analysis metrics for evaluating product development process task interactions[86]

NA Term	Definition	Product Development Application
Analysis Level: Overall Structure		
InDegree Centrality	Number of incoming ties for each node	Critical PDP task based on high information *collection* load
OutDegree Centrality	Number of outgoing ties for each node	Critical PDP task based on high information *dissemination* load
Analysis Level: Sub-Structure		
Clusters	Divide the network into N groups with maximized internal ties and minized external ties	Identifies the N groups of iterative tasks that can be conducted concurrently
Density	Number of actual ties between nodes or groups divided by number of possible ties	Measures iteration within a group of concurrently scheduled tasks and between groups of separate tasks
Analysis Level: Individual Tasks		
Influence	Degree of network dependence on a node (teacher of the next president)	Low visibility PDP tasks that have potentially important information
Betweenness Centrality	Number of times a node is on the shortest path between two other nodes	PDP "transistor" task that regulates information flow between two other tasks
Brokerage	What is the nature of B's role as a transmitter of information between A and C (co-ordinator, gatekeeper, representative, liason, consultant)?	Characterizes role of key tasks that transfer information within and between PDP task groups

InDegree Centrality measures the number of inputs a task has. This defines the information collection load of a task. OutDegree Centrality measures the number of outputs that each task has. This defines the information dissemination load a task has.

[85] Maqsood Sandhu and Petri Helo, A Network approach to project business analysis, *Logistics systems research group, university of Vaasa, Vaasa, Finland*

[86] Wasserman, S., Faust, K. (1999), Social network analysis methods and applications, *Cambridge University Press*, Cambridge

Influence measures the degree of connectedness between each task and every other task in the product development process. Tasks with high influence have information that is important to the entire product development process, even if they do not appear on the critical path. Tasks with low influence have information that is not important to the entire product development process.

Betweenness centrality measures the number of times a particular task is on the shortest path *(geodesic distance)* between two other tasks. Like standard critical path identification, tasks with high betweenness centrality have a high schedule impact on the product development process if they are not completed in time. It should be noted here that the DSM is constructed to present inputs which are critical to the *completion* of tasks, not on the *commencement* of tasks. So the shortest path *(geodesic distance)* between tasks can provide the quickest information route between two tasks and might define when the information receiving task can be commenced (the earliest possible time). As a measure of shortest information flow, betweenness centrality also identifies which tasks have the greatest control of information in product development process. Measuring betweenness centrality provides a way to measure whether tasks that are expected to be critical for product development process execution actually control information flow in task input/ output relationships.

Figure 3.5 illustrates the last metrics, brokerage. In this example B is a broker, because it passes the information from A to C. Brokerage indicates the individual position of a task. Task B can be a coordinator (if three tasks are all in the same group), a gatekeeper (A is in different group than B and C) or a representative (if C is in different group than A and B).

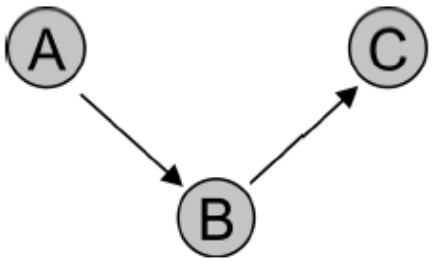


Figure 3.5 Brokerage[84]

Brokerage identifies tasks that have critical information sharing roles within and between task clusters. Expected brokerage calculates the number of times each task in a cluster would be expected to play each brokerage role based in the number of clusters and size of each cluster. Relative brokerage indicates the actual number of roles each task plays against the expected number of roles. High relative brokerage indicates a task is playing a particular role more than expected, and low relative brokerage indicates the task is playing a role less than expected.

3.3.1 Example of Network Analysis in practice [83] [84]

A company that used a common stage-gate model, having 5 gates (from 0 to 4), investigated matrix based project management tools during a quality improvement initiative to streamline its product development. This resulted in their product development process being captured and evaluated using traditional time based DSM, as shown in Figure 3.6. The lower left half of the matrix denotes feed-forward relationships, while the upper right half denotes feedback relationships. 1 denotes weak task dependency, while 2 denotes strong task dependency.

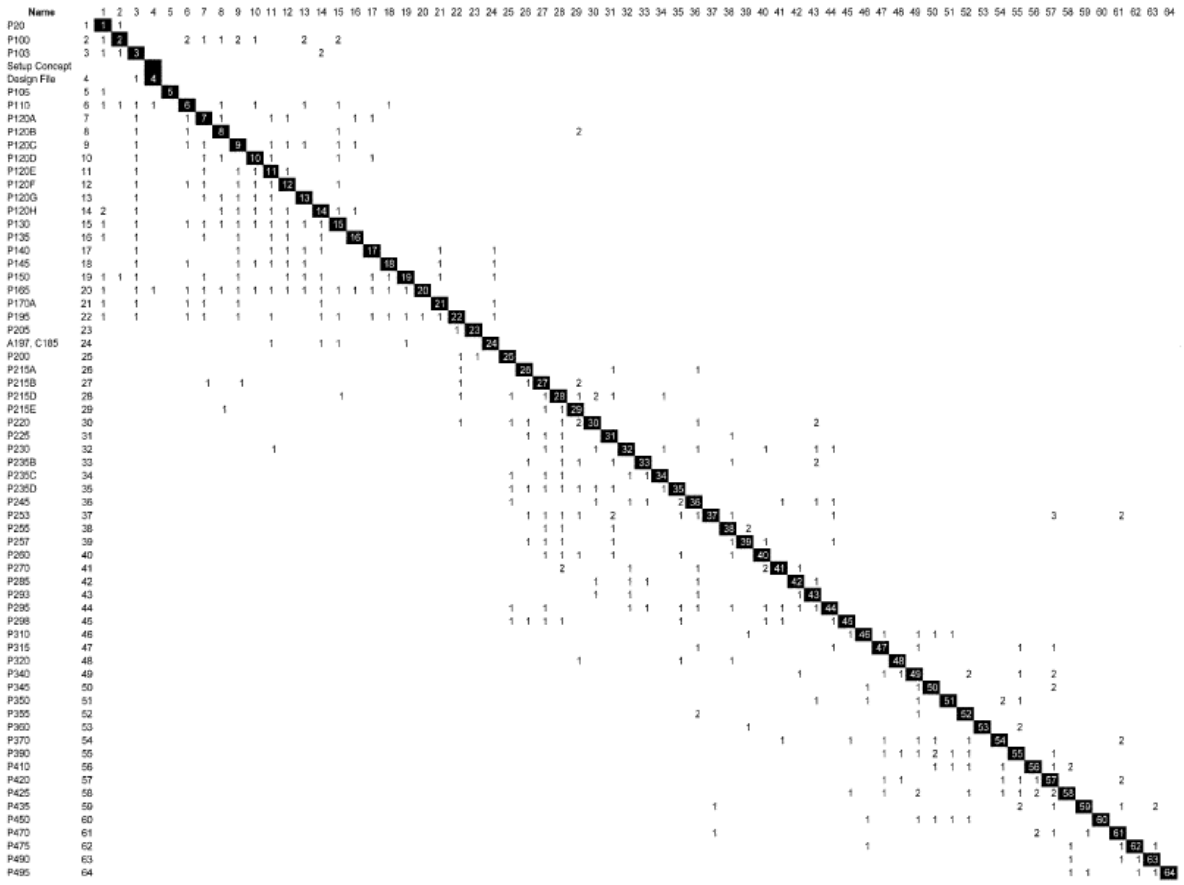


Figure 3.6 Full DSM of a product development process[84]

INDEGREE CENTRALITY AND OUTDEGREE CENTRALITY

An analysis of this DSM was made based on the metrics presented in table 3.1. Figure 3.7 shows control charts for InDegree Centrality and OutDegree Centrality on a number of tasks. The upper and lower control limits (UCL, LCL) are drawn based on the two-sigma point around the average value.

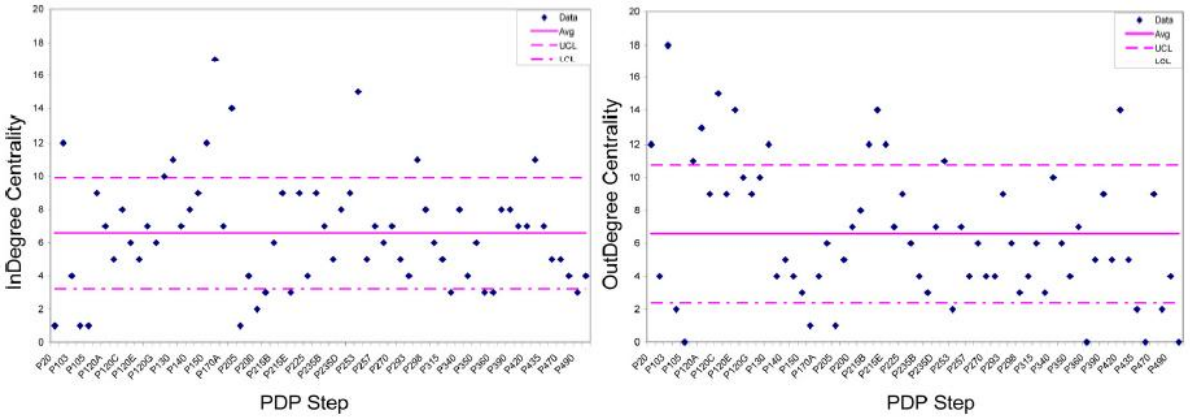


Figure 3.7 Control charts for InDegree Centrality and OutDegree Centrality[84]

From these control charts the company identified tasks that needed corrections. There is a task P105 "review lessons learned database", for example, which shows on the control chart as having low InDegree Centrality (few information inputs). There now existed a need to increase ties from P105 so that lessons

learned reviews are better integrated with the other design activities. There were also tasks which actually had too many inputs; this leads to large information collection burden.

The next level of analysis was identifying groups of tasks that interact more with each other than with other tasks. Clustering identifies groups of tasks that are strongly connected to each other as a basis for scheduling concurrent activity. Density measures the number of actual and potential ties within a cluster, and between clusters. Table 3.2 presents the results of using a special algorithm for these purposes.

Table 3.2 Clustering and measuring density[84]

	System Definition	Preliminary Functional Integration	Preliminary Mechanical Integration	Detailed Design & Verification	Validation
System Definition	**0.41**	0.03	0.02	0.00	0.06
Preliminary Functional Integration	0.01	**0.39**	0.18	0.05	0.03
Preliminary Mechanical Integration	0.00	0.11	**0.37**	0.05	0.00
Detailed Design & Verification	0.00	0.02	0.00	**0.45**	0.10
Validation	0.08	0.02	0.00	0.03	**0.16**

The upper half of the table shows feedforward ties. The lower half shows feedback ties.

The algorithm used above maximizes within cluster task-interaction and minimizes between-cluster task interaction. The diagonal values of table 3.2 measure the density of ties within the cluster. The off-diagonal values measures the density of ties between clusters. For example there are 41% of the potential connections within the System Definition phase and 3% of the potential connections between System Definition outputs to Preliminary Functional Integration inputs. Several tasks conducted early in the product development process execution are placed to the Validation cluster based on loose ties to the rest of the product development process. This placement shows that the relationship of these tasks to other System Definition tasks was not well defined. The low off-diagonal density values in table 3.2 show that the clustering intent, which was to identify groups of iterative or concurrent tasks that are independent of the rest of the tasks, was mostly successful. The primary exceptions are the ties between Preliminary Functional Integration and Preliminary Mechanical Integration phases which have 18% and 11% density values between them. Project planning could take this into consideration and define a single set of activities called Preliminary Integration, which would need coordination mechanisms to manage the interdependencies between these two sets of concurrent activities.

BETWEENNESS CENTRALITY

In the case of the example company gates 2 and 3 do not score high on betweenness centrality, meaning that their importance may have been overrated.

INFLUENCE

The most influential steps are P251E (detailed heat and mass balance) and P120B (design point definition). This makes intuitively sense, since steady state model and design points are concept level activities which contain information that constrain shapes of execution for the rest of the product development process.

BROKERAGE

Figure 3.8 shows the number of tasks in each cluster where the relative brokerage value is greater than 3. High coordination roles in all five task clusters identify tasks that manage design iteration in each phase. The coordination is especially high in the System Definition cluster, which is natural since this is the phase when the concept is taking shape. This attention to the required coordination within product development process phase is expected based on the case company's desire to define clusters of concurrently scheduled or iterative tasks. The coordination burden decreases from System Definition to Functional Integration, with an increase during Detailed Design and decrease during Validation. This shows a cyclical pattern of formal iteration as the design progresses through the product development process. The increasing Representative values from System Definition through Mechanical Integration shows a larger number of tasks involved in passing information to another phase. The increasing Gatekeeper values show a larger number of tasks involved in receiving information from another phase.

Brokerage roles are a standard set of NA metrics that can be used to facilitate discussions about the risk management, design iteration, and configuration control required to coordinate information transfer between these teams.

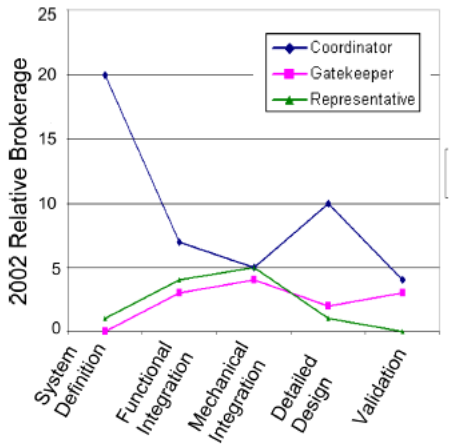


Figure 3.8 Number of tasks in each cluster where the relative brokerage value is greater than 3 [84]

Figure 3.9 depicts the critical task iterations detected according to Simmelian ties. A Simmelian tie is a reciprocal relationship between three nodes. Brokerage analysis revealed iterative relationships that the process architectures believed should be taking place, but which didn't necessarily occur in practice. Two sets involving reliability analysis were poorly executed. The same was the situation for electrical architecture which deals with coordinating design with electrical and mechanical engineering architectures. These were coordination themes that should exist, but which hadn't been successfully implemented.

CRITICAL TASK ITERATION IDENTIFIED BY SIMMELIAN TIES

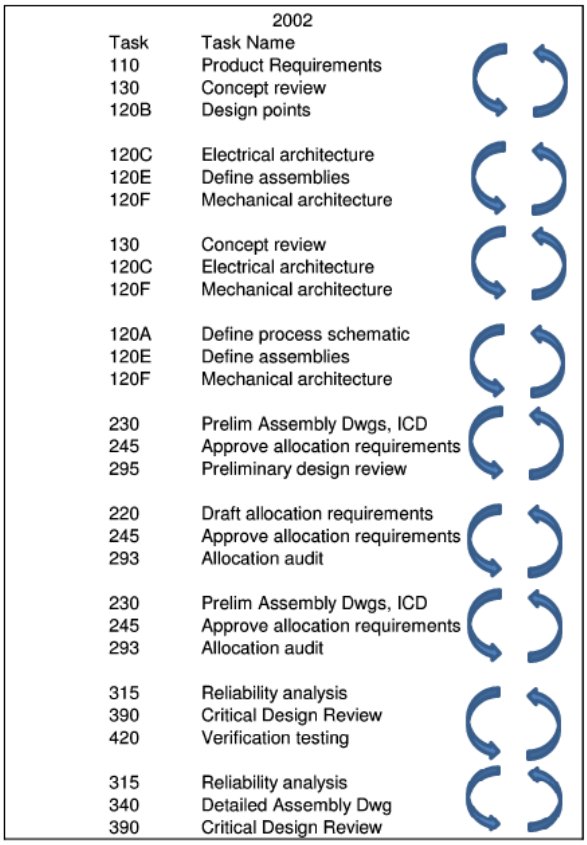


Figure 3.9 Detected critical task iterations[83] (modified by Kantomaa, T.T.)

A final benefit of NA techniques it the ability to visually display large systems of interaction captured in matrix form. Visualization is done here by displaying tasks sequentially (based on expected flow of project execution), and based on key NA metrics. Figure 3.10 presents the network map of the process on this example.

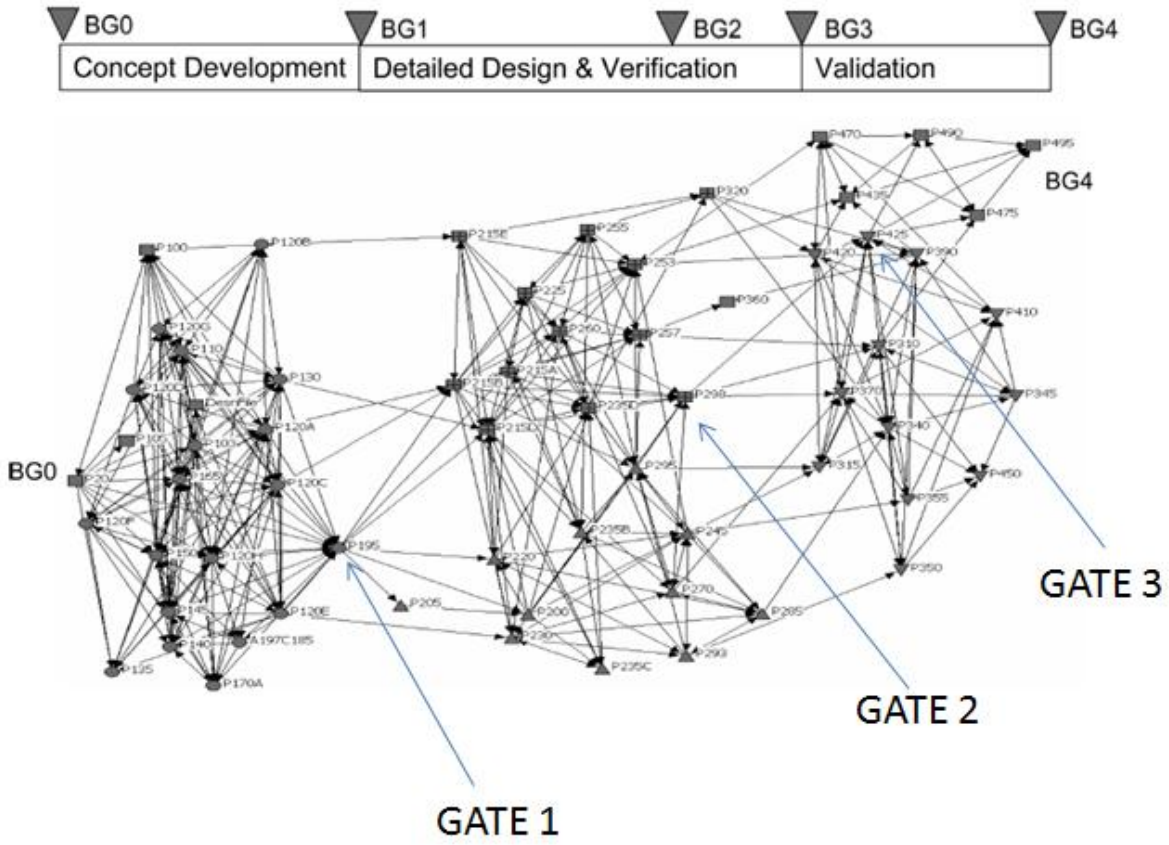


Figure 3.10 Network map[84]

Overlaying the business gate process with the network map reveals that the gates are not acting as the checkpoints they are supposed to. It seems the project is not controlled properly between gates 1 and 4. In the example several managers of the case company actually stated:"We don't really understand the purpose of Business Gates 2 and 3." The visual display confirms the finding from Betweenness centrality section, where it was stated that their importance may have been overrated. Now management has two alternatives. First, redesign the product development process to increase the information control authority of these tasks. Second, use the brokerage analysis to find the tasks that are acting as gatekeepers between phases, and focus energy on making sure those steps are robust.

Figure 3.11 shows a redrawn network map from figure 3.10. It has two dimensions: on the x-axis the influence and on the y-axis the betweenness centrality. The uniqueness of these tasks would not be immediately obvious using a swimlane, value stream map, or even normal network map. Filtering a small list of tasks from the entire product development process enables more effective examination of them. Where are the steps we claim are critical, like technical and business reviews? Should they be showing up with high values on key analytical dimensions? For the relationships on figure 3.11, there would be two questions. First, what tasks should have high control over information flow (betweenness centrality)? Second, what tasks should have the ability to reach the entire product development process with important design information (influence)?

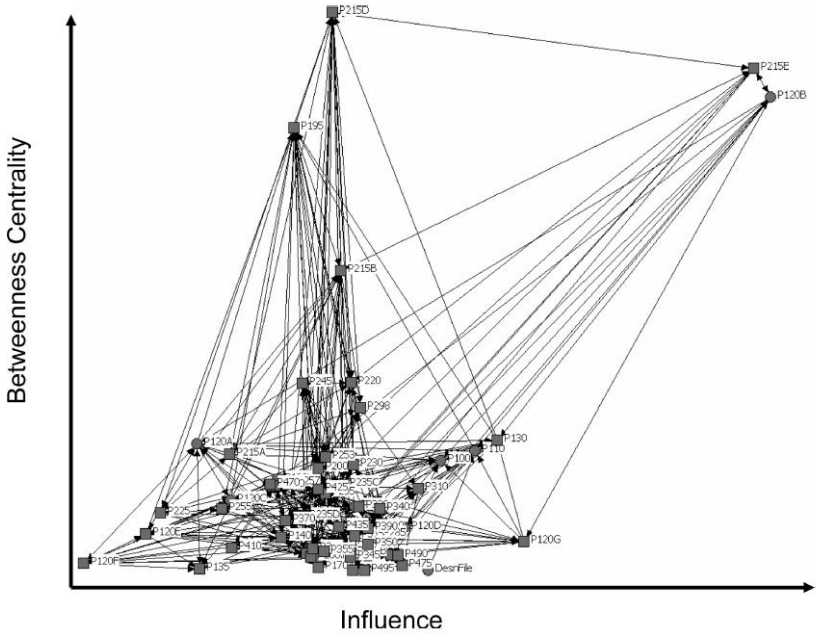


Figure 3.11 Displaying product development process task relationships along key dimensions.[84]

In this example, figure 3.11 was a valuable starting point for discussions with a group of engineering managers in the case company. It validated earlier experience about costly rework resulting from poor design point definition (task 120B that shows on the upper-right corner on figure 3.11).

3.3.2 Limitations of Network Analysis

There is constant development going on relating to the use of NA in product development processes. Current obvious limitations include for example the pointing out of critical iterations only through Simmelian ties. This implies that only iterations consisting of three nodes (not two or four etc.) are considered, although the other critical iterations can be spotted through manual examination. Cluster iteration activity is identified automatically. The adjacent matrix, in this case DSM, is created by using numbers 1 (to show weak relationship) and 2 (to show strong relationship) to indicate relationship. This is inadequate in describing the diversity of task relationship strengths. Here a quantified DSM (QDSM) could be implemented, but it is unknown to the author how the NA tools could cope with this. QDSM defines the relationships on a scale of 0-1 having decimal values as figure 3.12 depicts.

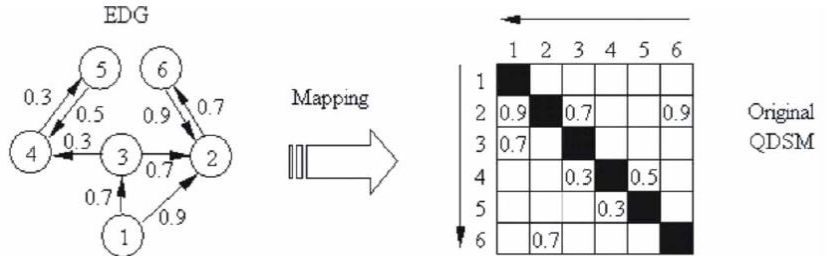


Figure 3.12 Extended Directed Graph (EDG) and Quantified Design Structure Matrix (QDSM)[87]

[87] Luh, D.B., Ko, Y.T., Ma, C.H. (2011): A Structural matrix-based modeling for designing product variety, *Journal of Engineering Design*, 22:1, 1-29

4 ITERATIONS IN PRODUCT DEVELOPMENT PROJECTS

4.1 Definition of iteration in product development

In this study iteration refers to *reworking* or *repeating* product development activities. Rework results from errors during development, whereas repetitions result from generation or incorporation of new information. Repetition means doing the same activity again, but at a different level of abstraction; the information content has changed. The source of new information can be

1) internal, for example, design-build-test cycle may reveal new information
2) external, for example, changing customer demands can result in specification changes

There are two main types of iterations:

- **Intentional iteration** is included in the process and is expected to happen. Intentional iterations allow interdependent activities to exchange knowledge and *converge* to a desirable solution.
- **Unintentional (unplanned) iteration** is the result of errors during development and unexpected arrival of new information which *diverges* the design. This information alters assumptions and causes upstream activities to do *rework*. Causes of unintentional iteration can be divided into **internal** (design flaws, poor communication etc.) and **external** (changes in customer preferences, competitor actions etc.) to the design process. [88] [89] [90]

Unintentional iterations increase NPD project lead-time while intentional iterations in the beginning of a project can reduce uncertainty related to the NPD project and the number of unwanted iterations later on in the project, leading to reduced project lead time. Intentional iterations can also decrease the length of individual design tasks. Instead of progressing in a sequential order, short iteration cycles between design tasks done concurrently may be used to get the design to converge to a desirable solution faster. Table 4.1 depicts these two main types of iteration with three possible sub-types.

Table 4.1 Two main types of iteration with three possible sub-types (Kantomaa, T.T.)

INTENTIONAL ITERATION		
SOURCE	**CAUSE**	**EFFECT**
INTERNAL	*NEW INFORMATION GENERATION*	*REPETITION*
UNINTENTIONAL ITERATION		
SOURCE	**CAUSE**	**EFFECT**
INTERNAL	*ERRORS*	*REWORK*
EXTERNAL	*NEW INFORMATION INCORPORATION*	*REPETITION*

[88] Browning, T.R., Sources of Schedule Risk in Complex System Development, *Systems Engineering* 3:129-142 ©1999 john Wiley & Sons, Inc. CCC 1098-1241/99/030129-14

[89] Costa, Ramon, and Durward K. Sobek II, "Iteration in engineering design: inherent and unavoidable or product of choices made?" DETC2003/DTM-48662, ASME 2003 Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Chicago, Illinois, September 2-6, 2003

[90] Dr Herrmann, J.W., "Managing Problem-Solving Iterations in Product Development", July 7th 2005, University of Maryland, USA

For example, a design-build-test cycle may reveal the need for extra designing. If the need is based on new information gained from the test the iteration can be seen as *intentional iteration* resulting from *new information generation*. The more complex the design is, the more *repetitions* are *expected*. On the other hand, if the need is based on correcting errors then the iteration can be seen as *unintentional iteration.* For example, poor communication in the product development unit leads to *unexpected* design errors and rework. Changes in the markets lead to new information *incorporating* unintentional iterations, which lead to *repetitions*.

There are also special occasions. When entering new application areas or introducing radical new products, the customer requirements may not be fully known in advance of development. These requirements get clearer as the development proceeds and more information is gained through customer feedback. The source of new information resulting in feedback incorporating iteration is external, but the iteration can still not be termed unintentional; in the case of new application area or a radical new product when the customer requirements are not fully known in advance of development customer feedback incorporating iterations are thought as intentional.

The discovery of the need for iteration may come from many events. For example, simulation, testing and information exchange between design areas in coupled design tasks may reveal the need for redesign.

The delay in discovering the need for iteration affects the impact of the iteration in respect to resource consumption and development lead time. If the need to repeat a design activity due to new information is discovered swiftly, thus avoiding a later harmful iteration, the following iteration can be termed *natural* to product development. **Design iteration** is natural, short cycled iteration that occurs between design activities which are dependent on the outcomes of each other, i.e. they are coupled. Design iteration means doing the same design activity again, but at a different level of abstraction; the designer now has less uncertain information content to work on[89].

But consider, for example, a test made at a later phase of development, which reveals the need for redesigning. If the test had been made earlier the redesign need would have been spotted earlier, there would have been less work to be repeated and thus less resources and time would have been consumed. Now, although the cause of the iteration is internal and no actual design errors have been made, this iteration can no longer be considered *natural* iteration. It is termed *harmful*.

Harmful iteration occurs when the need for iteration is realized late and has more severe consequences, than it would have had if its need had been realized on time. Unintentional iterations are always harmful, even if the need for the iteration is discovered swiftly.

Harmful iteration includes:

1) **iteration causing rework**
2) **iteration causing repetition, when then the need for iteration occurs late in the process and has more severe consequences, than it would have had if the need for iteration had been realized on time**
3) **iteration caused by external factors (excluding the special cases of new application areas or radical new products when the customer requirements are not fully known in advance of development and customer feedback incorporating iteration is used in early phases of development)**

4.2 Concurrent engineering

Concurrent engineering (CE), also sometimes called Integrated Product Development (IPD) - although some regard CE as a constituent part of IPD - is an approach where cross-functional teams and parallelization (performing tasks concurrently) of tasks are used in order to increase NPD speed, quality and cost, taking into account user requirements and all elements of the product life cycle. CE aims at discovering errors and the need for redesigns early in the design process, trying to avoid situations where the need for redesign is realized in later stages of the project resulting in large iterations, which lead to schedule and cost overruns.

Elements of CE include:

- **Cross-functional teams:** all the functions needed in developing a new product are integrated.
- **Concurrent activities**: tasks are performed in parallel to increase NPD speed.
- **Constant information sharing**: information sharing between project members is done in a continuous basis.
- **Iterative development in the early phases to ensure smooth back end execution**: the iterative design process is cyclic and major iterations that do occur will occur before the design becomes final.

The traditional sequential model of product development has been changed into an iterative and evolutionary mode. Traditionally, CE has been adopted in single or closely partnered companies. Since CE requires overlapping tasks and increased integration, the need to coordinate activities also increases. This results in integrated technical tools.[45] In effectively implementing knowledge embedment and modular information support within many computer aided systems, such as CAD, CAM, CAPP applications, software's role becomes much more significant as the engineer's design models must be able to "talk" to each other[91] [92].

[91] Shah, J. J., & Mäntylä, M. (1995). *Parametric and feature-based CAD/CAM: Concepts, techniques, and applications.* John Wiley and Sons, Inc.

[92] Otto, H. E., (2001). From concepts to consistent object specifications: Translation of a domain-oriented feature framework into practice. *Journal of Computer Science & Technology,* 16 (3), pp. 208-230

4.3 Iterative development process model

A further enhancement of concurrent engineering is iterative product development[25]. This product development approach is currently being used for rapid software product development. Key to iterative product development is a modular product architecture, in which modules can be inserted or replaced without major impact on the design.[15] Every module should represent an entity that can be tested for functionality. Although iterative development models are mostly used in rapid software development, because the nature of SW development allows modularity, some of the features of iterative development could also be applied more widely.

The building blocks of an iterative development process are an active and early exploration of possible failures and corrective measures. The opportunity that the iterative model gives is the possibility to learn early in the process. This will considerably speed up the learning process and by doing this, also speed up the development process itself. Early iterations enhance the effectiveness of product development when dealing both with high technology and the market uncertainties.[15]

4.3.1 Spiral development model

The biggest competitors to traditional staged models are the variants of more flexible spiral processes that have been adopted from software development (Cooper and Edgett 2008[93]). Spiral development is a family of (software) development processes characterized by repeatedly iterating a set of elemental development processes and managing risk so that it is actively being reduced. Spiral model can be used for a more cost-effective incremental commitment of funds.

Spiral model features include:

- cyclic concurrent engineering
- risk driven determination of process and product
- growing a system through risk-driven experimentation and elaboration
- lowering development cost by early elimination of nonviable alternatives

As a result of planning and risk analysis, different projects – or sometimes even individual spiral cycles inside a project - may choose different processes. That is, the spiral model is actually a risk-driven *process model generator*, in which different risk patterns can lead to choosing one of the following process models:

- incremental
- waterfall (the staged process is called waterfall process by some practitioners[94])
- evolutionary prototyping
- other subsets of the process elements relating to the spiral model

Evolutionary prototyping focuses in gaining fast feedback from early prototypes. This can be advantageous when initial specifications are vague and would benefit from experimentation, but the process has no clearly defined end.[96] A process model answers two main questions: what should be done next and how long should it continue? Under the spiral model the answer to these questions are driven by risk

[93] Cooper, R.G., Edgett, S., 2008. Maximizing productivity in product innovation. *Research-Technology Management,* 51 (2), 47-58

[94] Erdogmus, H. and Williams, L., 2003. The economics of software development by pair programmers. *The Engineering Economist,* 48 (4), pp. 283-319

considerations and vary from project to project and sometimes from one spiral cycle to the next. Each choice of answers generates a different process model. [95]

Product development unit using solely sequences of incremental waterfall developments is not using a spiral process. Waterfall model has the underlying assumptions:

- the requirements are known in advance of implementation
- the requirements have no unresolved, high-risk implications
- the requirements will not change very much during development
- the right architecture for implementing the requirements is well understood

If all the above assumptions are met, the waterfall model becomes a special case of the spiral model. It becomes one possible process type in the arsenal of different process types in spiral development. The traditional idea of complete, consistent, testable requirements specification is not a good idea for certain product components. At some instances, the risk of locking in precise specifications in advance of development involves a high probability of locking incorrect specifications into the development contract, while the risk of not making precise specifications could be low. At other instances, risk patterns make it very important to have precise specifications early on.

Spiral development has two main features: cyclic approach for incrementally growing a system's degree of definition and implementation while decreasing its degree of risk and anchor point milestones for ensuring stakeholder commitment to feasible and mutually satisfactory system solutions. Anchor point milestones drive the spiral to progress toward completion and offer a means to compare progress between one spiral project and another. Figure 4.1 depicts a general spiral process.

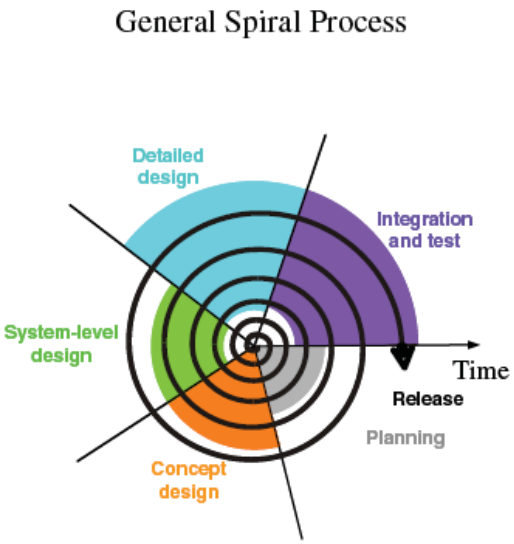


Figure 4.1 General spiral process[96]

[95] Boehm, B., edited by Hansen, W.J. "Spiral Development: Experience, Principles, and Refinements", Spiral Development Workshop, February 9, 2000, Carnegie Mellon, Software Engineering Institute

[96] Darian Unger& Steven Eppinger (2011) *Improving product development process design: a method for managing information flows, risks and iterations* Journal of Engineering design, 22:10, 689-699

As figure 4.1 depicts a single spiral sequence doesn't necessarily contain all the phases in spiral development. Further, the elements of the spiral development don't need to be visited in a precise order.[95]

The spiral process also has disadvantages. First, its complexity requires significant management attention. Second, the lack of rigid specifications can potentially lead to delays in developing complex subsystems[97].

[97] Unger, D., 2003 *Product Development process design: improving company response to market pressure, regulation, and changing customer needs.* Thesis (PhD). Cambridge, Massachusetts Institute of Technology.

4.4 Overlapping activities

Overlapping product development activities can help firms develop products faster[98]. Overlapping development activities may save time, but it may also increase iterations or reworking for the downstream activities and increase resource usage. Project managers should be careful in deciding whether two activities can be overlapped and how much they should be overlapped.[99]

Studies of the structures of many industrial product development processes suggest that the flow of information and the process execution in practice is largely sequential, with information being generated and finalized by the upstream activities before being absorbed by the downstream development activities[98]. Figure 4.2 (a) illustrates such a sequential process where a downstream activity (such as product prototyping) receives and utilizes design information only after it is finalized by the upstream activity (such as concept development). One approach to accelerating such a process involves removing the dependency between the activities, as shown in Figure 4.2 (b), thereby enabling the two activities to be executed in parallel. However, such an action generally requires a fundamental redefinition of the development activities and/or the product architecture, which may be undesirable or difficult at best. [98] One approach to remove task dependencies is to clearly define an interface between two interacting components early in the design process, so that the subsequent design of the two components can proceed independently and in parallel. Another way to reduce project lead time is to consider sets of solutions (set-based approach) instead of point-value estimates of design parameters. In some cases the use of sets or ranges may facilitate faster convergence of coupled tasks. This means the downstream activity would start its development activities with a set of possible outcomes regarding the upstream activity and come up with a number of solutions, each one applying to one of the possible outcomes. This approach is only suitable in some situations.

An alternative approach to acceleration is to overlap the activities through more frequent exchange of preliminary information (see Figure 4.2 (c)). In contrast to the one-time transfer of finalized information in a sequential process, frequent exchange of design information enables the concurrent execution of dependent activities in an overlapped process. In the sequential process of figure 4.2 (a), the downstream activity doesn't begin until finalized information (shown using a black arrows) is available at the completion of the upstream activity. In the overlapped process, the downstream activity begins earlier by using preliminary information (shown using hollow arrows).

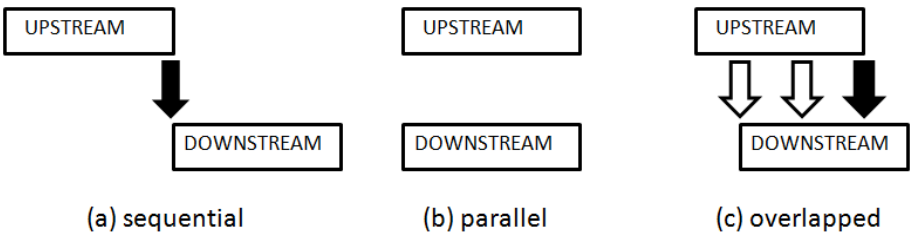


Figure 4.2 Sequential, parallel and overlapped processes (Kantomaa, T.T.)

[98] Krishnan, V., Eppinger, S.D., Whitney, D.E., "A Model-Based Framework to Overlap Product Development Activities", *Management Science,* Vol. 43, No. 4, April 1997

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

In order to overlap nominally sequential activities, the pattern of information exchange between them needs to be altered. This may involve disaggregating the exchanged information into parts. Other parts of information are exchanged multiple times in a preliminary form and other parts only once but in a finalized form at the completion or before the completion of the upstream activity. The question now is which parts should be utilized by the downstream activity only in finalized form, and which parts may be used in preliminary form?

THE RISKS UNDERLYING OVERLAPPING

The reduction in lead time in overlapping activities doesn't always equal the overlap period. This is because the duration of the downstream activity may be altered in converting the sequential process into an overlapped process. When preliminary upstream information is utilized by the downstream activity too early, then the future changes have to be incorporated in time consuming iterations, resulting in an increase in the downstream duration and resource consumption.

Krishnan et al suggest that the overlapping of activities should be done according to two properties of the design process; information evolution and sensitivity.[98]

4.4.1 Upstream information evolution

In product development each development activity is responsible for generating the required product or process knowledge based on the received input information from other activities. The information evolution of a development activity is considered to as a gradual refinement of the product or process knowledge from its preliminary form to a final form over the period of activity execution. Faster evolution represents an early resolution of technical uncertainty.[80] Figure 4.3 depicts how an upstream task's information evolution narrows the exchanged parameter to a final value.

Overlapping may be facilitated when realizing that the upstream activity may continually narrow and refine the generated information. Here, preliminary upstream design information is presented as an interval value.

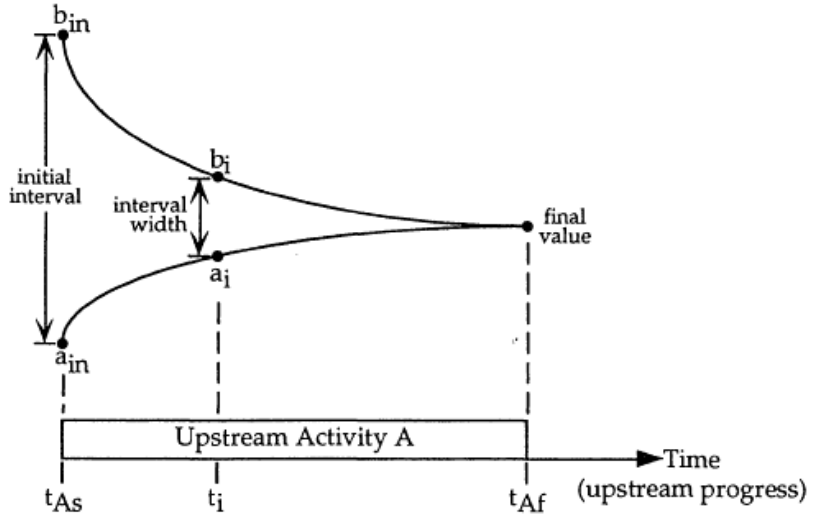


Figure 4.3 Upstream activity narrows the exchanged parameter to a final value[98]

In the preliminary stages, the exact value of parameter X at t_i, X_i is unknown, but an interval $\{a_i, b_i\}$ is known such that $a_i \leq X_i \leq b_i$. At the beginning of the upstream activity, let this interval be $\{a_{in}, b_{in}\}$. It is also called the *initial interval*. During the course of the upstream activity, as product development is carried out, and as external inputs are received, the parameter X is gradually narrowed to its final value X_f, a point within the initial interval. The interval within which the parameter is known changes continuously until when the parameter attains its final value at t_{Af}.

The degree of information evolution (Є) measures how close the incomplete design parameter is to its final value. The smaller the interval width is, the closer the parameter is to its final value. The degree of information evolution at t_i is expressed by the formula:

$$\in_i = 1 - \frac{b_i - a_i}{b_{in} - a_{in}} \quad (1)$$

For simplicity, the degree of evolution is defined to be a linear function of the width of the design interval. The evolution is said to be monotonic when for $t_j > t_i$, the interval $\{a_j, b\}_j$ lies within $\{a_i, b_i\}$.

A plot of information evolution is called the *evolution function*. The evolution function indicates how (at which rate) the upstream activity narrows the width of the parameter interval to zero. With the evolution function and the initial interval as input, it is possible to determine the interval width at various points in time during the design process, but not the exact interval endpoints of the parameter X. [98]

4.4.2 Downstream iteration sensitivity

In the overlapped process, the upstream activity shares preliminary upstream information available in the form of interval endpoints with the downstream activity. In many practical situations, however, the downstream activities (such as prototyping) require a point value of the parameter X to perform development activities. It is assumed that, when the downstream activity begins to perform a development activity at time t_i ($<t_{Af}$), it uses the expected value, x_i, of the exchanged information X_i. Further, downstream iterations in the overlapped process are used to accommodate the changes in the upstream information in the downstream development.

Downstream sensitivity means the relationship between the duration of downstream iteration and the magnitude of change in the upstream information value. The function relating to downstream iteration duration, d_i, and the magnitude of change in the upstream activity, Δx, is called the *sensitivity function,* ϕ(Δx). Except for a few special cases, the sensitivity function for product development processes is non-decreasing, i.e., larger changes in the value of the exchanged information require longer iterations to process those changes.

4.4.3 Implications of Upstream evolution and downstream sensitivity

Understanding upstream evolution and downstream sensitivity has several implications for managing the design process. If the evolution of the upstream information is such that large changes were to happen near the completion of the upstream activity, it would be difficult to reduce the lead time by overlapping – as the subsequent design iterations would be of significantly large duration. However, if the evolution of the design information is such that major changes occur during the initial period of the upstream activity, and only minor changes happen during the final period of the upstream activity, then overlapping is more likely to help reduce lead time.

If the design information evolves such that as the upstream activity progresses the increase in the degree of evolution during a given interval of time gets progressively smaller, then the amount of change decrease as the upstream activity progresses. This is termed *fast evolution*, since major changes happen early in the design process. When the increase in the degree of evolution gets progressively larger as the upstream activity progresses, the change in the exchanged information also increases as the upstream activity progresses, and the evolution is said to be *slow*. This happens especially when:

- the activity has interfaces with several other change-prone activities
- information is required from an extraneous factor, and it is not available until late in the process
- the activity involves solving a complex technical problem

In these cases finalizing upstream information early in the upstream process would either be impossible or impractical.

Downstream sensitivity is said to be low when substantial changes in the exchanged information value used by the downstream activity can be accommodated with ease. Downstream sensitivity is said to be high when even small changes require design iterations of large magnitude. Figure 4.4 depicts these functions.

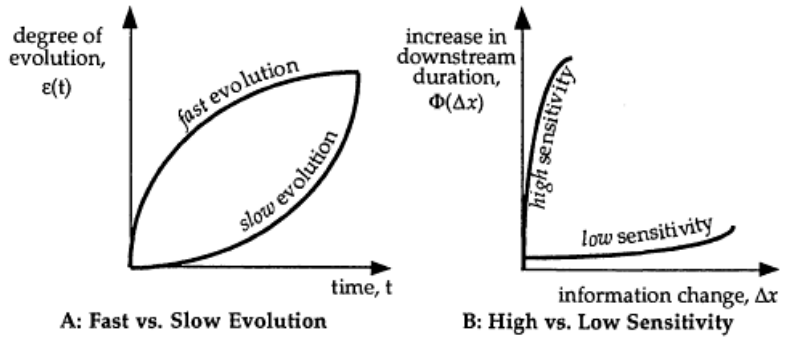


Figure 4.4 Degree of information evolution and sensitivity[98]

Figure 4.5 shows four special cases of these two functions; when the upstream evolution is fast or slow combined with when the downstream sensitivity is high or low. The mechanism used for overlapping differs for each combination.

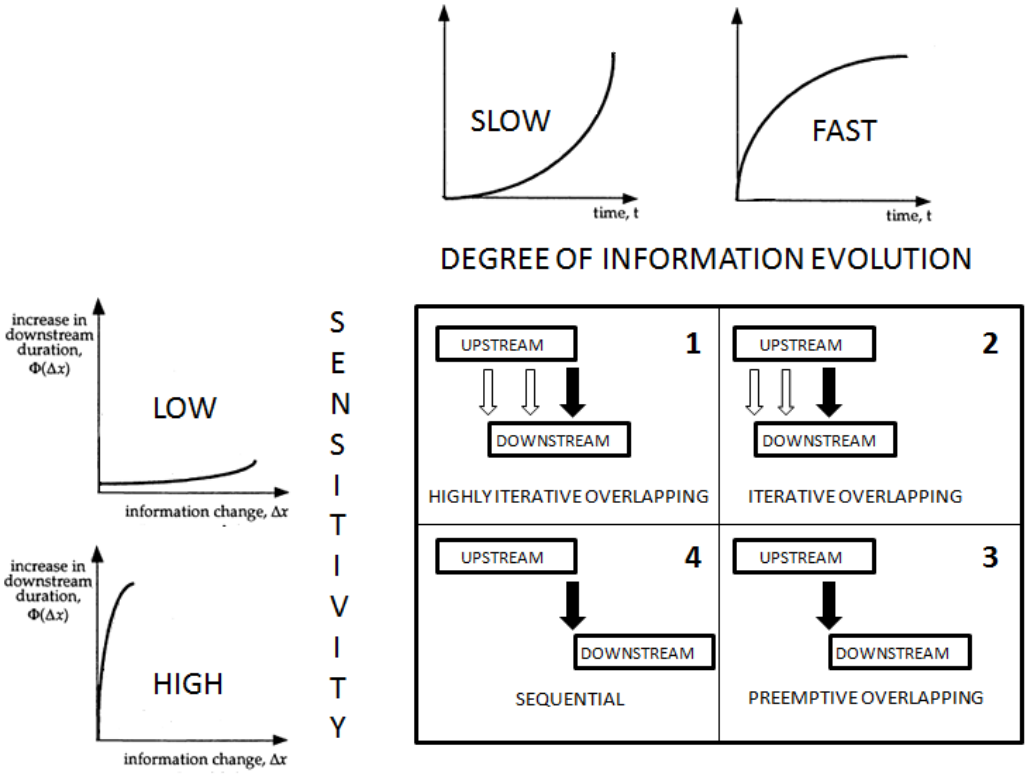


Figure 4.5 Types of overlapping based on evolution and sensitivity (Kantomaa, T.T. modified from[98])

Case 1: When the sensitivity is low, it is possible to commit downstream resources based on preliminary upstream information. In this case, even if the changes in the exchanged information are large, their effects on the downstream activity are not. Further, when the upstream evolution is slow – major changes happen until late into the upstream process before which information cannot be finalized – then the overlapping is said to be *highly iterative*. In this case, the activities are overlapped by beginning downstream activity with preliminary information, and incorporating design changes in subsequent downstream iterations. The design information is not finalized until the nominal completion of the upstream activity.

Case 2: When both the upstream information evolves rapidly and the downstream sensitivity is low, it is possible to both start downstream activity with advance information and to preempt late changes in the exchanged upstream information. This overlapping is also *iterative*, but there are only few (or none) iterations.

Case 3: Downstream sensitivity is high, but the upstream information evolution is fast (information can be finalized early in the upstream activity). In this case activities are *preemptively overlapped,* meaning the exchanged information is to be preempted by taking its final value at an early point in time. There are no subsequent downstream iterations.

Case 4: The last scenario occurs when downstream sensitivity is high and the upstream information evolution is slow. In such a case, activities are done in sequential order.

Often the evolution and sensitivity of the components may be different from the aggregated information[99]. In such a case, the exchanged information is disaggregated into components to see if some information

evolves more quickly or is of use to a downstream activity in preliminary form (since some part in the downstream activity has low sensitivity). The overlapping scheme might be changed for these components.

It is noteworthy that the different types of overlapping result in different tradeoffs among the performance parameters. In iterative development for instance, more downstream resource consumption is needed for reduced lead time.[98]

The presented four situations were special cases. There is continuity between these extreme cases. Managerial discretion should be used in defining where to use overlapping and to what extent.

4.4.4 Coupled design tasks

So far we have contemplated situations where a task is dependent on another task, but not the other way around. Coupled tasks are mutually dependent, i.e., each task requires the result of the other tasks in order to be completed. Coupled tasks either must be executed simultaneously with continual exchanges of information or must be carried out in an iterative fashion. When coupled tasks are completed iteratively, the tasks are performed either sequentially or simultaneously with the understanding that the results are tentative and that each task will most likely be repeated one or more times until the team converges on a solution. Figure 4.6 depicts these task alignments.

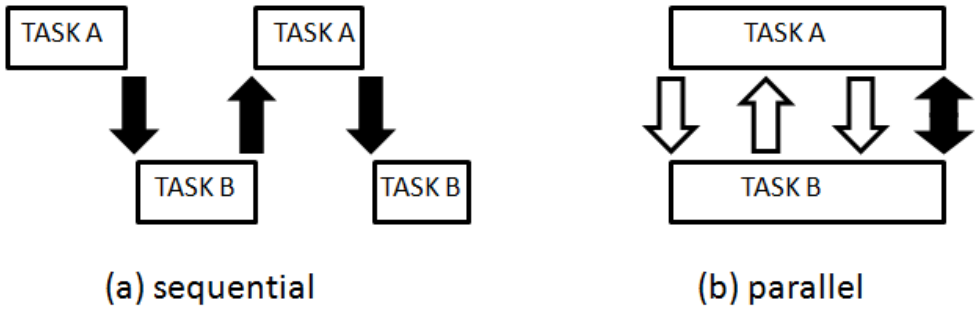


Figure 4.6 Coupled design activities (Kantomaa, T.T.)

If decoupling of activities is not possible, then the way to perform the iterations more quickly is to form multi-functional teams where team members work closely together; the mechanical engineer, electrical engineer and testing personnel work closely together and exchange information rapidly.

The scheme to schedule coupled tasks differs from scheduling non-coupled tasks. Now managers have to take into consideration the order in which to perform the tasks (unless executed in parallel). The order of tasks affects the speed of convergence (different tasks have different information evolution) and the resource consumption of the individual activities. The task that starts the sequence has relatively more work than if it would not be the sequence initiating task, since the initiating task has more uncertain information available.

Managerial discretion is used to consider information evolution, activity sensitivity, and resource availability of different activities simultaneously to sequence the tasks according to the desired trade-off between speed and resource consumption.

4.5 Controlling NPD iterations through matrix representation

In this chapter mathematical methods for controlling iterations are presented. These presentations may be somewhat too complex to be implemented in practice, but nevertheless they show, by means of mathematics, the basic principles of controlling iterations. Understanding these principles help managers in controlling iterations just by the means of common sense.

The interdependencies among design tasks give rise to complex information flows as the execution of a design task may create additional information or conditions that affect other independent tasks. As all the coupled tasks are executed concurrently, design decisions made using incomplete or imperfect information are revisited through *design iteration*. [100]

Concurrent design iteration can be expressed by the state equation:

$$\boldsymbol{x}(k+1) = \mathbf{A}\boldsymbol{x}(k) + \mathbf{B}\boldsymbol{u}(k) \tag{2}$$

where *k*, a discrete variable, represents a finite number of iterations, ***x(k)*** is the task state vector where each variable x_i represents the fraction of the initial work remaining of that task after *k*th iteration. **A** is the DSM, or actually a transformation of it called Work Transformation Matrix (WTM). WTM shows how working on a specific task creates rework for other dependent tasks. **B** is a matrix representing shared resources between tasks and **u**(k) is a matrix representing the additional resources required by tasks, resulting from external disturbances, in order for the entire design project to arrive at the desired states. The control input of a task u_i may be the additional resources required by a design task to cope with or to reduce the amount of rework. In figure 4.7 a WTM of four coupled tasks is presented. The rework created from completing a task can be seen from this matrix. For example if task D is completely worked on in a certain stage of iteration, then 10% of the work of task C, 30% of task E and 10% of task F must be redone in the subsequent stage of iteration (every column corresponds to a task's completion effects on other tasks).

$$\mathbf{A} = \begin{array}{c} \begin{matrix} \boldsymbol{C} & \boldsymbol{D} & \boldsymbol{E} & \boldsymbol{F} \end{matrix} \\ \begin{bmatrix} 0 & 0.1 & 0.2 & 0.3 \\ 0.3 & 0 & 0.4 & 0.2 \\ 0.1 & 0.3 & 0 & 0.5 \\ 0.1 & 0.1 & 0.2 & 0 \end{bmatrix} \end{array} \begin{matrix} \boldsymbol{C} \\ \boldsymbol{D} \\ \boldsymbol{E} \\ \boldsymbol{F} \end{matrix}$$

Figure 4.7 WTM[100]

The open-loop state-space representation of equation (2) incorporating the influence of external disturbances is termed *non-homogenous state-space system* (NHSS).

A *homogenous state-space system* (HSS) does not consider external disturbances:

$$\boldsymbol{x}(k+1) = \mathbf{A}\boldsymbol{x}(k) \tag{3}$$

There are three assumptions before linear algebraic analysis can be done:

- All tasks are executed concurrently

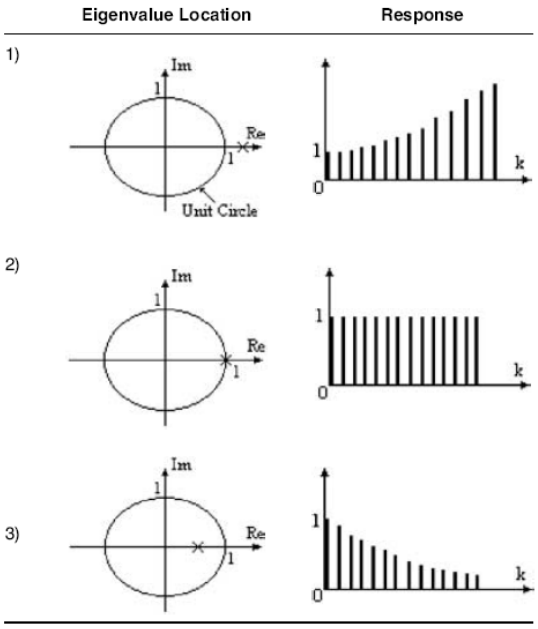
[100] Lee, S.G., Ong, K.L. and Khoo, L.P. *Control and monitoring of concurrent design tasks in a dynamic environment* Concurrent engineering, Vol 12 number 1 March 2004

- The quantum of rework of a task is a linear function of the work done by other coupled tasks in the preceding stage of iteration
- The parameters do not vary with time.

The second and third assumptions are difficult to justify empirically, although it has been observed that the time per iteration does decrease as the design tasks progress [101]. Nevertheless, this presentation serves our purposes well in illustrating the basic mechanism in how iterations can be managed.

4.5.1 Stability of design tasks

The stability of a system may be derived from the eigenvalues of the WTM matrix A. The WTM can have many eigenvalues and the critical eigenvalue is the one that has the largest real value. Analogy to the stability of an electrical system can be found here as figure 4.8 illustrates



- If λ> 1, the amount of remaining work increases with time, i.e. the entire design project is running out of control. The tasks become more laborious as the number of iterations increases. In real life, this scenario may be attributed to ill-defined tasks or unrealistic specifications.
- If λ= 1, the amount of remaining work is constant, i.e. despite the infusion of resources, the amount of work associated with each task remains the same. In real life, this situation could be due to ill-defined objectives or incomplete information.
- If 0< λ< 1, the amount of remaining work will be monotonically decreasing, i.e. the entire design project will converge to its completion.

Figure 4.8 Stability of a system comprised of coupled tasks[100]

System representation of task interdependencies involves no imaginary numbers, so positioning the eigenvalue in the unit circle is done only in one dimension, along the x-axis.

The next chapter shows that even divergent systems can be made to converge through feedback control.

4.5.2 Controlling iterations through feedback loop

If the system is initially divergent or if the system is stable, but the convergence rate of design tasks is slower than expected, *state feedback control* (SFC) can be employed.

As discussed in the previous chapter the stability of a system depends on the eigenvalues of the WTM matrix. In order to achieve the desired stability, an appropriate *state feedback gain matrix*, **K**, is defined. In the case of concurrent design iteration, the **K** matrix encompasses the degree of control exerted by one task on others. The control input of an open-loop system shown in equation (2) is assumed to be a vector:

[101] Smith, R.P., and Tjandra, P. (1998). Experimental Observation of Iteration in Engineering Design, *Research in Engineering design,* **10**:107-117

$$\boldsymbol{u}(k) = -\mathbf{K}\boldsymbol{x}(k) \tag{4}$$

where **K** is the state feedback gain matrix.

By substituting equation (4) into equation (2) and assuming that resources are not shared among various tasks (i.e. the **B** matrix is an identity matrix), the closed-loop state feedback equation becomes

$$\begin{aligned} \boldsymbol{x}(k+1) &= (\mathbf{A} - \mathbf{IK})\boldsymbol{x}(k) \\ &= \mathbf{A}^*\boldsymbol{x}(k) \end{aligned} \tag{5}$$

where $\mathbf{A}^* = (\mathbf{A} - \mathbf{IK})$, **I** is an identity matrix.

The stability of the closed-loop state feedback depends on the eigenvalues of the $\mathbf{A}^*$matrix, notably on the gain matrix **K.**

The state of the system at the (*k*+1)th stage of iteration, ***x***(*k*+1), is now the summation of the homogenous state **A*x****(k)* and the control input ***u****(k)*, which is a function of the preceding state ***x****(k)* and the gain matrix **K**.

When the state of the previous stage *k* deviates from its expected value because of some disturbance, the control input ***u****(k)* and the gain matrix **K** can be revised to cope with the deviation.

EXAMPLE OF NATURALLY CONVERGING SYSTEM

From the stability analysis the system which has a WTM as depicted earlier in figure 4.7 is stable since the largest eigenvalue of **A** is 0,674. Table 4.2 presents the natural response of tasks C, D, E and F. These values are derived from the WTM matrix **A** in figure 4.7.

Table 4.2 Natural (slowly converging) response of tasks C, D, E and F[100]

k	x_C	x_D	x_E	x_F
1	0.600	0.900	0.900	0.400
2	0.390	0.620	0.530	0.330
3	0.267	0.395	0.390	0.207
4	0.180	0.278	0.249	0.144
5	0.121	0.182	0.173	0.096
6	0.082	0.125	0.115	0.065
7	0.055	0.083	0.078	0.044
8	0.037	0.056	0.052	0.029
9	0.025	0.038	0.035	0.020
10	0.017	0.026	0.024	0.013

The slowest converging task is D since it takes at least 7 iterations before the percentage of remaining work is 10% of the original (a task is considered to be finished when less than 10% of the original work is remaining). However, consider if it was wanted that task D was completed after only four iterations. Now the state feedback control is employed. By setting $x_D(4)$=0,1, the desired eigenvalue and eigenvector matrices are obtained (using a specific eigenstructure algorithm) and the gain matrix **K** is computed as figure 4.9 depicts. The elements of gain matrix denote the degree of influence of other tasks on the resource usage of a task. For instance, 0.1741 in the second row denotes that when the work of task C at stage *k is* 100%, then task D needs an extra 17,4% of its original resources to do the additional work. Negative elements in the same row imply that fewer additional resources and less additional work are needed by tasks D and F. This is intuitively true as working on a task helps to reduce its own rework. Hence,

all the diagonal values of **K** are negative. The design manager can identify the tasks that cause other tasks to save or consume resources by inspecting the elements of the **K** matrix.

$$\mathbf{K} = \begin{bmatrix} -0.0819 & 0.0094 & 0.0758 & 0.1445 \\ 0.1741 & -0.1393 & 0.2089 & -0.0393 \\ -0.0171 & 0.1704 & -0.1778 & 0.2775 \\ 0.0343 & 0.0273 & 0.1002 & -0.1249 \end{bmatrix}$$

Figure 4.9 Gain matrix K[100]

It can be seen from table 4.3 (a) that in order that the amount of work remaining for task D at the fourth stage of iteration is 10%, the amount of resources allocated to task D, u_D, at the beginning should be 20,5% more. At iteration stage *k*=1 the amount of remaining work for task D is 69,5% and an additional 10,4% of resources are needed to achieve the target of 4 iterations. The negative sign of elements of the control input ***u(k)*** implies that extra work or resources are needed.

Table 4.3 Table of remaining work proportions after each iteration and the additional resources needed[100]

k	x_C	x_D	x_E	x_F	u_C	u_D	u_E	u_F
(a) Naturally slowly converging								
0	1	1	1	1	−0.144	−0.205	−0.254	−0.037
1	0.457	0.695	0.646	0.363	−0.068	−0.104	−0.097	−0.054
2	0.239	0.364	0.339	0.190	−0.036	−0.055	−0.051	−0.028
3	0.125	0.191	0.178	0.100	−0.019	−0.029	−0.027	−0.015
4	0.066	0.100	0.093	0.052				
(b) Naturally unstable								
0	1	1	1	1	−0.579	−1.243	−0.964	−0.674
1	0.421	0.657	0.536	0.526	−0.368	−0.574	−0.469	−0.460
2	0.225	0.351	0.286	0.281	−0.197	−0.307	−0.250	−0.245
3	0.120	0.187	0.153	0.150	−0.105	−0.164	−0.134	−0.131
4	0.064	0.100	0.082	0.080				

EXAMPLE OF INITIALLY UNSTABLE SYSTEM

Let us consider a situation where we have a design task system that has a WTM of figure 4.10.

$$\mathbf{A} = \begin{bmatrix} 0 & 0.5 & 0.2 & 0.3 \\ 0.8 & 0 & 0.9 & 0.2 \\ 0.7 & 0.3 & 0 & 0.5 \\ 0.1 & 0.9 & 0.2 & 0 \end{bmatrix}$$

Figure 4.10 WTM of an initially unstable system[100]

The critical eigenvalues of the system, λ_1 is 1,4081 which indicates the system is initially unstable. To attain the same desired status, i.e. $x_D(4)$=0,1, the gain matrix **K** is computed as figure 4.11 shows. Through state feedback control, task D converges to 10% of its initial work after only four iterations. It can be seen from table 4.3 (b) that in order for the amount of work remaining in task D to be 10% at the fourth stage of iteration (*k*=4), at the initial stage (i.e. *k*=0) the amount of resources allocated to task D, u_D, should be 124% more than planned. At iteration stage *k*=1 the amount of remaining work in task D is 65,7%. At stage *k*=1, additional 57,4% of resources for task D must be allocated and so on. It can be seen that the amount of additional resources required to stabilize an initially unstable task is much greater than the amount of

additional resources required to expedite the task if it is stable initially. This is because more work is needed to rectify the causes of instability of an unstable design task. Figure 4.12 shows the natural (unstable) and controlled response of task D.

$$\mathbf{K} = \begin{bmatrix} -0.1230 & 0.3837 & 0.0969 & 0.2207 \\ 0.6085 & -0.1812 & 0.7395 & 0.0765 \\ 0.5438 & 0.1522 & -0.1310 & 0.3992 \\ -0.0534 & 0.7549 & 0.0714 & -0.0990 \end{bmatrix}$$

Figure 4.11 Gain matrix K of an initially unstable system[100]

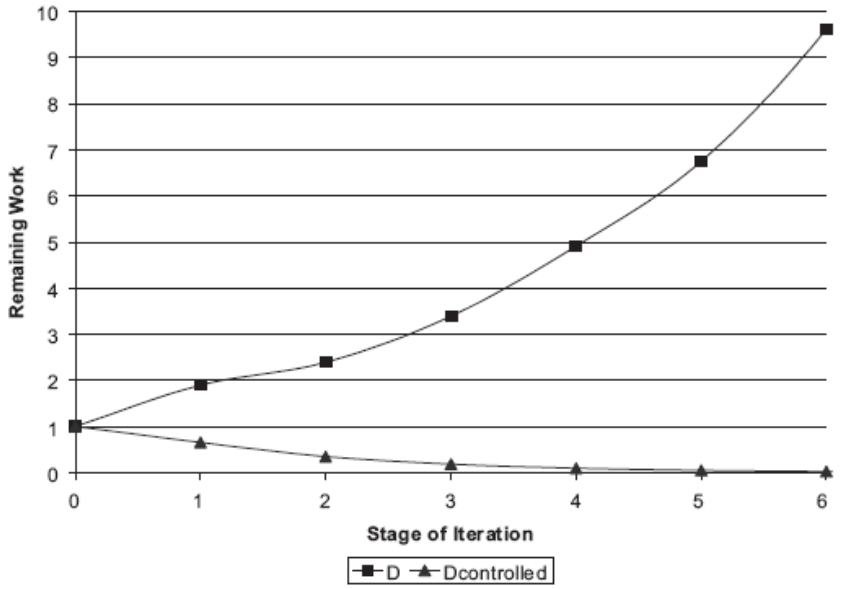


Figure 4.12 Natural (unstable) and controlled response of task D[100]

KEEPING ITERATIONS CONSTANT IN THE FACE OF DISTURBANCES

If the response of a system is initially within the desired response, at every stage of iteration *k*, the state of the following stage may be predicted by substituting the state of the current stage ***x(k)*** into the homogenous state equation (shown already earlier):

$$\boldsymbol{x}(k+1) = \mathbf{A}\boldsymbol{x}(k) \tag{2}$$

If the response lies outside the desired response, then state feedback control can bring it back within the desired response. The way this is done is exactly the same as presented in the previous examples. This way the design team can dynamically counter disturbances in the tasks to keep the number of iterations constant.

PRACTICAL NOTIFICATIONS

The above presented mathematical presentation assumed linearity although non-linear concurrent design tasks are often the case in real life.

Another thing to consider is the fact that it is not always desirable to make the number of iterations as small as possible. Browning and Eppinger (2002) account for activity overlapping and show that minimal iteration is not optimal when tasks of long length can be moved off the critical path through iterative development [102]. Then consider *threshold* iterations; the minimum number of iterations that must be made to accomplish a set of tasks. The number of iterations cannot fall below the number of threshold iterations relating to coupled tasks.

Negatively correlated workloads

Negatively correlated workloads appear in situations of coupled tasks where the amount of work can be shifted from work function to another. Consider for example main circuit design and mechanical design. If mechanical design and thermal simulations are done accurately from the very beginning of the design process, it will help the convergence of main circuit design. On the other hand, if main circuit design were let to evolve further before mechanical design would be made accurately this would diminish mechanical resource consumption, but consume more main circuit design resources, since main circuit design would be based on less accurate information in the beginning of the design process. This way workload can be shifted from a function to another.

Negatively correlated workloads appear in situations of coupled tasks where the amount of work can be shifted from work function to another. The effect of negative correlation between workloads was also neglected in the mathematical processing of iterations, i.e. the amount of work generated to other tasks from working on a specific task was always presented by a positive sign. Taking into account negatively correlated workloads the situation might be that after a certain point, working more on a specific task would actually diminish another tasks remaining work.

[102] Browning, T.R., S.D. Eppinger 2002 Modeling impacts of process architecture on cost and schedule risk in product development. *IEEE Transactions on Engineering Management* **49**(4) 428-442

5 RESULTS

5.1 Empirical part of the study

The empirical part of the study was commenced after the theoretical study. The main source of information for the empirical part was the elaborate interviews. A total of 37 people were interviewed; 15 designers, 3 main designers, 9 project managers, 2 product managers, 1 line manager, program manager, 1 product development manager and 5 people from other functions (sourcing, production, manual team). These interviews lasted from 1 hour to as long as 5 hours. The average time was somewhere around 2 hours. There is no interview template in the appendices, since all the interviews were individualized according to the function and personal history of the interviewee. If all the questions and topics asked in different interview sessions were to be listed the list would go on for tens of pages so there is no sense in doing so.

Documents found in corporate databases were also studied: specifications, product determination table documents, project descriptions, project schedules, project risk identification documents, project steering group meeting documents etc. All and all a comprehensive empirical research was made to gain understanding about the practicalities and constraints of the present situation in the company's product development environment.

Product development in four profit centers was examined in the study. These profit centers include High Power Drives (HPD), Low Power AC (LAC), Wind AC (WAC) and Solar inverters. The main attention was in HPD profit center, LAC profit center was also examined, whereas Wind AC and Solar inverter profit centers were left on less attention. However, these two previously mentioned profit centers affect HPD projects, since they have common base-projects, and their effect on these projects was examined.

5.2 Lean principles

A set of 9 guiding lean principles was formed earlier in this study. Although lean principles were not among the main research topics of this study, the way these 9 lean principles are presently utilized in the company were briefly examined in this study. Table 5.1 summarizes the way Lean principles are presently applied in the company.

Table 5.1 Lean principles application in the company (Kantomaa, T.T.)

LEAN PRINCIPLE	FULLY APPLIED	APPLIED EXTENSIVELY	SLIGHTLY APPLIED	NOT APPLIED
ELIMINATE WASTE	SUBJECTIVE			
MODULARITY IN DESIGN		X		
WORKLOAD LEVELING			X	
CONCURRENT ENGINEERING	X			
CROSS-PROJECT KNOWLEDGE TRANSFER			X	
PROCESS STANDARDIZATION			X	
DELIVER AS FAST AS POSSIBLE	SUBJECTIVE			
EMPOWER THE TEAM	SUBJECTIVE			
BUILD INTEGRITY IN	SUBJECTIVE			

Some of the lean principles' applications in table 5.1 have been labeled as "subjective". This is because the "correct" application of these principles is highly dependent on the assessor. Table 5.1 shows that lean principles are not fully adhered to in the company, which can be accounted for by the complexity of the product development environment in the company. Monetary considerations also explain why, for example, modularity in design is not fully applied in the company. Modularity in design incorporates platform approach and design flexibility for being able to adapt to design changes more easily (designing excess capacity, making more spacious designs etc). The former has bidirectional effects on the end products costs case by case, but the latter will inevitably increase the costs of the end product. Therefore, it is not explicit that the set of guiding lean principles even should be fully applied. The following chapters go through each of the principles one by one.

5.2.1 Eliminate waste

In the theoretical part of this study, activities in PD were divided to value adding work, necessary waste and waste. In this section activities in PD are classified in a slightly different way for better applicability. Here activities in product development are categorized as:

- **Actual design work** - this category also includes building prototypes, participating in tests, working with suppliers in the case of customized components etc.
- **Abstract design work** - not actual design work, but aiding in design work, i.e. meetings and conversations with colleagues about work related issues.
- **Work done primarily to support other functions** - not actual design work, not aiding in design work, but work that needs to be done, primarily to support other functions, i.e. production, sales, service etc. This category includes BOM's, manuals, brochures, aiding in service plans, aiding in making training materials etc.
- **Clear waste** – this category includes coffee breaks, waiting in projects, using work time on own agenda etc.

Noteworthy is that categories (A) to (C) also include waste. For example, multitasking, generating defective information and overproducing information are all forms of waste, but they are included in actual design work in this categorization. The designers practically always have something to do in different projects so category "Clear waste" primarily includes breaks and using work time on own agenda; when idle time in some project or sub-project occurs, the designer proceeds to work on another project.

The reason for this kind of categorization was that a questionnaire for the designers was made concerning how they spend their work time. It would have been extremely difficult for the designers to estimate their time spent on value creating activities and time spent on waste since the boundary between value adding and non-value adding work in PD is a bit unclear, to say the least. Further, it depends on the point of view. Think about a situation where one design option is at later phase realized to be unfeasible. Has the work so far made by the design engineer been waste, or has it been value adding since by doing this trial-and-error cycle the design team has learned something that will help them in the next design round. Making estimations on time spent on different activities becomes much easier when the activities are divided into the four categories presented above.

A total of 17 designers from one profit center (High Power Drives, 4 from each team, except 5 from main circuit design team) responded to the questionnaire. Designers were asked to estimate their time spent on these four different categories but estimations on category "Clear waste" were not considered; only the ratios between the three other categories were considered. This was done because people tend to

underestimate their idle time. Category's "Clear waste" percent share was an estimation made by the author based on 3 years of experience of working in the PD environment in question. The ratios of the other three categories were compared to their aggregated sum, and the resulting percentages (totaling 100%) were then multiplied by a factor of 0,8 to form a "Clear Waste" section of 20%. This is how the percentages were calculated and the designer's estimation of their own idle time omitted.

Designers were also asked to make the estimations based on a long time average, not incorporating just the latest periods. To form a sample that would be representative to reality, experienced designers along with less experienced designers were incorporated in the questionnaire with a right distribution.

The results of the questionnaire are presented in figure 5.1. The questionnaire form can be found from appendix B.

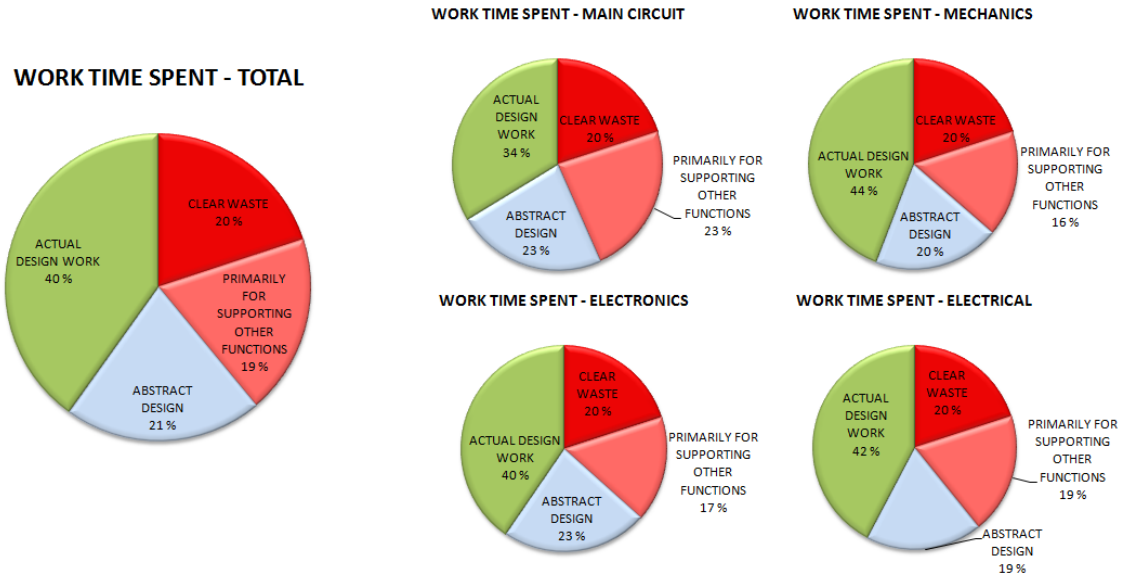


Figure 5.1 Allocated work time in PD (Kantomaa, T.T.)

The way designers allocate their time seems to be quite similar across teams. Noteworthy is perhaps the fact that main circuit designers spend the most time on activities that support other functions, and thus time spent on actual design work is the least from all of the design teams.

The list of activities done primarily to support other functions includes, among other things:

Bill of materials (BOM) and entering related material codes in ERP system, secondary sourcing activities, manuals, working on marketing/trade show models, creating rating database (RDB), inbound logistics related activities, aiding in service plans, assisting in making training material, assisting sales, assisting Product Engineering (PE), assisting service, brochures.

Including BOM in category (C) is debatable, since making BOM is an essential part of the design work, whereas many of the other activities listed above are activities done in an assistive role. Including sourcing activities in category (C) is also debatable, since most of the time spent on sourcing activities should not be stated as work done to support other functions. These activities need to be done by the designers themselves, as they require detailed technical information about the requirements of the components to be selected. These sourcing activities are termed primary sourcing activities. Nevertheless, there is some part in the activities that could be separated and done by a "component engineer". This would be someone

who would act in the boundary between sourcing and PD and aid designers in their sourcing activities. This person would need detailed technical knowledge about the components and their application. Although designers were asked to separate the share from total time spent on sourcing activities which could be performed by a component engineer, i.e. secondary sourcing activities, and include only that part in the questionnaire as "work done primarily to support other functions", some designers have clearly included sourcing activities altogether in category (C) in the questionnaire. This is why discretion is advised when contemplating on the share of secondary sourcing activities in category (C). Designers estimated that sourcing activities take a total of 9% of working time, and that 4% of the total working time is spent conducting secondary sourcing activities.

There should be a person from sourcing in every PD project who is truly interested about the progress of the project and committed to it. This component engineer has knowledge and necessary skills required to make a preliminary scan on the supplier field when designers provide specifications for different components. After making the preliminary scan on the suppliers the component engineer provides the designer with a list of preferred suppliers, from which the designer, after having correspondence with the suppliers, makes the ultimate choice. The component engineer makes the procurement initiatives and also follows them through.

Designers were asked to specify which activities done to support other functions take the most time by making a list of 5 most time consuming activities. The first one on the list is given 5 points and the last one 1 point. The results state that the configurable BOM and the SAP code world behind it (46 points), sourcing activities (20 points), manuals (14 points), making marketing/trade show models (13 points) and supporting sales (12 points) are the most time consuming activities. The three first mentioned are quite consistent for every team. Only main circuit designers and mechanical designers spend notable time in making marketing/trade show models and assisting sales, according to the respondents. A lot of main circuit designer time is also consumed by creating a RDB (12 points).

Activities done to support other functions should be synchronized correctly with development, so that the right information is at the right place at the right time. This reduces the need for doing the same things over and over again, i.e. reduces futile iterations.

Along with other design activities, it is activities done to support other functions that count for multitasking; it's not about how many projects you are on, designers are multitasking already within one project. There are occasions where a designer has had a presumption that his day would start with actual design activities, when other things suddenly require attention. May they be assisting sales by making a design sizing case or having to provide the manual team with design information, but nevertheless after having handled these issues it is late in the afternoon when the actual design activities can be started.

Finally main circuit design is contemplated here further because it has the largest share of activities done to support other functions. Main circuit designers spent on average only 34% of their time doing actual design work and 23% in supporting other functions. Out of the activities done to primarily support other functions making BOM and entering related material codes to ERP system is the most time consuming, which is followed by generating a RDB, assisting marketing and sales, and conducting sourcing activities (primary and secondary).

Main circuit designers also estimated that on average 23% of the work categorized as actual design work could be performed by someone else, rather than a competent designer. These tasks include mainly

manual, repetitive work. By letting someone else do them the designers could focus more on the actual design work. Multitasking would also be diminished this way.

These percentages imply that the absolute time spent by main circuit designers on actual design work could be increased greatly either by increasing the proportion of the actual design work compared to other categories, or by streamlining the actual design work by concentrating all manual, repetitive work to a handyman recruited in that design team.

As a final note on this section it must be stated that the designers in the PD environment are already working as hard as they can. Designers have their hands filled with work and are struggling to get everything done as it is. The total time spent by designers on work related issues cannot be increased, but it can be streamlined.

5.2.2 Modularity in product development

Modularity in PD means completely different thing than it does in production, service, or sales. In these areas modularity means how quickly and easily *products* are assembled (just click-click), serviced, or customized. In product development modularity can be achieved through platform approach and by design flexibility. The degree of design flexibility is determined by how changes in one part of the *design* affect the total *design*, i.e., how modular or integrated the *design* is. The design can be modular to full extent, it can be integrated to full extend, or in reality it is something between these two extremes. Modularity can also mean knowledge modules, i.e. the captured design intent behind the design.

Platform approach is used in the company's product development. This has been extended in recent years to cover also software development.

The flexibility of the design can be divided into three dimensions: mechanical, electrical and thermal. Spacious design increases flexibility for making changes to the design and also helps assembly and installation. Trying to avoid restricting electrical component choices and using extra capacity in some components increase flexibility in the design. Not dimensioning components to the verge of thermal limits also increase flexibility. These dimensions are in many cases interdependent; change in one of them affects the others. For example, mechanical dimensioning affects cooling of components, which further affects the performance of electrical components. Figure 5.2 depicts the flexibility of a design. The area of the triangle represents the degree of flexibility.

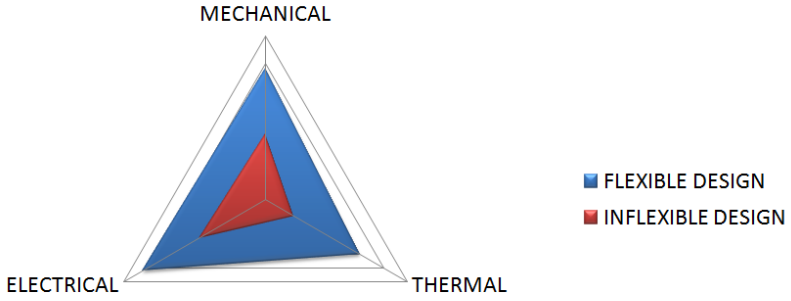


Figure 5.2 Flexibility of a design (Kantomaa, T.T.)

Increasing design flexibility might be useful during the PD project for being able to adapt to design changes more easily or it might become useful when making later changes to the product when it has already been introduced to markets, but increasing flexibility in design would raise the cost of the end product.

5.2.3 Workload leveling

Resource consumption is not planned on an accurate level and on a cross-project basis. The unpredictable nature of task durations, and hence, project schedules makes it difficult to plan resource consumption on a detailed level for long periods ahead. Planning the utilization of testing facilities also has deficiencies.

Although resource consumption is not planned on a detailed level on a cross-project basis, idle time for designers is avoided through assigning designers to more than one project or sub-project at a time. This way if there exists a waiting period in the primary project or sub-project there is always the secondary project to work on. However, resource pikes are not leveled in any way. Resource pikes mean occasions when the designer must work on more than one project on a day to day basis because of simultaneous urgencies within different projects.

5.2.4 Concurrent engineering (CE)

CE is applied, and orchestrated through a Stage-gate® model in the company. The Stage-gate® model synchronizes different functions in the product development process.

Concurrent engineering encourages iterative development in the beginning of projects to avoid harmful iterations later on. Assembly friendliness of products is frequently inspected in meetings between product development and production, and these meetings are held already from the beginning of PD projects. Sourcing is involved from early on as well. Some say sourcing is involved even too early and too enthusiastically, since sourcing activities can slow down development activities. The procurement initiatives for ordering customized parts for prototypes go through sourcing department. Designers would want these components to be delivered as fast as possible from nearby. Sourcing wants to reach economies of scale and incorporate suppliers which are able to supply globally and with volume, and with a cheaper price than nearby suppliers. In many cases low cost countries (LCC) are emphasized, which naturally leads to longer delivery times. Compromises between development and sourcing department are in many occasions needed when balancing between speed of component deliveries and making deliveries from LLC countries.

Stage-gate® model is used to synchronize the development efforts *within* a project. There have been some deficiencies in synchronizing development *between* projects in recent years. For projects that have interdependencies, this issue needs scrutinizing.

5.2.5 Cross project knowledge transfer/Amplify learning

There is much improvement to be done in this area. Profit centers are unaware of other profit centers' past project experiences and of their own profit center's past projects, especially if they date back several years. The inevitable consequence is that the same mistakes are made again and again and the solutions to these problems are invented again and again. The fault lays both in the way projects are documented and in the way these documents are utilized; both need to be scrutinized.

It has been said by the project managers that they do know what happens in other profit centers' projects as long as they have personal contacts there. On these occasions project managers exchange information with other project managers or designers located at a different profit center through informal conversations. As soon as these contacts cease to exist the connection is lost.

5.2.6 Process standardization

Instead of physical modules, modularity can also mean knowledge modules. Through the reuse of knowledge modules efficiency and increased NPD speed can be achieved. The basic idea is to capture and re-use the *engineering intent* during a product development process. There is ongoing development in capturing design information in different design areas through process standardization and building libraries of design information. The aim is to create process standards and databases that would condition designing and capture the design information. When capturing design information, besides capturing the actual design, it is equally important to capture the engineering intent behind the design; why it was designed this way.

5.2.7 Deliver as fast as possible

The development projects under examination develop products having long life cycles. This means the quality of the product outweighs the speed of development. Nevertheless, short time-to-market could provide a competitive advantage. Since the products have long life cycles, risk taking is minimized. This means no unverified technology is to be used in the products and risks are mitigated. This master's thesis is basically about using product development resources more efficiently. When resources can be used more efficiently, project lead times are decreased and new projects can be started more frequently.

5.2.8 Empower the team

There is a conception that the developers of products are not empowered, or listened enough when it comes to making decisions in product development projects. The belief is that money rules, and this belief is being shared by many designers, especially amongst seasoned designers. It is said that many technically capable solutions are rejected or not developed to full extend if the monetary benefits cannot be stated clearly from the very beginning of a possible development initiative. On the other hand, outside of product development the mood is that PD actually has too much power in decision making. This is a prime example of how different departments perceive things differently in organizations.

5.2.9 Build integrity in

A system is perceived to have integrity when a user thinks – "Yes" That is exactly what I want". The markets and the preferences of the customers are said to be well known. Still some designers say they are missing information on how detailed technical solutions are perceived by end customers. For example, electrical designers state that they do their design the way they have done it in the past without knowing if it is the right way of doing things; they don't receive feedback from end users so they don't know if what they have designed is appreciated by them and whether it should be done differently.

6 ITERATION IN PRODUCT DEVELOPMENT

6.1 Product development project types in the company

Platform projects create common platforms (panels, software, fieldbus modules etc.). Base projects, which can be shared between profit centers, create end products, modules for the end products, or in some cases both; a module can be an end product and a building block to a cabinet-built product. Modules can also be sold to external system integrators (SI's). Cabinet projects (Variant projects) utilize platforms and modules to generate products. Order-based engineering can be done to customize the products according to customer preferences.

Projects, that create end products, have different starting points according to the type of project and according to whether a technology project has preceded the PD project. Basically there are 4 different kinds of projects for creating end products:

- next generation project
- new technology or new application area project
- quality improvement or cost reduction project
- continuous improvement project

Next generation projects may have a technology project that has preceded them. In this technology project new technologies and new partial concepts that are to be used in the PD project should be tested and verified. There are bad experiences about technology projects where concepts have been created, but nothing has been tested. This undoubtedly leaves technological uncertainties for the PD project. The purpose of the technology project should be to verify the technologies used in the PD project. In many instances the prerequisite for the technology project to be successful is that main parts of specifications and application areas are clear before the project commences. However, the technology project should not perhaps go too deep into the applicability issues; rather this should be left for the PD project where the experiences of the application areas reside. There are experiences from past projects where technology department's involvement has gone too far resulting in unprofessional solutions.

With products that have radical new technology and/or new application area PD projects must be preceded by a strong technological ground work. It is said that in these technology projects much time should be devoted to evolutionary development using trial-and-error methods; the developers should be given enough freedom to try out different kinds of things, without locking in precise solutions too early in the process. It has also been said that more innovative solutions should be formulated in these projects. The feeling is that even if the developers strongly believe new kind of technological solution would have great benefits technologically and in monetary terms, the development effort can be buried if the developers cannot clearly state already from the beginning of development the monetary benefits of the new technological solution.

Quality improvement or cost reduction projects begin from a totally different starting point compared to the other previous two project types. The main physical design already exists, and the constraining factors are prevalent and defined by the flexibility of the design. The product cannot be changed much if the design is inflexible. Only slight modifications can be done, or if really big quality issues arise, then the situation becomes troublesome. Product Engineering (PE) can do slight modifications to existing products,

but when large scale changes are needed the competence of PE is no longer enough, and it is PD that needs to start a project concerning the matter at hand.

Continuous improvement projects use project wide iteration cycles to refine the design of a product family, on a continuous basis. The groundwork for this kind of development is laid by a mother project, where the basic solutions are decided, but the detailed design for each product in that product family is left undone. When a specific customer order arrives (larger customer order or a customer order potentially leading to larger orders in the future), which cannot be fulfilled with existing products from earlier daughter projects, a new daughter project is launched in which the basic solutions laid by the mother project accompanied with customized detailed design are used to create a product. During these daughter projects more optimal design solutions may be found, compared to previous daughter projects of that product family. This way each project represents an iterative development cycle, leading to more and more optimal solutions. These cycles may result in changes to previously designed products as they are ordered again. In some occasions a customer may want to order exactly the same kind of product it had previously ordered. This results in generating alternatives of the same product; the old design for the customer who has previously ordered the same kind of products and the new design for new customer orders. This kind of project wide iterative development gives flexibility to development; design solutions may be changed later on and new ideas can be executed.

6.2 From specifications to a product

Requirements include descriptions of customer needs and in some cases elaboration about main features, performance levels, constraints on the system's operation, regulations, standards etc. Specification can be thought as a collection of requirements to be implemented. Specifications also include the application areas of the product, power ratings, operating conditions, as well as cost and size targets, usually compared to previous generation product. For example, cost reduction target can be set to -40%, and size target can be set to be less or equal compared to the previous generation product. Each profit center is responsible for bringing up their requirements for a PD project if the output of that project is to be utilized across different profit centers. Specifications can also specify some sub-concepts like I/O connections and option modules. Figure 6.1 illustrates some of the content in specifications.

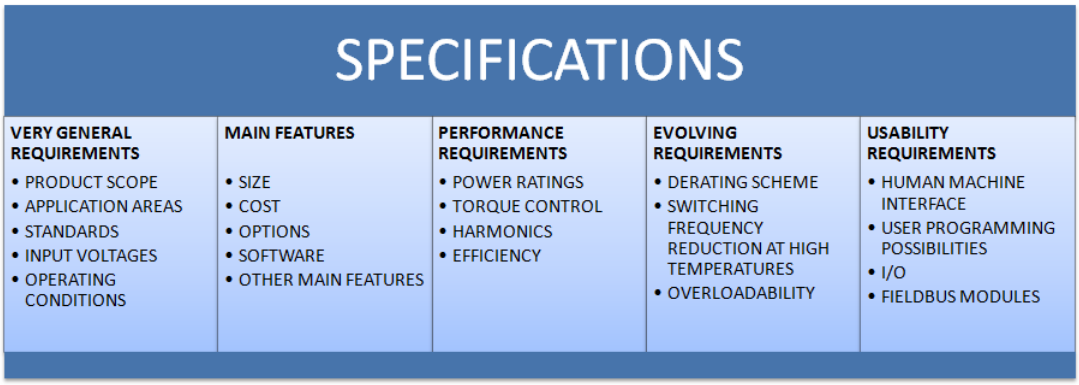


Figure 6.1 Specifications (Kantomaa, T.T.)

Technical requirements describe the wanted features to be implemented on a more technical level than specifications. Technical requirements are meant to describe how the features are meant to be implemented whereas specification mostly answers what is to be implemented. Iterating between specifications and technical requirements leads to different sets of possible technical concepts. Product management, project management and designers are all involved in this iterative process where the applicability of initial set of requirements, or *preliminary specification,* is evaluated on a technical level taking into consideration monetary issues. This process transforms *preliminary specification* into *final specification,* which is meant to be realized by the chosen technical concept. When technical concept is nearing finalization *design specifications* for different design areas should be made. *Design specifications* elaborate on technical requirements taking into consideration the entity and enable the detailed design process to commence. The technical concept shapes into its final form during the detailed design process. The earlier the technical concept shapes into its final form the less iteration there will be in the detailed design process. Figure 6.2 depicts this process.

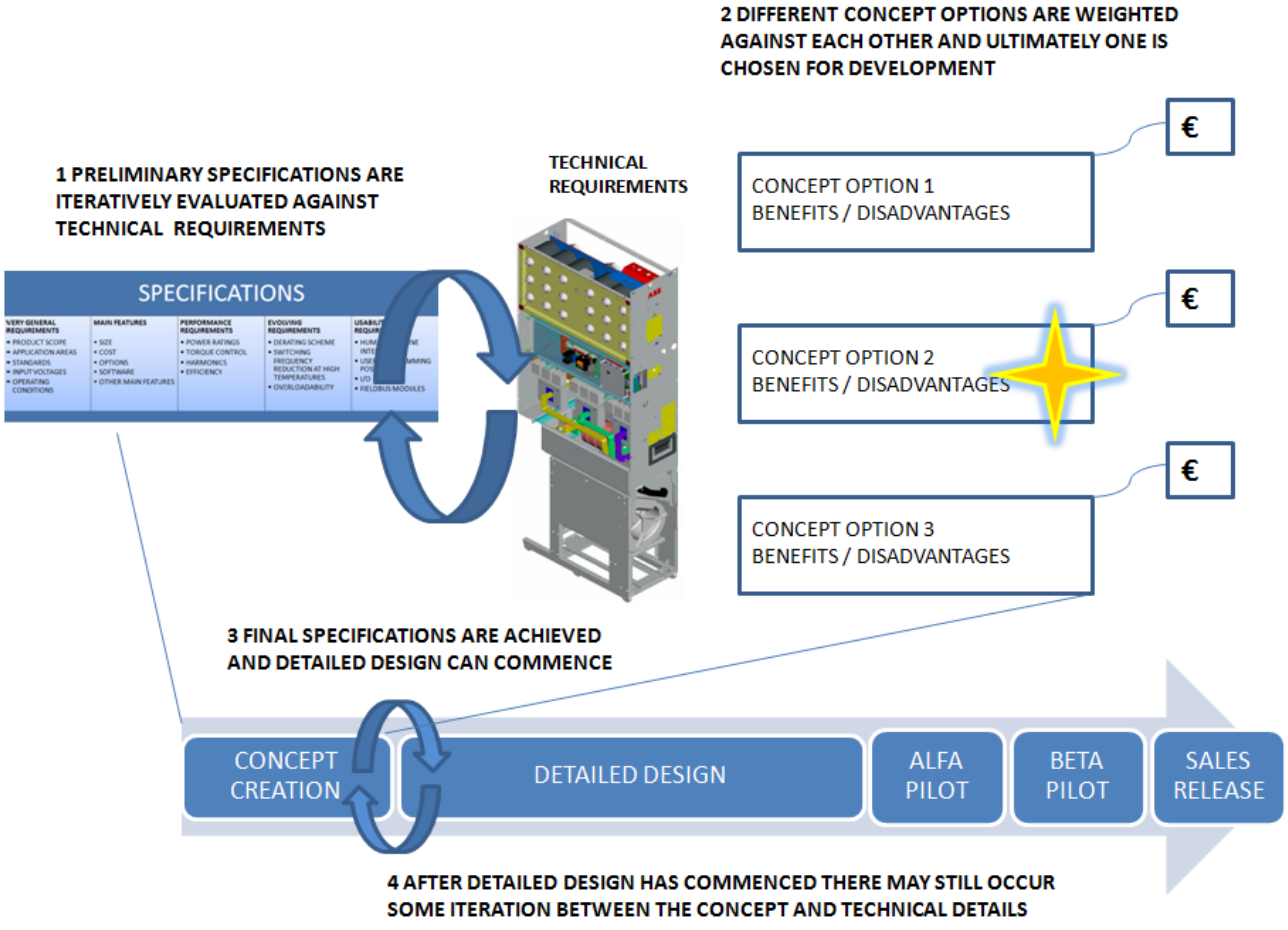


Figure 6.2 Iteration between technical concept and specifications (Hautakorpi, E., Kantomaa, T.T.)

Technical concept is a construct of sub-concepts of which some are defined already before a New Product Development (NPD) project commences in a technology project. The technology project can, for example, define the mechanical concept and the main circuit components, although these can be changed in the NPD project if a need arises. Sub-concepts include, for example;

- main circuit design
- mechanical concept
- I/O
- choke concept (AC/DC, air cooled/liquid cooled)
- auxiliary power supply
- charging concept
- IP-class
- control (SW and PCBA's)
- interface between main circuit and control board
- option modules
- panels

6.3 Detailed design process

Detailed design process, including prototyping, is the phase where most of the iteration occurs during PD projects. Many designers state, that it seems they do their designing many times over and over again during projects. This iteration, though, is mainly natural and cannot be avoided. Figure 6.3 depicts a sketch of the detailed design process.

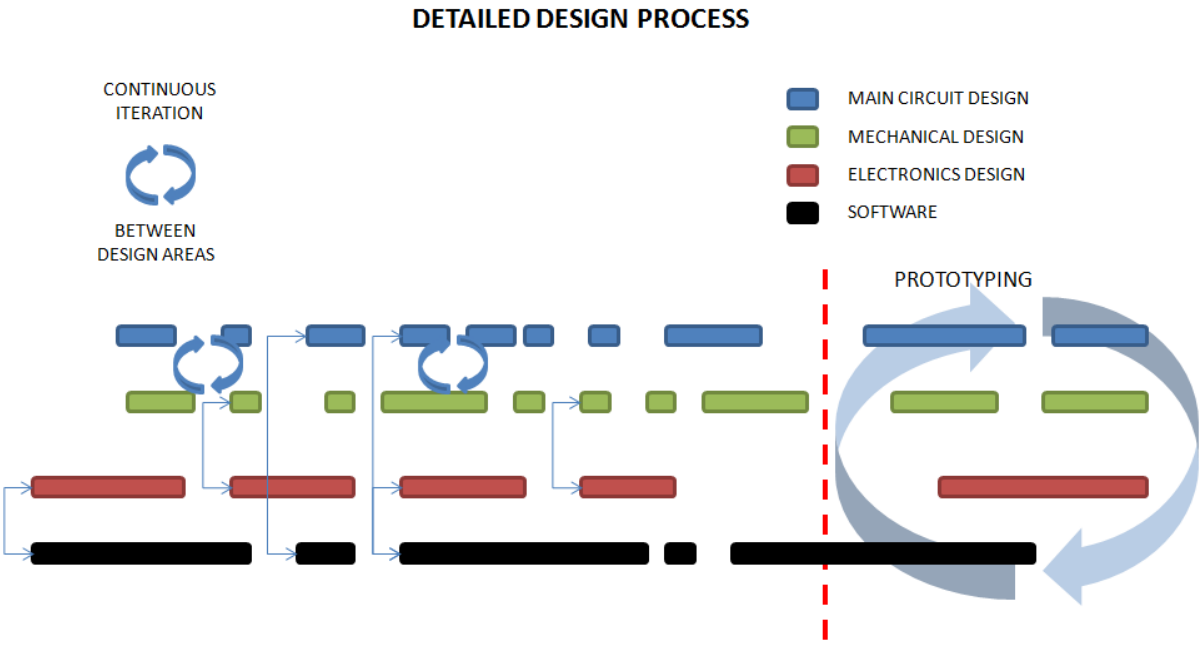


Figure 6.3 Detailed design process (Kantomaa, T.T.)

The activities in the design process can be sequenced based on their input and output requirements such that the need to send information backwards or upstream in the process and the scope of these feedbacks are minimized. Processes not organized on this basis are more likely to experience unintentional iteration simply because the right information is not available at the right place at the right time.[6] [73] [103] [104] Therefore,

[103] Browning, T.R., Modeling and analyzing cost, schedule, and performance in complex system product development, Ph.D. Thesis (TMP), Massachusetts Institute of Technology, Cambridge, MA, 1998.

activity sequencing quality not only affects the number of unintentional iterations, it also affects the number and scope of activities comprising any iterative cycle.

Also in many instances the sequence between design activities defines resource consumption between design areas. Think about main circuit and mechanical design in module projects, for example. If mechanical design and thermal design are done accurately from the very beginning of the design process, it will help the convergence of main circuit design. On the other hand, if main circuit design were let to evolve further before mechanical design would be made accurately this would diminish mechanical resource consumption, but consume more main circuit design resources, since main circuit design would be based on less accurate information in the beginning of the design process. If different design teams can transfer workloads between them their workloads are said to be "negatively correlated". The above example of main circuit and mechanical design can be augmented, for example, by mechanical and electronics designing. These design areas can also exchange workloads between them; will adjustments be made to circuit board dimensions or mechanical dimensions in case of conflicts. Usually mechanical design is changed since mechanical design can accommodate changes more easily than circuit boards can.

It is difficult to state how much streamlining there could be in the detailed design process. The detailed design process has not been analyzed deeply in the company so there might be a lot of streamlining possibilities. Mapping several PD processes from different PD projects might reveal that a standard process cannot even be mapped; every project has a unique way of proceeding so an effort to map out one "standard process" could be impossible. The standard activities in the process do not occur in the same order in every project so even those activities that are fundamental to every PD project have different order of appearance in the process. Nevertheless, conditioning fundamental sub-processes *within* and *between* design areas and using these procedures in every process, even though the process otherwise is unique, might streamline the design process. Conditioning would also result in not having to invent the wheel over and over again by gathering design information and by making design guidelines.

To be able to analyze the detailed design process on an accurate level the detailed design process should, first of all, be mapped and described. This was realized to take a huge amount of time and as such, was out of the reach of this thesis. Another approach was then chosen; system dynamics.

[104] Smith, R.P. and Eppinger, S.D., Deciding between sequential and parallel tasks in engineering design, *Concurrent Engineering Res Appl 6 (1998),* 15-25

6.4 System dynamics

Many process improvement activities require an overview of the design process. However, while individuals may have a deep understanding of the aspects of the process in which they participate, companies often have few generalists who can claim to understand the process in its entirety.[105]

System Dynamics (SD) was pioneered by Forrester in the early 1960s when he applied knowledge about systems from his work in electrical engineering to everyday kinds of systems[106]. When the structure and behavior of the design process is neither static nor well understood system dynamics modeling may support designers and design teams in improving their understanding of process behavior. Modeling, analyzing, and improving the control of dynamic systems is the objective of applying system dynamics in many domains.

Task network model, such as a DSM model, captures the architecture of a specific engineering design process. Table 6.1 summarizes the differences between task network models and system dynamics model.

Table 6.1 Differences between task network models and system dynamics model (Kantomaa, T.T.)

	TASK NETWORK MODEL (E.G. DSM)	SYSTEM DYNAMICS MODEL
TYPICAL PURPOSES	• Design process analysis • Activity sequencing	• Process behavioural analysis • Iteration management policy definition
KEY VARIABLES/ ATTRIBUTES	• Network structure	• Rework cycle • Feedback loops • Process environment • Process control mechanisms
STRENGTHS	• Capturing process architecture • Process scheduling	• Process behavioural analysis • Modeling influence of rework cycle and dynamic feedback structure on process behaviour
WEAKNESSES	• Requires extensive data volume • Little traceability of process behavior	• Little reference on actual process architecture

Knowledge about the process architecture ensures that the impact of iteration on the performance of a specific process is taken into consideration. The feedback structure of causes and effects, and its impact on process behavior can be better explored using a high-level modeling approach, such as System Dynamics[107]. In the SD model typical influencing variables and feedback structures that make up the process environment and process control mechanisms are incorporated. Together process architecture, process environment and process control mechanisms exert strong influence on process behavior. Figure 6.4 depicts model transformation.

[105] Eckert, C. M. and Clarkson, P. J., 2003 "The Reality of Design Process Planning", *ICED'03 Stockholm*

[106] Sterman, J. D., 2000. "Business Dynamics: Systems Thinking and Modeling for a Complex World." McGraw-Hill, Boston, Massachusetts, USA.

[107] LE, H.N., WYNN, D.C. and CLARKSON, P.J. (2010) "Evaluating the impacts of iteration on PD processes by transforming task network models into system dynamics models" in ASME 2010 International Design Engineering Technical Conference & Computers and Information in Engineering Conference (IDETC/CIE 2010), Montreal, Quebec, Canada

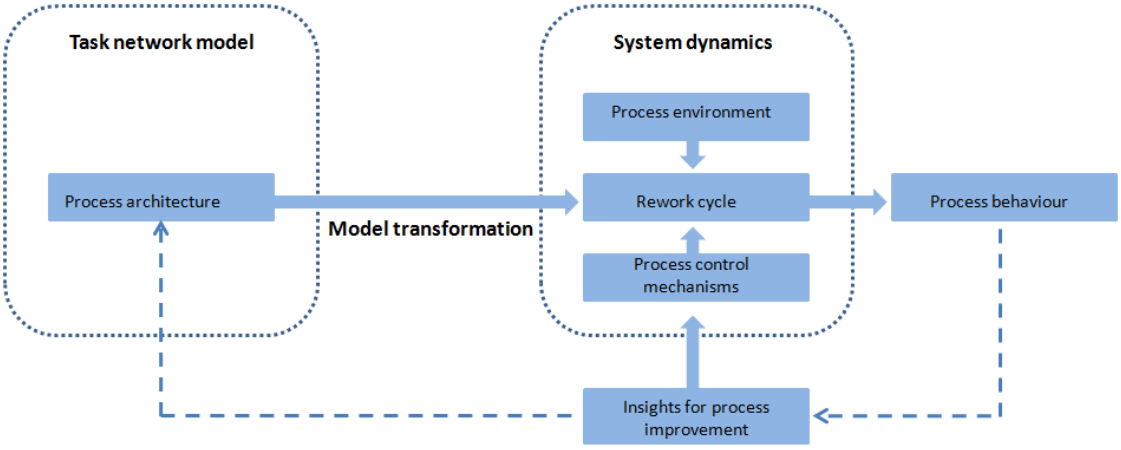


Figure 6.4 Model transformation (Kantomaa, T.T.)

REWORK CYCLE

The rework cycle's recursive nature in which rework generates more rework that generates more rework etc., creates problematic behaviors that often stretch out over most of a project's duration and are the source of many project management challenges. In an SD model, a PD process can be represented by work packages which flow through a standard "rework cycle". The rework cycle, depicted in figure 6.5 operates as follows. At the beginning, all work packages are in the stock "Original Work to Do". They are processed at a given rate through "Work in Progress" until they reach "Work Done", where all work packages need to arrive to complete the process.[108]

However, some work will contain errors which need to be *corrected*. Also, some work may need to be done again, but at a different level of abstraction, i.e. they need to be *repeated*. Work that needs to be repeated is the result of arriving *new information* during development. The source of new information can be

1) internal, for example, design-build-test cycle may reveal new information
2) external, for example, changing customer demands can result in specification changes

Process concurrency increases repeated work since less work is started with finalized information, and changes in design information needs to be incorporated with iteration.

The delay in discovering the need for iteration affects the impact of the iteration in respect to resource consumption and development lead time. The structure of the design process, the process environment (organizational and human related factors) and the process control mechanisms dictate the amount of rework and repeated work.

Once work to be corrected or repeated has been discovered, this work goes from "Undiscovered work to be corrected/ repeated" to "Work to be corrected/ repeated". The "Rework/repetition rate" defines how much of this work will proceed to "Work in Progress". The rest is left as "Quality issues".

[108] Lyneis J.M., Ford D.N. (2007) "System dynamics applied to project management: A survey, assessment, and directions for future research." *System Dynamics Review* 23(2-3): 157-189

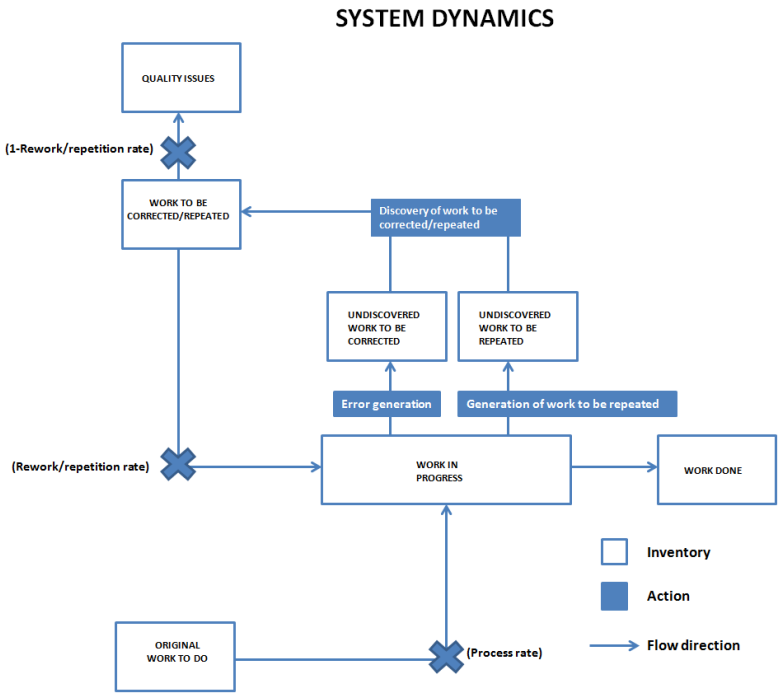


Figure 6.5 Rework cycle (Kantomaa, T.T.)

When the rework cycle is complemented with different influencing factors various feedback loops begin to appear. When the most obvious effects and causalities are marked to the rework cycle we get the frame of the SD model which is depicted in figure 6.6.

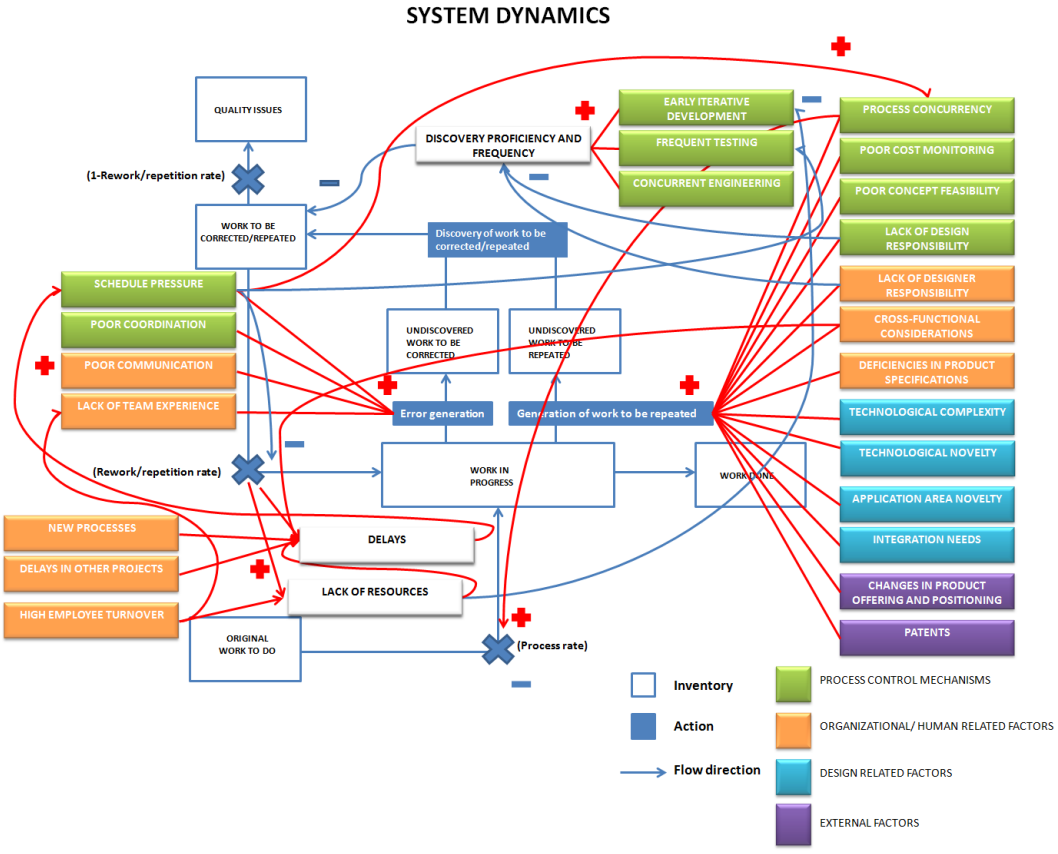


Figure 6.6 Frame of the SD model in the company (Kantomaa, T.T.)

6.4.1 Organizational/ human related factors

Organizational/ human related factors are the result of higher level organizational politics or the organizational culture and procedures (the way of doing things).

"Designer responsibility" refers to designers having responsibility over their design areas integration to the design entity and doing follow-up. Currently this is up to the designer; some designers are more concerned with the integration of the design than others. Designers should also think about the applicability of their design; where is it utilized. This is especially important with cross-profit center designs.

"Other projects" naturally affect individual projects when they have interdependencies. "New processes" create delays as it takes time to get accustomed to the new processes/tools. "Communication" refers to the natural tendency of designers to communicate with each other in the product development environment. If the natural tendency between designers is to communicate proficiently, then less coordination from behalf of project management is needed and vice versa. "Lack of team experience" results in more design errors. "High employee turnover" can result in "Lack of team experience" and in lack of resources (especially efficient resources). "Product specifications" refers to the overall specifications of the product to be developed. "Cross-functional considerations" refers to other functions influence on product development. The more influence other functions have on product development the more repetitions and delays there will be in development.

6.4.2 Process control mechanisms

The management of design processes can also be a cause of iteration. Under "Schedule pressure" managers often rely on excessive "Process concurrency" based on early release of information[109]. As a consequence, tasks need to be repeated when input information is finalized and the amount of rework may vary according to the chosen degree of concurrency[110] [111]. Rework is also likely to occur when deadlines are set in a way that available time is not adequate to workloads, causing high work intensity and overtime for designers, and resulting in higher chances of flawed work[108]. Lack of commitment to schedules will also occur in situations where the deadlines are made impossible to reach. Another human related aspect is the nature of concealing problems and need for rework under schedule pressure, causing even more rework to be required later in the design process[112].

System level design defines the architecture of a product. System level definition is a challenging task and requires expertise on multiple design areas. In system level designing the outputs of different projects (platform projects, module projects from different profit centers), product options and different sub-concepts are merged into one functional entity. Understanding the system architecture and interdependencies between functionalities and product options is very challenging. System level designing also extends to inter-project level. There are cross-profit center projects which utilize each other outputs, and in these occasions the receiving project should be able to clearly define its technical requirements, which together with other requirements comprise the design specifications for the feeding project.

---

[109] Joglekar N.R., Yassine A.A., Eppinger S.D., Whitney D.E. (2001) "Performance of coupled product development activities with a deadline." *Management Science* 47(12): 1605-1620

[110] Krishnan V., Eppinger S.D., Whitney D.E. (1997) "A model-based framework to overlap product development activities." *Management Science* 43(4): 437-451

[111] Roemer T.A., Ahmadi R., Wang R.H. (2000) "Time-cost trade-offs in overlapped product development." *Operations Research* 48(6): 858-865

[112] Ford D.N., Sterman J.D. (2003) "The liar's club: Concealing rework in concurrent development." *Concurrent Engineering-Research and Applications* 11(3): 211-219

Principal designer ("design responsible") is involved in system level designing and in determining design specifications for design areas. Requirements specification doesn't show what should be done on detailed technical level and one of the main tasks of the principal designer is to elaborate the technical requirements to the designers for the detailed design process to commence.

Concept development phase locks-in most of the costs of the end product under development. It has been said that about 80% of the costs are decided in the concept development phase which makes it very important to make as accurate initial cost structure calculations as possible. But it is in the detailed design phase that the critical cost line is either passed underneath or exceeded. That is why "Cost monitoring" should be done on a continuous basis. "Concurrent Engineering (CE)" should be applied to facilitate short-cycled iteration with respect to cross-functional aspects (production, service, product engineering, sourcing). "Early iterative development" aims at discovering errors and the need for redesigns early in the design process, trying to avoid situations where the need for redesign is realized in later stages of the project resulting in large iterations, which lead to schedule and cost overruns. Simulation, concept testing and concept prototyping are constituent parts of early iterative development. "Testing" refers to component testing and testing partial entities of the design entity. It is important to test partial designs in advance of integration. "Prototyping" refers to testing sub-concepts or the whole design entity, and it can also include gathering customer feedback on the prototype. "Concept feasibility" refers to the applicability of the technological concept. The more proficient the early iterative development is the sooner the concept is revised to its final form. The less spontaneous communication there is between designers within and between profit centers the more "Coordination" is needed in behalf of project management.

PROCESS CONCURRENCY

Process concurrency and Concurrent engineering (CE) means two different concepts in this thesis. CE means cross-functional development where cross-functional considerations are taken into account with short-cycled iteration to increase NPD speed. Process concurrency, on the other hand, refers to overlapping design activities. Excess process concurrency increases work to be repeated as more work is begun without finalized information.

CORRUPTION

Rework due to corruption refers to rework or repetition of downstream tasks when a change in an upstream task occurs, when the downstream task otherwise would not have needed an iteration. In other words, some tasks need to be reworked because they start on incorrect or not finalized information from upstream tasks.[113] Process concurrency increases corruption.

Process concurrency on task level: Overlapping development activities may save time, but it may also increase iterations or reworking for the downstream activities and increase resource usage. When preliminary upstream information is utilized by the downstream activity too early, then the future changes have to be incorporated in time consuming iterations, resulting in an increase in the downstream activity duration and resource consumption. If the degree of information evolution of an upstream activity is thought to be slow and iteration sensitivity of a downstream activity high then these activities should not be overlapped too much to avoid futile iterations. Special cases exist, though. If the activities are such that

[113] J. Lin, K. H. Chai, Y. S. Wong, and A. C. Brombacher, "A dynamic model for managing overlapped iterative product development," European Journal of Operational Research, vol. 185, no. 1, pp. 378–392, 2008

one design area is not aware of other design areas capabilities or constraining factors, then these activities need to be done simultaneously and by exchanging information continuously.

Process concurrency on project level: When a NPD project commences it is usually preferred to have platforms ready, and the same applies to other feeding projects as well. On the other hand, overlapping interdependent product development projects can speed up the development. Overlapping interdependent product development projects can also be useful because a downstream project can give feedback to the upstream project while it is still in progress. Modifications can be made in the upstream project which will aid development/integration in the downstream project and thus, flexibility is increased.

When projects are overlapped excessively a delay in an upstream project leads to delays in every downstream project. Delays can also lead to iterations.

6.4.3 Design related factors

"Technological complexity", "Technological novelty", "Application area novelty" and "Integration needs" all influence the amount of iteration needed during the product development project. Technological complexity results in more iteration as tasks must be revisited to achieve a technical solution. Technological novelty refers to new kind of technology being applied, whereas application area novelty refers to existing technology applied to new application areas. Similarly, iteration is likely to occur when interface or system integration issues are considered because new information may surface or it may become clear that design objectives are not met when testing is performed.

6.4.4 External factors

Besides of the above mentioned categories there are external factors affecting development. One of the most harmful causes for iterations in product development projects is changes in product offering and positioning. Changes in product offering and positioning are caused by changing customer demands, competitor actions or upper management decisions.

In HPD competitor benchmarking is utilized to some extent in main circuit design and mechanical design. In electrical design competitor benchmarking is not done. Although main circuit design and mechanical design utilizes competitor benchmarking it is said this area still needs highlighting. At some point in the development process there may be input, for example, from product management about competitor's products and this can lead to iteration. Figure 6.7 depicts three levels of competitor benchmarking.

Figure 6.7 Three levels of competitor benchmarking (Kantomaa, T.T.)

Competitor benchmarking can be done in three levels. In the first level the competitor's offering and the main features of their products are known on paper. There is a new portal for easy monitoring of competitors offerings in the company (Competitor Comparison Portal). In the second level the main technicalities concerning competitor's products are known. In the third level the competitor's products are reverse engineered and every little design detail is examined.

There is not a systematic way to benchmark competitor's products. The competitor's offerings are known, but not always on a detailed technical level, i.e. reverse engineering is not always done. Competitors on the other hand have put effort on this area; some competitors systematically buy the company's products and reverse engineer them. It has been said that it can be seen from the competitor's products that they have copied the company's products.

There has been the perception that the company is the technological leader in frequency converter business, and perhaps this has resulted in not benchmarking competitors; "we don't need to benchmark others, they benchmark us". Especially in cabinet designs the company's products are said to be way ahead of competitors. Nevertheless, this does not mean that every little design detail is done better compared to competitors, and thus, reverse engineering could provide valuable improvement insights.

Regulatory issues can also cause iterations. Patents cause iterations quite often in mechanical design, although these are mainly minor iterations. Patent checks should be carried out with proficiency to avoid futile rework due to patent related issues and patents can also be a source of new ideas.

6.4.5 Adverse dynamics

Ripple effects: Many well-intentioned project control efforts may have unintended consequences. "Ripple effects" is the name commonly used in projects to describe the primary side effects of well-intentioned project control efforts.

Knock-on effects refer to the secondary impacts of project control efforts, i.e., the impacts of ripple effects, often caused by processes that produce excessive or detrimental concurrence or human factors that amplify the negative effects via channels such as morale. Capturing knock-on effects in project models uses the concept of unintended side effects to explain project behavior and performance.

CAUSAL CHAINS AND REINFORCING LOOPS

Network of iteration causes and effects involves causal chain loops with reinforcing character. When contemplating the SD model frame in figure 6.6 different causal chains can be seen. For instance, when too much schedule pressure is exerted to the development team more errors are generated and to gain time compression process concurrency is applied. Error generation creates more work to be corrected. Process concurrency increases work to be repeated as more work is begun without finalized information. If work to be corrected/repeated is not done quality issues may arise. Corrections and repetitions (defined by Rework/repetition rate) create delays and take resources away from working with other tasks, which again leads to delays. This loop becomes self-enforcing as delays cause more schedule pressure and the loop starts again. Figure 6.8 depicts a simplified causal chain of events.

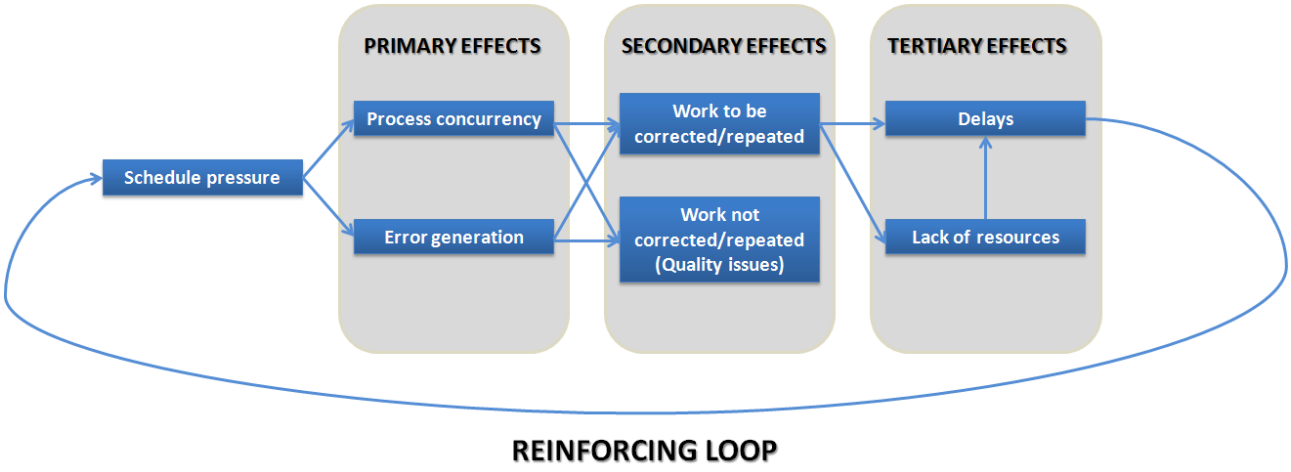


Figure 6.8 Relation between delays and iteration (Kantomaa, T.T.)

When examining factors causing iteration it might not be self-evident that delays are among them. Through System dynamics though, the relation of delays and iteration can be seen.

FROM THEORY: CONCEALING REWORK REQUIREMENTS IN CONCURRENT DEVELOPMENT[112]

The "90% syndrome" is a common concurrent development problem in which a project reaches about 90% completion according to the original schedule but then stalls, finally finishing after about twice the original project duration has elapsed. [112]

Excess process concurrency not only increases the vulnerability of projects to changes and errors requiring rework, but also increases the fraction of work released that will require changes. Increased concurrency reduces the availability of final information leading to additional rework and iteration. In addition, behavioral effects arising from the increased pressure felt by developers to meet schedules that do not account for the greater iteration and coordination required by concurrency contribute significantly to the poor performance of many such projects.

Concealing known problems can be used to temporarily reduce work. For example, an engineer in a leading electronics company reported that design engineers regularly delayed revealing problems they discovered to avoid time-consuming document control work required by the organization's engineering change notice process[114]. The practice of concealing rework requirements is reinforced by people's dislike of bad news and information that contradicts their beliefs. People in authority often "shoot the messenger."

Concealment is often standard practice. At a major defense contractor, weekly meetings of project team leaders were known as "the liars' club" because everyone withheld knowledge that their subsystem was behind schedule. Members of the liar's club hoped someone else would be forced to admit problems first, forcing the schedule to slip and letting them escape responsibility for their own tardiness. Everyone in the liar's club knew that everyone was concealing rework requirements and everyone knew that those best able to hide their problems could escape responsibility for the project failing to meet its targets. [112]

[114] Ford, D.N., Hou, A. and Seville, D. (1993). "An Exploration of Systems Product Development at Gadget Inc." *System Dynamics Group Report* D-4460, Sloan School of Management. Massachusetts Institute of Technology, Cambridge, MA.

The behavioral pattern of concealing rework requirements was not seen to occur in the company. But there are other behavioral patterns that have the same effects, i.e., the need for rework and repetitive work is realized late requiring even more resources to be consumed in the corrective and repetitive tasks.

Technical problems are brought up in the company, but some interviewees felt that discovered problems are not always resolved to full extent before proceeding further because of schedule pressure. These problems are later resolved by unconventional means, if the need arises. Other designers have not let the schedule pressure affect them. Instead, they stated they try to resolve all problems to full extent before proceeding to next stages.

Others say platform projects and the fact that things are sliced to many projects may have caused irresponsibility in problem solving. Some designers involved in platform projects are just designing partial entities and performing tasks; they are not concerned about the integration of the end product. A designer might see problems in the integration of the total design, but because it is not his responsibility he might not point it out.

LACK OF RESOURCES

When faced with the need for more resources, project management basically has four options: slip schedules, put pressure on work force to work faster, work overtime or hire more people. If delaying the project is not an option then only the latter three come to question. Figure 6.9 depicts the effects these measures can have.

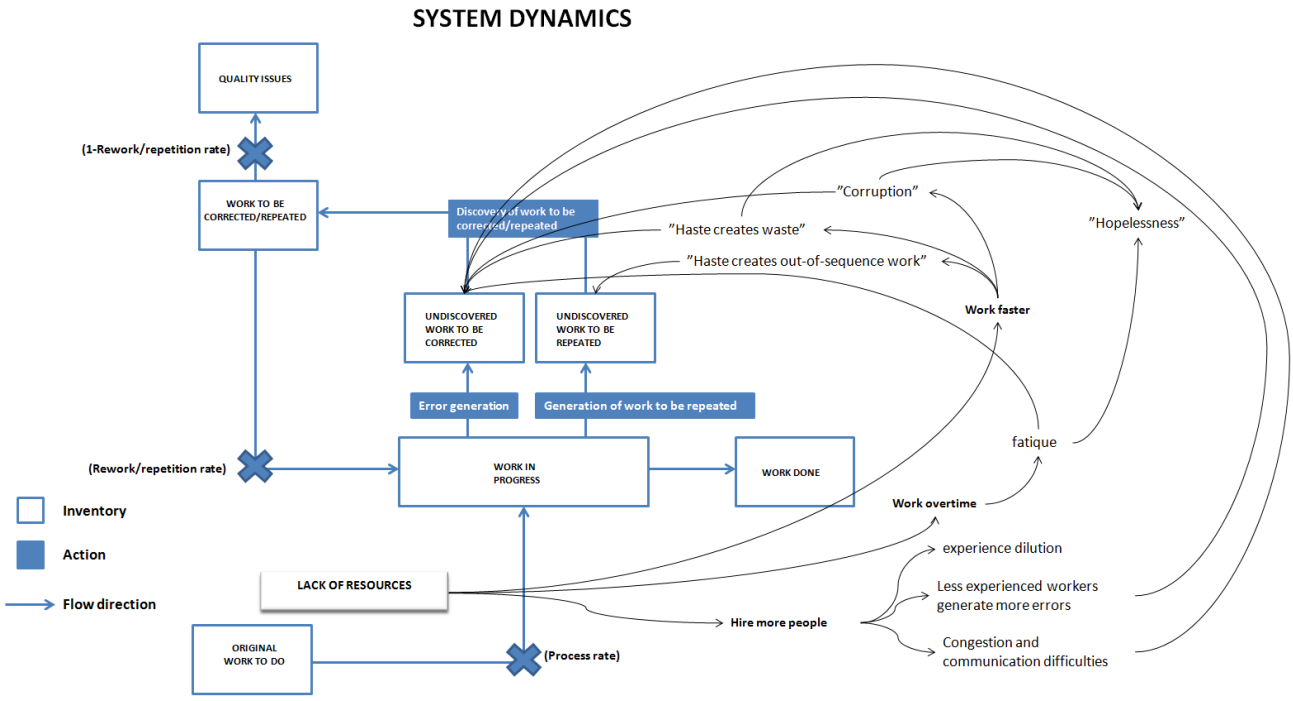


Figure 6.9 Adverse dynamics resulting from increasing resource availability (Kantomaa, T.T.)

If the adverse dynamics of these measures are not understood properly reduced productivity and increased rework can lead to the amount of work remaining greater than it would have otherwise been, thereby increasing resources needed to finish on time.

Ripple effects:

- Hiring can dilute experience as workers with less skill and/or less familiarity with the project are brought on, and because they require experienced developers to divert time to training instead of doing development.
- Less experienced workers generate more errors.
- Larger workforces can increase congestion and communication difficulties, which increase errors and decrease productivity. This is especially true with dispersed resources.
- Overtime leads to fatigue that also increases errors and decreases productivity.
- Higher work intensity increases errors. [108]

Knock-on effects:

- "Haste creates out-of-sequence work"—trying to accomplish more tasks in parallel than physical or information constraints allow, whether by adding resources or exerting schedule pressure, can cause work to be done out of the desired sequence. This reduces productivity and increases work to be repeated.
- "Corruption"—undiscovered errors in upstream tasks are inherited by downstream tasks.
- "Hopelessness"—morale problems can exacerbate the effects—fatigue and rework can create a sense of "hopelessness" that reduces productivity and also increases employee turnover. [108]

The process of correcting errors can increase the number of tasks that need to be done in order to fix the problem, or can increase the work required because fixing the errors takes more effort than doing the original work. This will happen especially if the errors are not noticed until later phases of development. [108]

From theory: staffing

Figure 6.10 illustrates typical possible behaviors for project staffing, according to theory. Planned staffing often builds up to a peak, and then gradually declines; actual staffing, however, can deviate significantly from the plan. Often the ramp-up of staff is delayed, then overshoots the planned peak and remains high longer (sometimes with a second hump). Such projects typically experience both schedule and budget overruns, and also deliver projects with reduced scope and lower quality than desired or required.

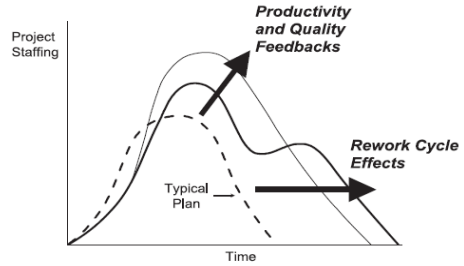


Figure 6.10 Planned and actual staffing [108]

The ramp-up of staff is often delayed either because of external conditions such as late finish of other projects, but also because management overestimates productivity and underestimates true project scope and the delays in getting resources. The "hiding" of undiscovered rework further delays the recognition of staffing needs. As the need for additional staff is recognized and controlling actions taken, ripple effects and knock-on effects on productivity and quality of managerial responses tend to increase effort required and therefore staff needs, causing project staffing and labor hours to peak higher and later than planned. Cycling of work through the rework cycle also pushes project

completion later in time.

Projects that are underestimated end up costing more because of the adverse ripple effect dynamics incurred once the underestimate is discovered; projects that are overestimated also end up costing more than they otherwise would because of the tendency to slack off and/or "gold-plate" when there is insufficient schedule pressure. [108]

Project staffing in projects is in many occasions late in the company because of delays in other projects. Also, in many instances the technology project or the early phases of a PD project suffer from lack of early iterative development, because of lack of resources or testing facilities. Lack of testing before integration has also caused some problems.

Together schedule pressure, excess process concurrency, lack of early iterative development and lack of testing, which are process control mechanisms, can lead to the causal chains depicted in figure 6.11.

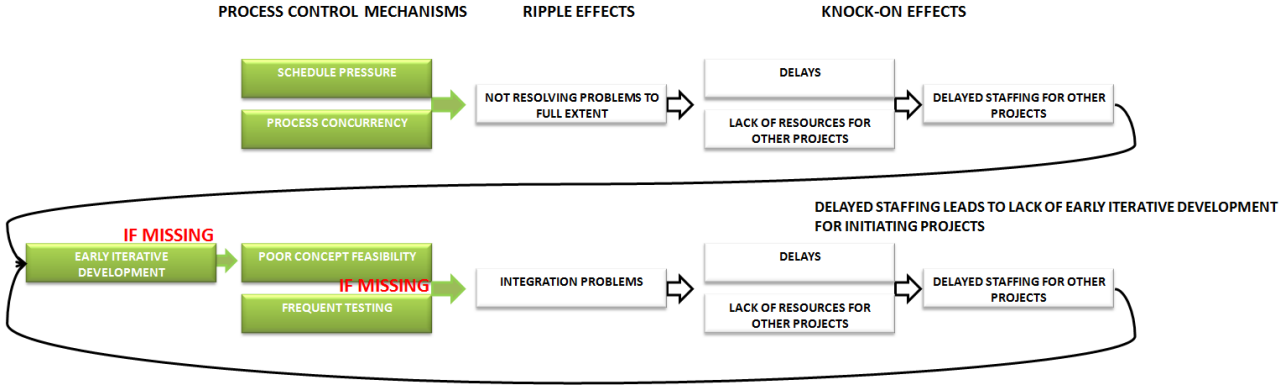


Figure 6.11 Possible causal chains of events starting from process control mechanisms (Kantomaa, T.T.)

Delays and lack of resources are caused by postponing problem solving to later phases, where they will have more severe effect on resource consumption and development lead time.

One simulation revealed the effects of time, effort and process concurrency. In this simulation it was assumed that "Overtime/Work intensity" and the resulting "Fatigue" have a negative influence on the "Progress rate" (or work productivity). SD models were then applied to execute simulation with different degrees of overlapping from 15% to 50% for two scenarios: (1) No resource constraints; and (2) with resource constraints and work intensity. Figure 6.12 illustrates these situations. [107]

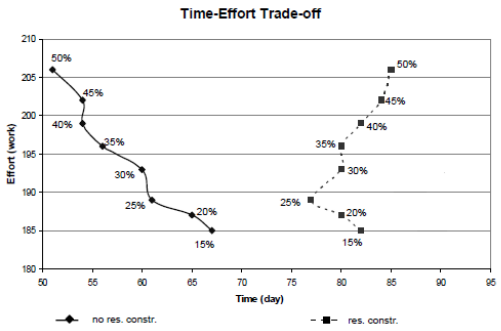


Figure 6.12 Time-Effort Trade-off [107]

Percentage numbers in the graph indicate the overlapping degrees. The most obvious observation is that the resource constraint causes longer process duration. However, more interesting is the observation that in a project setting with resource constraints a higher degree of overlapping does not imply shorter process duration. In this example, process duration is reduced through higher overlapping degree until the overlapping degree reaches 25%. From 25% onwards, a higher overlapping degree brings about the opposite effect, i.e. it causes process duration to increase. This tipping point occurs when the time saving benefit due to activity overlapping is outweighed by the negative impact of rework iteration arising from it. This negative impact results from decreasing productivity due to ever-increasing overtime/work intensity, caused by a higher overlapping degree and resulting extra rework. In reality, there are other factors that have an equally relevant contribution to work productivity, either in a positive or negative way. For managers, it is essential to understand these influences and to be able to estimate this threshold value. [107]

6.4.6 ABC-analysis

Perhaps even half of the product development resource consumption in the company is caused by iterations, mainly from harmful ones, based on the interviews.

ABC-analysis is used to divide the causes of *harmful iterations* into three classes. Harmful iteration includes:

1) iteration causing rework
2) iteration causing repetition, when the need for iteration occurs late in the process and has more severe consequences, than it would have had if the need for iteration had been realized on time
3) iteration caused by external factors (excluding the special cases of new application areas or radical new products when the customer requirements are not fully known in advance of development and customer feedback incorporating iteration is used in early phases of development)

Also factors causing delays are included in harmful iterations. System dynamics approach revealed how delays cause iteration, and this was further verified by empirical observations.

The causes of harmful iterations are classified according to their severity on resource consumption. The causes are not prioritized within classes. In class A, the causes of iteration cause severe rework in nearly all projects. Mitigating their effect would have a great influence on streamlining resource consumption. Class B iterations are also present in nearly all projects, but they cause less rework than Class A iterations. Class C represents causes of iteration that have occurred irregularly or are already under control, which means they do not cause severe rework.

In the company causes for iterations in PD projects were examined in HPD and LAC profit centers. WIND, SOLAR and platform projects were mainly excluded from this examination. Nevertheless, whenever these previously mentioned projects affect HPD and LAC projects, their effect is also included. Other profit centers do cause iterations to HPD and LAC projects since some projects are shared between several profit centers. Platform projects also cause iterations to base and variant projects. What also must be noted is that the causes of iteration are classified by examining projects executed in the 21th century. This means not just the latest or ongoing PD projects are incorporated.

The categories in figure 6.13 were formed based on interviews and corporate documentation. It must be noted that the list is an aggregation; a single actor may perceive the severity of causes of iteration differently in the PD units under examination based on his/her own experience.

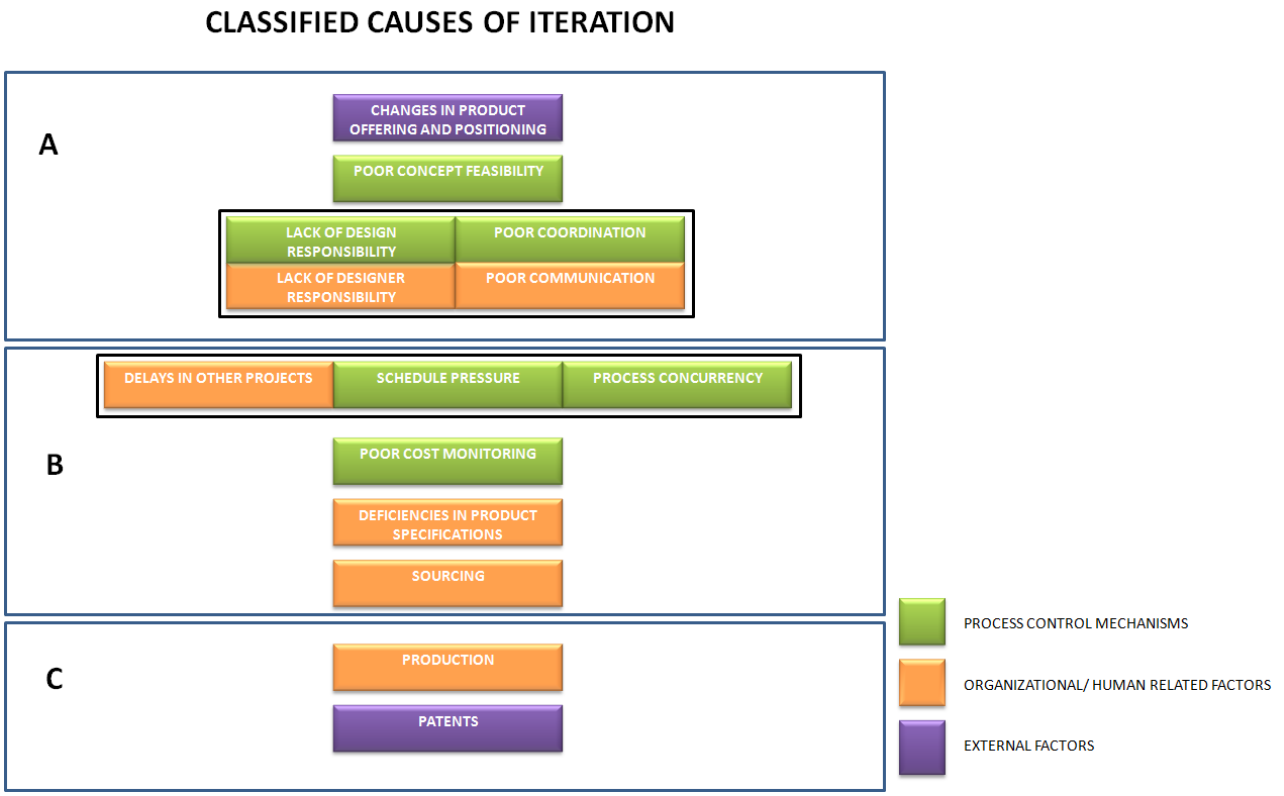


Figure 6.13 Classified causes of iteration (Kantomaa, T.T.)

"Lack of design responsibility", "Lack of designer responsibility", "Poor coordination" and "Poor communication" together results in "Deficiencies in thinking over the design entity". "Delays in other projects", "Schedule pressure" and "Process concurrency" are all related to scheduling.

The next chapter presents solutions on managing the causes of iteration.

7 MANAGING ITERATION IN PRODUCT DEVELOPMENT

The previous chapter listed, described and classified the causes for iterations in the company's product development environment. This chapter elaborates more on these issues, provides insights in how to manage process behavior taking into consideration the adverse dynamics of process control mechanisms, and a framework which aims at early iterative development for avoiding large scale iterations later on in the development process.

7.1 General insights

Causes under management control and influence could be directly mitigated if the cause-and-effect relationship is well understood. Project planners find it seductively easy to ignore the adverse dynamics created when a project falls behind and actions are taken to bring it back on schedule. Overly aggressive plan can actually make the performance of the project worse.

Actions should be taken to minimize the rework cycles consequences. Specifically, what should be done according to theory is to:

- Improve quality and reduce errors, even if those efforts reduce productivity. Doing work fewer times, even at lower productivity, is generally beneficial. One approach is to slow down and do work right the first time (i.e., reduce work intensity), even if this might cause some "slacking off".
- Recognize the existence of undiscovered rework and avoid its consequences, primarily the "errors create more errors" dynamic. Undiscovered rework can be reduced, for example, by prioritizing rework detection and correction over starting new work.
- Early testing made to discover problems rather than testing to pass tests can also reduce rework cycle consequences.
- Avoid the tendency to start downstream work too early and thereby increase unplanned concurrency and reallocate "excess" staff that will be needed later when the rework is discovered. Companies often overlap development activities to compress total lead time. This can lead to increase in the amount of "corrupted" work. [108]

Managers can significantly improve project performance through efforts to manage ripple and knock-on effects; how managers respond when the existence of an infeasible initial plan is discovered, or when changes or other risks materialize, has a significant impact on project dynamics. The best approach to getting the project back on track is not always obvious. From a dynamic systems perspective, this is particularly difficult to confirm because the strengths of the feedback loops differ across projects and are dynamic during projects. Heuristics for managing project dynamics to improve performance are difficult to create. Each project is different, so there is no specific advice such as "use x% overtime while hiring y% more staff". The following guidelines from theory should be considered with discretion:

- Ease performance targets, such as by slipping the completion or milestone deadlines, increasing the budget, reducing the scope, or accepting a higher fraction of flaws in the final product. These actions reduce ripple effects by reducing the need to change project management.
- A moderate amount of schedule pressure is optimal
- Be aware of the adverse dynamics of increasing personnel, especially less experienced personnel

- Sustained overtime does not increase productivity in the long run (longer than 3 months)[115]
- Extra time spent during requirements and design result in a higher quality project at lower cost
- Use of "experts" can significantly improve project performance
- An appropriate approach to problem solving can help to reduce iteration effort and by doing so minimize "unnecessary" iteration. Many authors suggest application of a breadth-first approach to explore the possible design space at a higher design level where alternative solutions are considered, in order to achieve a more comprehensive evaluation of feasibility and a thorough understanding of design interface issues. [107]
- In general, it is agreed that comprehensive exploration and refinement iteration at the early stage of the design process will offer "more chances for a hit" and help to cope with uncertainties and changing environment. A thorough exploration of design alternatives can help to reduce the cost and effort of requirement change implementation at later stages in the design process. [107]

7.2 How to diminish the impact of harmful iterations?

The causes of iteration revealed by the ABC-analysis can be divided into four categories: process control mechanisms, organizational/ human related factors, design related factors and external factors. Process control mechanisms are under project management control. Organizational/ human related factors are the result of higher level organizational politics or the organizational culture and procedures. Changes in the organizational culture and procedures would require much effort. In this analysis design related factors will be considered as given. The process structure should reflect the degree of technological complexity, degree of technological novelty, degree of application area novelty and integration needs of the design. Iterations caused by external factors cannot be *prevented* by process-internal means, but market uncertainties should affect the process structure.

Table 7.1 summarizes the properties and implications of these categories.

Table 7.1 Properties and implications of the four different categories causing iteration (Kantomaa, T.T.)

	PROPERTIES	IMPLICATIONS
PROCESS CONTROL MECHANISMS	ARE UNDER MANAGEMENT CONTROL.	MECHANISMS APPLY DIRECT INFLUENCE TO PROCESS BEHAVIOR.
ORGANIZATIONAL/ HUMAN RELATED FACTORS	ARE THE RESULT OF HIGHER LEVEL ORGANIZATIONAL POLITICS OR THE ORGANIZATIONAL CULTURE AND PROCEDURES.	CHANGES IN ORGANIZATIONAL POLITICS REQUIRE HIGH-LEVEL DECISIONS. CHANGES IN ORGANIZATIONAL CULTURE AND PROCEDURES REQUIRE MUCH EFFORT.
DESIGN RELATED FACTORS	TECHNOLOGICAL COMPLEXITY, TECHNOLOGICAL NOVELTY, APPLICATION AREA NOVELTY AND INTEGRATION NEEDS OF THE DESIGN ARE DESIGN RELATED FACTORS.	DESIGN RELATED FACTORS SHOULD AFFECT THE PROCESS STRUCTURE.
EXTERNAL FACTORS	CHANGES IN CUSTOMER DEMANDS, COMPETITOR ACTIONS, UPPER MANAGEMENT DECISIONS OR REGULATORY ISSUES ARE EXTERNAL FACTORS.	MARKET UNCERTAINTIES SHOULD AFFECT THE PROCESS STRUCTURE.

[115] Graham, A.K. 2000. Beyond PM101: "Lessons for managing large development programs." *Project Management Journal* 31(4): 7–18.

The classified causes of iteration are now dealt with one by one starting with the least significant. The least significant causes are not contemplated far, because their meaning is small compared to the more significant causes. External causes of iteration are omitted, unless they should have an effect on the process structure.

7.2.1 Production

ORGANIZATIONAL/ HUMAN RELATED FACTORS

There are some examples of harmful iterations caused by assembly problems discovered in late phases of a PD project. One example is from a project where the falling frame was discovered to be difficult from the assembly point of view. This lead to a harmful iteration since extra design work had to be done. Production aspects have also caused some delays by implying conformity (making sure standard screws are used etc.) on a too early stage, some say. Also, on some occasions production considerations have not been delivered on time, or the representation of production has been missing in review sessions, which have led to slight delays.

Nevertheless, in the past years, no big issues have arisen out of production considerations. Currently the assembly point of view is taken into account from the very beginning of PD projects via constant meetings where different design options are reviewed together with PD and production personnel, and through small iterative cycles the assembly problems are solved.

7.2.2 Sourcing

ORGANIZATIONAL/ HUMAN RELATED FACTORS

Sourcing has caused some iteration due trying to enforce preferred suppliers, which have not been able to deliver components with enough quality. These suppliers have then been changed to others, which have resulted in extra verification rounds. This has also caused delays, which are the more substantial effects sourcing considerations have had on development.

The procurement initiatives for ordering customized parts for prototypes go through sourcing department. Designers would want these components to be delivered as fast as possible from nearby. Sourcing wants to reach economies of scale and incorporate suppliers which are able to supply globally and with volume, and with a cheaper price than nearby suppliers. In many cases low cost countries (LCC) are emphasized, which naturally leads to longer delivery times. Compromises between development and sourcing department are in many occasions needed when balancing between speed of component deliveries and making deliveries from LCC countries.

There is ongoing development in behalf of sourcing to streamline PCBA components selection process. The goal is to build a single PCBA component library where all components get an individual ID number. This will result in synergy, since at the moment designers have their own libraries which overlap with each other. Especially newly started designers will benefit from this as the libraries are ready for them. For older designers this can slow down the PCBA design process at first when they have to get accustomed to the new process, but in the long run it will speed up the design process. Also, applying the new process should diminish the amount of different PCBA components in use and improve the quality assurance of the PCBA's.

Nevertheless, in overall, if sourcing considerations are kept high in priorities, iterations and delays cannot be avoided. This is ok, though, as long as the consequences are understood. High-level organizational policies determine the sourcing departments influence on development.

7.2.3 Deficiencies in product specifications

ORGANIZATIONAL/ HUMAN RELATED FACTORS

There have been deficiencies in *product* specifications because of five reasons, according to the interviewees;

- Some parts in the specifications cannot be stated accurately in advance. These parts get clearer as the development proceeds. These parts of specifications are said to be *evolving*.
- Some parts are intentionally left ambiguous for later updating for avoiding locking in specifications too early.
- Making specifications is hard work and making them as accurate as possible and plain for the designers would require much more focus.
- Some issues have been left vague because of wanting to avoid responsibility.
- Communication problems have occurred.

Some parts in specifications are said to be vague, conflicting or having non-transparent requirements, and other parts omitted completely. More responsibility in elaborating specifications and in some situations determination in decision making is longed for. Those parts of specification that are evolving should be clarified as soon as it is possible. There are deficiencies in this section, as designers state that it is either hard to communicate with people in charge of specifications (because of the complexity of the technical issues) or there is unwillingness to make decisions concerning some points in specifications. This can cause delays or harmful iterations when designers make their own conclusions about which direction to proceed. There have also been some deficiencies in determining cross-project specifications between profit centers.

Deficiencies in product specifications do not mean the same as changes in product specifications which, as mentioned earlier, are mainly the result of changes in the product offering and positioning. Product specifications are the result of considerations between upper management, product management, product development and sales. Specifications development and requirements engineering could be a topic of a completely separate master's thesis because of the broadness of these subjects, and they are not elaborated here further.

7.2.4 Cost monitoring

PROCESS CONTROL MECHANISMS

Deficiencies in cost monitoring cause problems practically in every project. Poor cost monitoring mostly leads to less harmful iterations which are not crucial for the success of the whole project. Still, there are some examples of projects which have run into real problems because the cost structure of the end product is at some point realized to be much greater than can be accepted.

The final costs of components and pieces of sheet metal are difficult to estimate *during* the product development project. The costs occurring when ordering parts for prototypes *during* PD projects are not the same as they will be when the product is under production and economies of scale are reached. This makes estimating the cost structure of the end product difficult.

Nevertheless, this area needs more emphasis and effort especially because cost monitoring is a process control mechanism, which means it is under management control and influence. Hence, the negative effects of poor cost monitoring can be more easily mitigated than the negative effects of some of the other iteration causing factors.

7.2.5 Scheduling

PROCESS CONTROL MECHANISMS

Deficiencies in scheduling have led to unrealistic schedules and insufficient schedule details, which have led to rework and delays. Deficiencies in scheduling also lead to difficulties in planning resource consumption.

In some cases unrealistic schedules can be the result of lack of references; how long does it take to develop platform software to the point where it can be released? But in most cases it is just a question of not putting an effort to make detailed schedules, or if they are done in the beginning of projects they are not updated during the execution of projects. To be able to sequence activities properly detailed schedule information is needed. Poor scheduling and poor communication can lead to asynchronous development; it especially involves other functions and global PD projects. Other functions may start their activities before they actually should. This leads to later iterations for other functions: manuals, service part lists, production etc.

There is a great need to emphasize detailed scheduling in projects since there have been uncertainties in different functions about the readiness of certain projects. In many occasions it is not enough to inform in what phase the project is, rather detailed information is needed about what different tasks have or have not been done and is the design information final or not. Other functions may intervene with the product development project too early, for not wanting to be the last ones to finish. This leads to iterations to PD since the information once fed to these functions needs to be refreshed by the designers later on.

HOW TO IMPROVE SCHEDULING

Timetables should be done from bottom to top. This means the project manager should make the project schedules in detail. By doing detailed schedules the project management gains arguments against predetermined sales release date that has been handed from top to bottom. The schedules should also be updated during the execution of projects, which is depicted in figure 7.1.

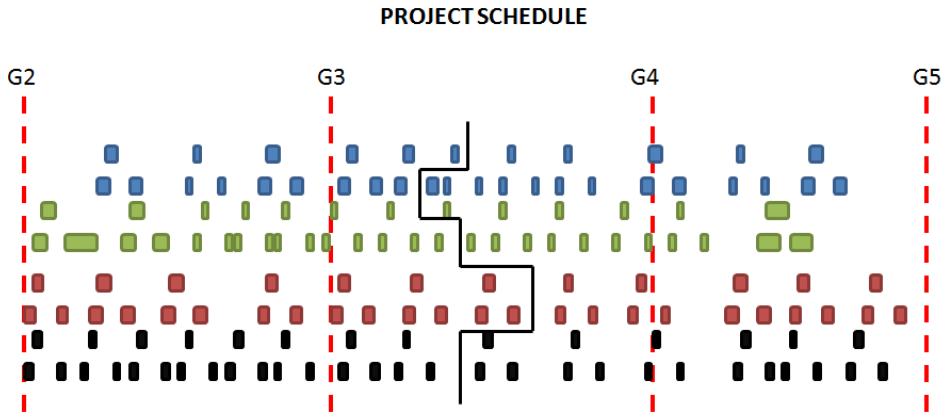


Figure 7.1 Detailed schedule which is updated on a continuous basis (Kantomaa, T.T.)

Detailed schedules also assist in keeping the *process rate* constant. Generally speaking, in any kind of constant process the best efficiency is achieved by keeping the process rate constant, whereas deviations are to be avoided. Many projects face schedule pressure towards the end of the project when it is realized how much work still needs to be done. On the other hand, in the beginning of the project there might be no sense of urgency. These are the result of not having detailed schedules and clear milestones, and lead to possible slacking-off in the early phases and schedule pressure in the later phases of a project.

Perhaps the schedules should also be varied. There would be three timetables: optimistic, realistic and worst case. The benefits of this kind of scheduling would be to acknowledge the fact that the project might not be ready according to the optimistic schedules and prepare stakeholders, especially sales, for delays.

Graphical Evaluation Review Technique (GERT) is an extension of PERT. It is one of the network-based tools proposed to deal with iterations under uncertain conditions. GERT is a procedure for stochastic network analysis in which activities have a probability of occurrence and their duration is treated as a random variable. [116] [117] [118]

In GERT the occurrence of major iterations leading to delays and rework in projects are evaluated in the beginning of a project. These iterations are given a probability and their effect in the case of realization is estimated, i.e., what kind of redesign path would they cause if they were to be realized. When the extra work and the probability of different occurrences are taken into consideration a more realistic schedule is achieved. If all the iterations are thought to happen with certainty the worst case scenario is reached. The benefits of this kind of scheduling would be to acknowledge the fact that the project might not be ready according to the optimistic schedules.

GERT has not gained compelling acceptance in the NPD management. Given that GERT requires an up-front definition of the process structure for its analysis, it may not be convenient in exploring process structure alternatives. That is why GERT should not perhaps be utilized in the company to its fullest extend taking into consideration every possible iteration cycle.

IN CASE OF ITERATION

Nevertheless, what would be reasonable is to make new/updated schedules whenever a need for a large scale redesign occurs, i.e. to schedule redesign paths. These redesign paths should be communicated to all stakeholders, including project sponsor and STECO, for making sure enough resources are committed for the project and that unrealistic time pressure is not directed to the project when large scale iterations do occur. There are examples where communication has been done poorly in case of realized redesign needs which has lead to insufficient resources being at the projects disposal. Making redesign paths and new schedules would also reveal delays caused to other design areas besides of those that are concerned with redesign, as well as to other functions, and by communicating these issues properly these resources could be released to other use. These measures leave enough time for development, hold back sales, increase the efficient use of resources, and hold back other functions from intervening with development too early. Figure 7.2 depicts a redesign path.

[116] Martinez, C., (2010) "A Coordination Mechanism for Lean Product Development", *Department of Industrial Engineering*, Texas Tech University, Lubbock, Texas

[117] Pritsker, A.A.B., and Happ, W.W., 1966, "GERT: Graphical Evaluation and Review Technique Part I Fundamentals", *The Journal of Industrial Engineering*, 17(5), 267-274

[118] Whitehouse, G.E., 1970, "GERT, A Useful Technique for Analyzing Reliability Problems", Technometrics, 12(1), 33-48

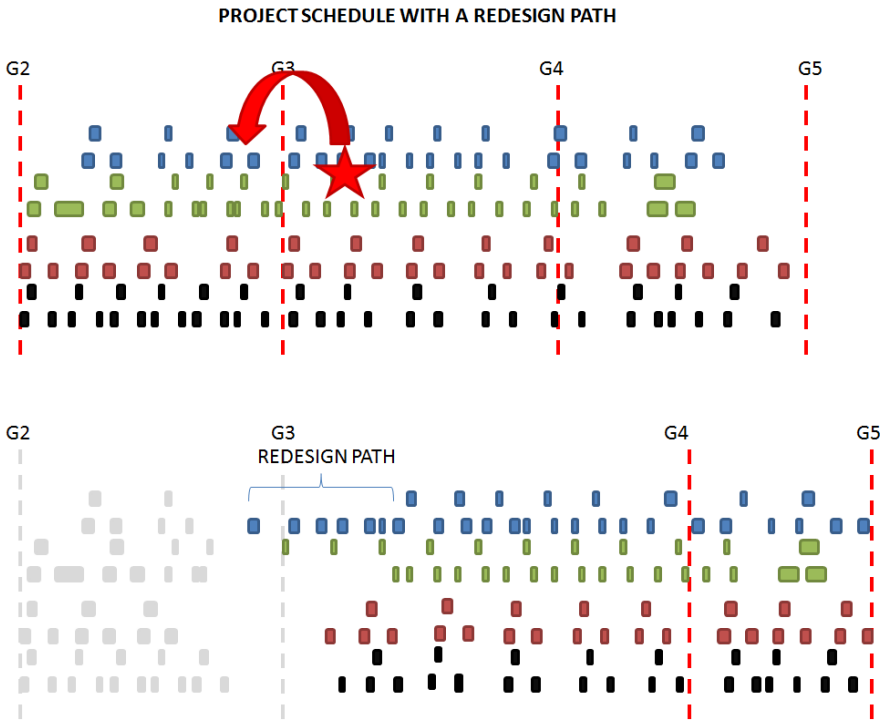


Figure 7.2 Iteration occurs and the project is rescheduled accordingly (Kantomaa, T.T.)

In case of large scale iteration it is important that:

- the amount of extra work is understood
- schedules are updated
- the need for extra work and new schedules are communicated to STECO and project sponsor to ensure sufficient amount of resources and to avoid excess schedule pressure
- the delays and new schedules are communicated to design areas excluded from the redesign path and to other functions to increase efficient resource usage (these resources may be used elsewhere)

These actions guarantee proper activity sequencing, relieve time pressure on the product development team which assures quality targets are met, make sure the project has the resources it needs for the completion of the project and increase efficient resource usage.

The amount of extra work is understood when project management, principal designers, and designers communicate with each other. If detailed schedules have been made in the beginning of a PD project, estimating the total needed extra work gets easier.

Once the amount of extra work needed is comprehended new schedules should be made. These schedules should incorporate the redesign path. The length of the redesign path is not the same as the delay for the whole project unless the redesign path is on the critical path.

The communication of these new schedules becomes very important, especially to sales, STECO and to project sponsor. Sales can put a lot of pressure on product development by promising sales release dates. By informing STECO and project sponsor it is assured that proper resources are guaranteed until the completion of the project and that resulting time pressure from unrealistic schedules is relieved.

After the project the progress of the project should be assessed to find out where the most deviations from the plans occurred and analyze the causes to those deviations. This would help in the future scheduling and process improvement efforts.

To summarize, project schedules should be scrutinized at three stages during a project:

1. Upfront, to adjust traditional estimates based on known or expected uncertainties from typical projects.
2. During the project to adjust to any changes that have occurred and to keep the schedules updated.
3. After the project to assess where the most deviations from the plans occurred and analyze the causes to these deviations.

7.2.6 Deficiencies in thinking over the design entity

Many problems and iterations in the company originate from deficiencies in taking into account all design related aspects. These problems can be allocated to deficiencies in the following areas: design responsibility, designer responsibility, communication and coordination.

Deficiencies in these areas have led to:

- not thinking thoroughly the application areas of the design and the following implications
- not thinking thoroughly the application of a sub-concept in different design configurations
- late arriving technical requirements
- lack of integration between design teams
- integration problems between module and cabinet designs
- not stating implicit assumptions have caused problems

Not thinking thoroughly the application areas of a design refers to, for example, not considering all related standards for a Marine application, which in turn results in late arriving technical requirements. Not thinking thoroughly the application of a sub-concept in different design configurations refers to, for example, not considering the cross-profit center application of a circuit board. Lack of integration has resulted in, for example, software problems encountered in hardware tests, which have not been resolved due to SW personnel not being present. The problem has been omitted simply by rebooting the HW, which leads to the SW problem being discovered later. The integration between module and cabinet designs has traditionally been a "no man's land". Not stating implicit assumptions from behalf of the people in charge of product specifications to the executing team has also led to some problems.

The roles and responsibilities have not always been clear when design details for designers have been generated. There have also been design responsibility issues; who looks after the design entity? Also simply forgetting to take all things into consideration has lead to deficiencies in design specifications. Designers also play a role in this; they have to take responsibility over the design and think thoroughly the application of their design.

Poor communication between designers has caused problems. There are examples where designers who are physically located next to each other have different assumptions on design details. At some point this information asymmetry is noticed leading to a harmful iteration. Designers not sharing information with

each other on a continuous basis will face problems later on. This applies to both designers in the same design area and to designers between design areas. This is something that occurs basically in every project.

Poor communication and coordination has also caused problems concerning specifications. There have been some deficiencies in determining cross-project specifications between profit centers. Product management communication between profit centers is crucial to avoid these in the future. If a profit center commences a project that has interdependencies with another profit center's project that is intended to be commenced much later the gathering of mutual requirements can be difficult. This is because people involved in the project which commences later in the other profit center can be occupied with other issues at the time when requirements should be gathered. There have also been situations where the people responsible for specifications have not been present in cross-profit center meetings where specifications have been dealt with.

The lack of communication between project managers/principal designers in different profit centers leads to deficiencies in design specifications. There are examples of designs that are to be utilized in several profit centers, but the cross-profit center aspect is not taken into account. The application of that design is not communicated properly. In many occasions designers receive information from project management or from a principal designer later than needed to avoid iteration.

Information exchange between project management and the project sponsor needs to be working properly for communicating the effect of changes in project pipeline for individual projects.

Profit centers are also unaware of other profit centers' past project experiences. Organizational learning suffers as a consequence. It has been said by the project managers that they do know what happens in other profit centers as long as they have personal contacts there. On these occasions project managers exchange information with other project managers or designers located at a different profit center through informal conversations. As soon as these contacts cease to exist the connection is lost.

Communication in the PD environment within and between projects needs to be enhanced. All fundamental communication links in the PD environment need to work properly; product management, project managers, project sponsor, principal designers and designers need to be connected the right way. More integration is needed. In big projects at least, it would be better for people to get to know each other well in the beginning of projects.

In the future more and more designing will be done in virtual teams; teams scattered around the globe. This means communication across project teams will gain more importance and managing projects becomes more challenging.

The stage-gate model, which is used to synchronize development efforts *within* a project, cannot counter the problems faced by increasing coordination needs resulting from increasing cross-project interdependencies and the distribution of development efforts (platform thinking and globally dispersed development). A program management effort has just recently begun in the company for addressing the need for more communication and coordination between projects.

INFORMATION ASYMMETRY

Actively informing the development team about the reasons for different decisions is important. By making sure people understand why some decisions have been made many misunderstandings can be avoided and

motivation can be improved. A clear information asymmetry exists between project managers and designers. Some designers have accused some sudden changes in past projects for wrong reasons. Designers become frustrated if one single actor is constantly thought to be the cause for sudden changes in projects. Also, it is good to communicate the state of the project to the development team. For example, if the technical concept is still under refinement and changes are expected to happen, informing this to the development team diminishes frustration when the changes eventually happen.

WELL DEFINED DEVELOPMENT PROCESS

A clearly articulated development process acts as a master plan which defines the **roles** of each of the players on the development team. This plan informs the members of the team when their contributions will be needed and with whom they will need to **exchange information**.

Network analysis (presented in chapter 3.3) could be used here for analysis purposes.

For the network analysis, not just task inputs and outputs, but also *information* inputs and outputs should be defined.

Network analysis would *reveal*:

- where most of the iteration occurs
- whether tasks are done in proper sequence
- whether an actor or a task is acting as an information collector or as an information distributor
- what nodes are especially important in the information flow

A well defined development process also contains natural milestones corresponding to the completion of each phase. The timing of these milestones anchors the **schedule** of the overall development project and makes the scheduling of projects easier and timetables more realistic.

A well defined development process is a benchmark for assessing the performance of an ongoing development effort. By comparing the actual events to the established process, a manager can identify possible problem areas for improvement. As such, a well defined development process would also help in **measuring** project success.

7.2.7 Concept feasibility

PROCESS CONTROL MECHANISMS

The concept development phase, as can be seen from the ABC-analysis, is one area that needs scrutinizing. The concept development phase is part of a slightly broader definition called the front-end process.

The front-end process generally contains many interrelated activities, ordered roughly as presented in figure 7.3. Rarely does the entire process proceed in purely sequential fashion, completing each activity before beginning the next. In practice, the front-end activities may be overlapped in time and *iteration* is often necessary.

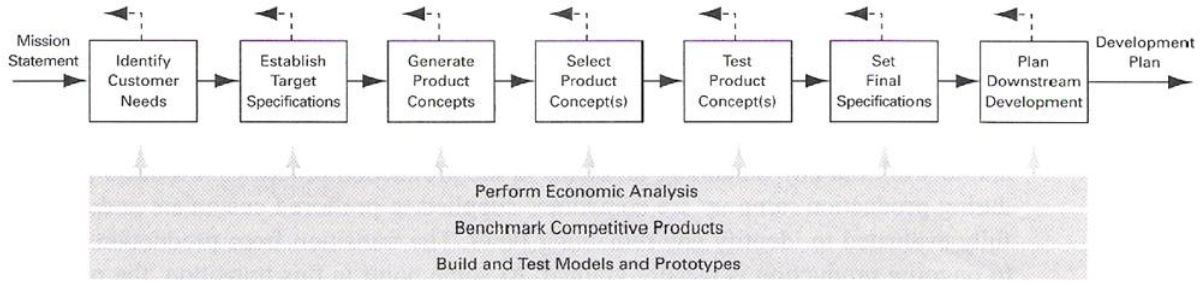


Figure 7.3 Concept development process[10]

The concept development process includes the following activities:

Identifying customer needs: The goal of this activity is to identify customer needs and to effectively communicate them to the development team. The output of this step is a set of customer need statements organized in a hierarchical list, with importance weightings for many or all of the needs.

Establishing target specifications: Specifications provide a precise description of what a product has to do. They are a translation of customer needs into technical terms.

Concept generation: The goal of concept generation is to thoroughly explore the space of product concepts that may address the customer needs. Concept generation includes a mix of external search, creative problem solving within the team, and systematic exploration of the various solution fragments the team generates.

Concept selection: Concept selection is the activity in which various product concepts are analyzed and sequentially eliminated to identify the most promising concept(s). The process usually requires several *iterations* and may initiate additional concept generation and refinement.

Concept testing: One or more concepts are then tested to verify that the customer needs have been met, assess the market potential for the product, and identify any shortcomings which must be remedied during further development.

Setting final specifications: The target specifications set earlier in the process are revisited after a concept has been selected and tested and the final specifications are set. Finalizing specifications is difficult because of trade-offs - inverse relationships between two specifications that are inherent in the selected product concept.

Project planning: At this stage a detailed development schedule is created, a strategy that minimizes the development time is formulated and the resources required to complete the project are identified.

Economic analysis: The team, often with a help of financial analyst, builds an economic model for the new product. This model is used to justify continuation of the overall development program and to resolve specific trade-offs among, for example, development costs and production costs.

Benchmarking competitive products: Understanding competitive products is critical for being able to successfully position new products. Benchmarking also provides a rich source of ideas for the product

design and production process design. Competitive *benchmarking* is performed in support of many of the front-end activities.

Modeling and prototyping: Every stage of the concept development process involves various forms of models and prototypes. These may include, among others: early "proof-of-concept" models, which help the development team to demonstrate feasibility; "form-only" models, which can be shown to customers to evaluate ergonomics and style; spreadsheet models of technical trade-offs; and experimental test models, which can be used to set design parameters for robust performance.

The phase of concept generation is very important; a good concept is sometimes poorly implemented in subsequent development phases but a poor concept can rarely be manipulated to achieve commercial success.

STRUCTURED APPROACHES REDUCE THE LIKELIHOOD OF COSTLY PROBLEMS

Table 7.2 shows common dysfunctions exhibited by development teams during concept generation according to theory, and compares them to practice.

Table 7.2 Common dysfunctions exhibited by development teams during concept generation (Kantomaa, T.T.)

FOUND IN THEORY	FOUND IN PRACTICE
Consideration of only one or two alternatives, often proposed by the most assertive members of the team.	YES
Involvement of only one or two people in the process, **resulting in lack of confidence and commitment by the rest of the team.**	NO
Ineffective integration of promising partial solutions.	YES
Failure to consider carefully the usefulness of concepts employed by other firms in related and unrelated products.	YES

A way to reduce the incidents of these problems is a *structured* method approach. This method encourages the gathering of information from many disparate information sources, by guiding the team in the thorough exploration of alternatives, and by providing a mechanism for integrating partial solutions.

A structured method approach is presented in the theory part of this thesis in chapter 2.2.1 "The activity of concept generation". More practical insights for improving concept feasibility are presented in chapter 7.3 "Framework for iterative development".

7.3 Framework for iterative development

The framework presented in this chapter aims at early iterative development to improve concept feasibility, early discovery of technical problems to diminish the amount of rework and repeated work, and shows how market uncertainties should affect the process structure. Through these measures reduced resource consumption and project lead time should be accomplished.

7.3.1 Project uncertainties

The previous chapter presented a generic concept development process. Project uncertainties, on the other hand, are distinct for every project and should be managed differently from project to project in order to diminish their impact. The overall **process structure should reflect project uncertainties.** Project uncertainties that cause schedule risks associated with product development projects can be controlled through iterative development. Figure 7.4 depicts uncertainties that should affect the process structure.

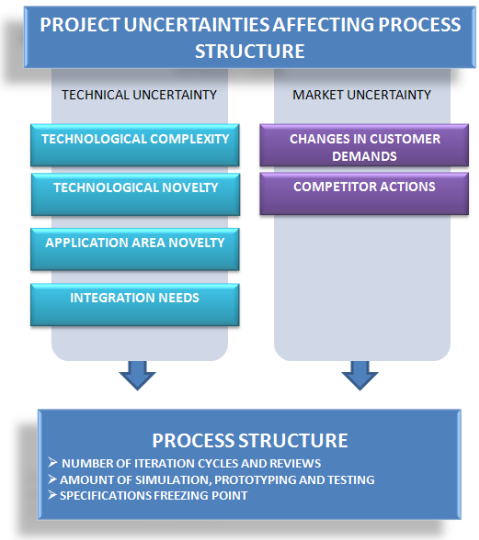


Figure 7.4 Project uncertainties (Kantomaa, T.T.)

In figure 7.4 those uncertainties, which should affect the *planned* process structure, are presented. There are other uncertainties as well, but in most cases they cannot be prepared to. These uncertainties include supply problems, resource availability, delays in other dependent projects, regulatory issues, unknown uncertainties etc.

Recognized market uncertainties should affect the *planned* process structure. Expected dynamics in the markets are countered by not freezing specifications too early and by trying to gain market feedback as soon as possible. In other instances changes in markets may become as a total surprise, in which case they do affect the *realized* process structure, but not the *planned* one.

When there is less uncertainties the process structure can be straightforward, but when uncertainties exist iterative development should be employed.

7.3.2 Iterative development model

The staged process is called the "waterfall" process by some practitioners because of its one way nature. The mindset in a staged process is straightforward and once a stage is complete, it is generally difficult to go back.

Waterfall model has the underlying assumptions:

- the requirements are known in advance of implementation
- the requirements have no unresolved, high-risk implications
- the requirements will not change very much during development
- the right architecture for implementing the requirements is well understood

If all the above assumptions are met, then waterfall model becomes applicable. Complete, consistent, testable specification in the beginning of development is not a feasible idea for many product development projects, though. Such technical risks may also exist, which need to be addressed already in the early phases of development, rather than in the integration and test phase to avoid problems later on. This chapter presents a framework for iterative development in the case of PD projects having some degree of technical risks along with market risks.

Iterative development has the following features:

- Concurrent rather than sequential determination of specifications, technical concept, and to some extend detailed design specifications
- Risk identification and determination of risk mitigation approach: simulation, prototyping, testing, market feedback
- Using risk considerations to determine the level of effort to be devoted to risk mitigation efforts
- Using risk considerations to determine the degree of detail in design specifications and the way to allocate resources

The risks faced in a specific development project are identified and prioritized. The risk identification is based on both past experience and recognized uncertainties. Uncertainties that are recognized help identify risks once probabilities of occurrence and potential impacts are assigned to those uncertainties. Basically there are two kinds of risks in this framework: technical risks and market risks. After risks have been identified they are then prioritized and each technical risk is assigned to a planned iteration cycle and to a design review. Technical risks can be mitigated by means of simulation, prototyping or testing (testing partial entities or components). Market risk stems from whether design specifications can meet customer needs; if not, a technically successful product could fail in the market. Market risks are addressed by not freezing specifications too early. This means specifications can be changed in case of necessity and this is communicated to the development team in order to avoid frustration in behalf of designers in case of specification changes. If market uncertainties cause high risk of specification changes it may be beneficiary to allocate resources to development efforts where the risk of specification changes is low.

Market risks can be managed by iterations that allow market feedback during the design process. This feedback includes elicitation of customer demands and information about competitor actions. High-priority risks are assigned to specific planned iteration cycles and reviews. Iteration cycles provide feedback which reduces risk in the next round. The result is a product development process that addresses a project's major development risks.

Figure 7.5 depicts risk identification and mitigation scheme.

Figure 7.5 Risk identification and mitigation scheme (Kantomaa, T.T.)

Using risk considerations to determine the level of effort to be devoted to risk mitigation efforts means that, there is a point after which risk has been minimized to an acceptable level. Spending significantly more time and resources than this is an overkill leading to late market entry and decreased market penetration. Spending significantly less time than this is an under kill, leading to premature development with significant delays due to unanticipated problems. In some instances while a risk may have a high profile it may not be feasible to mitigate it. In case of a technical risk this means that prototyping required to mitigate the risk would require so much effort that it is unfeasible to do so.

Using risk considerations when determining the degree of detail in design specifications refers to the fact that, making precise specifications in advance of development involves a high probability of locking in wrong specifications, while the risk of not making precise specifications is low. However, sometimes precise specifications are needed for the development to proceed. On these instances is must be communicated to the designers that the specifications may still change. This way frustration on behalf of designers in case of specification changes may be avoided.

Incremental commitment of resources to the development of the system, rather than fully committing resources to different development tasks from early on, is a great benefit that the iterative development model also brings. Development efforts can be focused on those parts of the design where the risk of design specification changes is low, instead of parts where the risk of design specification changes is high, or the resources can be used in another project or sub-project in case the risk of design specification changes is high. Figure 7.6 illustrates an iterative development process.

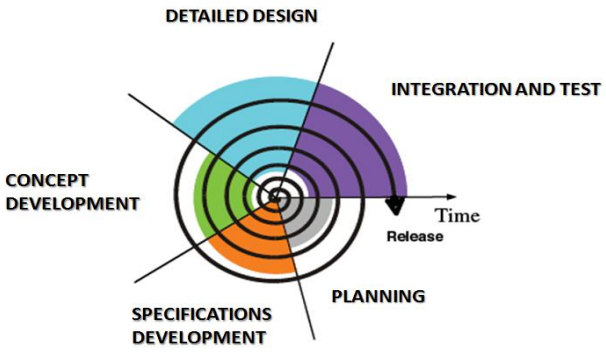


Figure 7.6 Iterative development process[96]

The iterative development process repeats regular steps, including planning, specifications development, concept development, detailed design, and integration and testing. The process is flexible; the actual number and span of loops can vary.

Figure 7.7 illustrates a staged process. The company uses this kind of formal process. Figure 7.7 also illustrates a functional development process on a rough level. This process description is closer to reality in the company. It can be seen that the functional process is actually iterative. The projects are, in almost every case, late in respect to the planned schedules in the company.

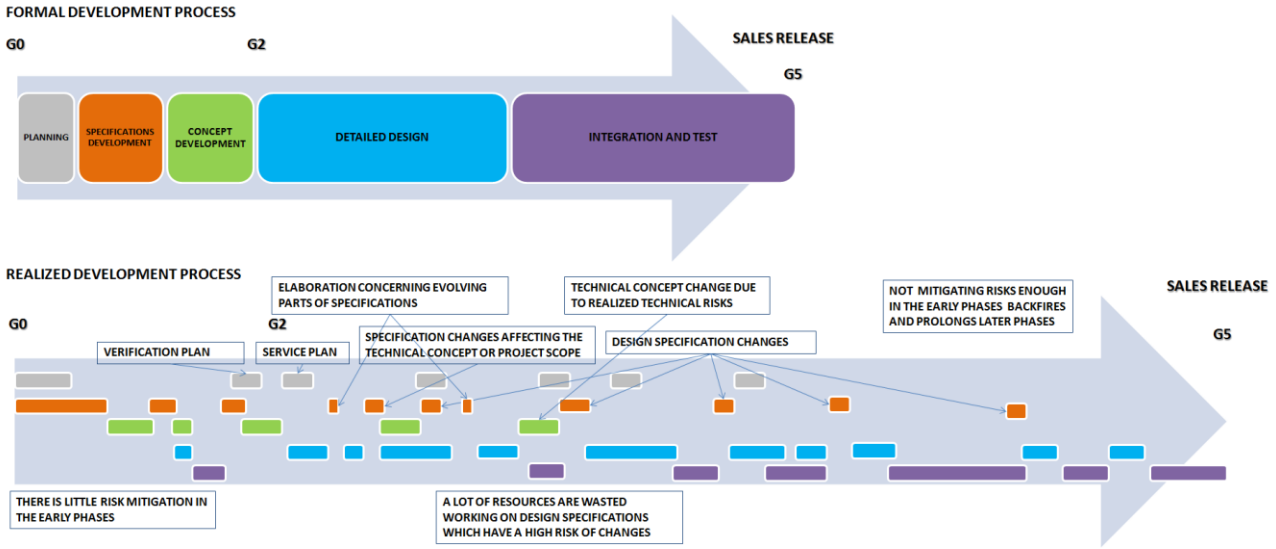


Figure 7.7 Formal and realized development process (Kantomaa, T.T.)

Figure 7.8 depicts the suggested iterative development process.

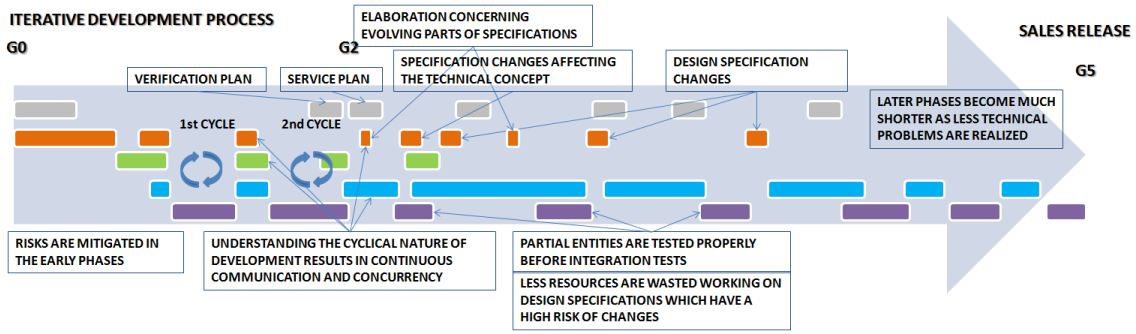


Figure 7.8 The suggested iterative development process (Kantomaa, T.T.)

The suggested iterative development process puts emphasis on early iterative development and risk mitigation. This may increase the time spent in early phases, but this effort is rewarded in later phases where, through risk mitigation, less problems are faced. Risks are identified and risk mitigation approach is determined. Risks can be mitigated through simulation, prototyping, testing or from gaining market feedback. Iteration cycles are then planned. After reaching the point where risk has been minimized to an acceptable level, the development team moves on to the next iteration cycle.

Specifications, technical concept, and to some extend detailed design specifications are determined in concurrent rather than sequential fashion. This results in faster response time and time compression. Risk considerations are used when determining the degree of detail in specifications. However, sometimes precise specifications are needed for the development to proceed. On these instances is must be communicated to the designers that the specifications may still change. This way frustration on behalf of designers in case of specification changes may be avoided. Evolving parts of specifications are to be resolved as soon as possible, which means the response time of product management should be short and responses should take on a clear stance on the issue at hand. It is easier to take a clear stance on an issue when it is realized that the specifications are not frozen and may still be changed later on.

On the other hand, if there is a high risk of specification changes it may be beneficiary to allocate resources to development efforts where the risk of design specification changes is low.

Table 7.3 compares phase-gate model to iterative model. It is noteworthy, that these two are different kinds of models. The phase-gate model is a business process model and the iterative model is a functional process model. Applying the iterative model on top of the business model would combine the benefits of them both.

Table 7.3 Comparison of phase-gate and iterative model (Kantomaa, T.T.)

	PHASE-GATE MODEL	ITERATIVE MODEL
EMPHASIS	FORMAL PROCESS FOR COMPLEX SYSTEMS	EARLY RISK MITIGATION
APPROACH	SEQUENTIAL PHASES	OVERLAPPING PHASES
ROLE OF SPECIFICATION	FIXED OVER SPECIFICATION PHASE	EVOLVING
CAPABILITY OF HANDLING UNCERTAINTY	LOW	HIGH
INFORMATION FLOWS	LONG FEEDBACK LOOPS	SHORT FEEDBACK LOOPS
MINDSET	STRAIGHTFORWARD	ITERATIVE

7.3.3 Iterative development model for addressing technical risks

Technology projects verify technologies for product development projects. However, PD projects are not always preceded by a technology project and those PD projects that are will still always have some technical uncertainties left to be resolved.

The risks faced in a specific development project should be identified and prioritized. There are often one or two high-probability risks that are likely to cause harmful iterations if they are not addressed. Once risks have been identified and prioritized, each risk is then assigned to a planned iteration cycle. The result is a product development process that addresses a project's major development risks early in the process.

Generally projects proceed to the detailed designing phase too fast and in a too straightforward way with concepts that have not been tested properly. Technical uncertainties having high probability of occurrence and severe consequences in case of occurrence should be tested in the beginning of PD projects before advancing too far in the detailed design process. If technical uncertainties can be solved in the beginning of development later iterations are avoided.

A lot more attention is required in the concept development phase. The detailed design phase should not proceed far until the risks associated with the technical concept have been dealt with. There will always be risks left for the actual prototyping (integration) phase, but the risk-level must be attempted to be minimized. When the management believes the risks associated with the concept have been diminished to an acceptable level, another iteration cycle is not considered to be productive or time and resources don't consent to any more iteration cycles, the project proceeds further to the detailed design phase. The opportunity that the iterative model gives is the possibility to learn early in the process, at a point in the PD process where design modifications are made much more easily than in later phases of the PD process. This will considerably speed up the learning process and by doing this, also speed up the development process itself.

Also smaller technical uncertainties should be resolved if feasible as the detailed design process proceeds. Especially technical novelties should be scrutinized. Technical novelties include novel/modified solutions from suppliers. In many instances a project enters prototyping phase and faces difficulties since there is a technical problem in the prototype that cannot be located (too many uncertainties are being tested at once) or has been located, but it takes time to resolve it and tests stand still. These situations can lead to futile changes to the design. There are examples of technical problems revealed in the prototyping phase where the cause of the problem has not been known. This has lead to multiple concurrent design changes for solving the problem, when only one design change might have resolved the problem. These minor technical problems will not result in changing the technical concept but they can still cause a lot of extra work. The later they are discovered the more extra work they cause. In some cases it would be wise to test more before integration tests and enter this phase with fewer uncertainties. The basic principle of agile product development is to test whenever it is feasible. Although agile product development is aimed at developing software, its basic principle is also applicable here. Some seasoned designers state this is exactly what they do – they try to resolve every individual problem they encounter to the fullest extend before advancing further to ease later development efforts. Of course, some uncertainties are always left to the integration phase, but the fewer uncertainties there are the easier the integration phase is to manage.

These iteration cycles don't necessarily solve the technical uncertainty completely – in many occasions it is enough to gather approximate information about a certain solution to aid in the development. The same applies with simulations. Simulations or virtual prototyping is an essential part of this framework. Simulations are used prior to prototyping for gathering approximate information about the behavior of a system. Complex simulations cannot be made for the whole system; it is still out of reach of the present simulation tools to simulate the behavior of a complex system with many interdependencies (for example, Electromagnetic compatibility of a system). Therefore, simulations can only be used to some extent. Nevertheless, the use of simulation tools is a good way to get estimates of the behavior of a system to set the approximate direction for development. Also, it can be used to compare alternatives against each other; even though the individual results of different alternatives are not accurate the best alternative can be spotted by comparing the alternatives against each other. This is the case for example, in thermal simulation where mechanical concepts are compared with each other. Accurate results can be obtained only by physical tests. Partial prototyping reveals some aspects, but the complete behavior of the system isn't uncovered until the integration tests commence. There are ongoing efforts in the company to get simulation tools of different design areas integrated and to make simulation more efficient.

The ways to mitigate technical risks before full integration include constructing partial prototypes, exploiting partially existing products in prototyping, using virtual prototyping (simulation) or by making component tests.

During the development technical uncertainties relating to the integration of the design are tested if feasible. Feasibility in this context means weighting the risks associated with a technical uncertainty against the efforts needed to resolve the technical uncertainty. Basically, it is said, everything can be tested; it is just a question of how accurate information can be gained from the tests and how much effort do the testing arrangements require. Nevertheless, the fewer uncertainties there are in the integration phase the more smoothly the integration will go.

RAPID PROTOTYPING

The purpose of rapid prototyping is to rapidly gain feedback about the feasibility of the technical concept. When incorporating new technology into the product technological uncertainties may exist. Rapid prototyping is used to detect and counter technical risks as soon as possible. The scheme here differs from piloting, since when entering normal piloting phase in PD projects the design is in many cases quite close to the final design and many areas in the design have already been very much detail designed. Rapid prototyping on the other hand utilizes prototypes that have not been detail designed anymore than needed to be able to construct a functional system or a part of it. The basic scheme in rapid prototyping could be to put aside for a while concurrent engineering principles, and concentrate solely on few main technical issues. This means the standpoints of production, sourcing, PE and service are thought less, and detailed design is applied only to the areas where needed to be able to conduct rapid prototyping. The goal is to rapidly encounter main technical uncertainties and mitigate the risks associated with them.

Rapid prototyping can also be used in situations when there are many concept options to gather information which will help in comparing the concept options between each other.

TESTING IN THE EARLY PHASES OF DEVELOPMENT IN PREVIOUS PROJECTS

In some occasions projects have not been able to use testing facilities and resources in the early phases of development because of other ongoing projects testing needs.

This might be a reinforcing loop. If early iterative development is not employed enough in the early phases technical problems are realized in later phases where they will consume more resources to be resolved than in the early phases. This again prevents these resources being used in other projects which are in their early phases.

HOW TO STREAMLINE TESTING?

Testing in the integration phase in the company is basically 90% prototype testing and 10% verification testing. Scheduling prototype testing is very challenging since it is hard to know beforehand what problems the tests will reveal. When problems turn up an iterative process between testing personnel and designers begins, and the length of this process is in many times unknown.

Partial entities should be tested as much as feasible to resolve problems before entering full integration tests. Much of the partial testing can be done in other places than in the actual high power testing facilities and by doing so the time needed to occupy high power testing facilities diminishes.

Testing specifications should be clear from the beginning of tests. It is not always so. Time is in many occasions wasted in situations where the equipment under test has been installed and the tests could be commenced, but there is no clear notion on what exactly should be tested.

The use of simulation tools is also an area which has promising future prospects. There are ongoing efforts in the company to get simulation tools of different design areas integrated and to get simulation more streamlined.

7.3.4 Product offering and positioning

EXTERNAL FACTORS

It has been said that one of the most harmful causes for iterations in the company are changes in the product offering and positioning. These changes can be caused by competitor actions, changing customer demands, upper management decisions, or late requirements from different profit centers.

This category would be easily deleted from the list of causes of iteration by making a guideline that after gate 2 the very general requirements and main features of the product under development cannot be changed. Only the evolving parts of specification and some smaller issues would be allowed to be specified during the detailed design phase. In some projects this has actually been the case; the specifications have been frozen after gate 2 and changes have not been allowed. Nevertheless, it should be kept in mind that the products under development have long life cycles so the performance and quality of products is more important than the speed of development, even though speed can give a competitive advantage.

Changes in product positioning can also result from poor initial product positioning, which is later corrected by management decisions. Late requirements arriving from different profit centers (especially SOLAR, WIND) can cause changes in the product attributes. Late requirements arriving from these profit centers are mainly due to lack of cumulative market knowledge and the fact that these profit centers operate in more dynamic markets compared to the better known and less dynamic industrial markets.

7.3.5 Iterative development model for addressing market risks

Market risk stems from whether design specifications can meet customer needs; if not, a technically successful product could fail in the market. Market risks are addressed by not freezing specifications too early. This means specifications can be changed in case of necessity and this is communicated to the development team in order to avoid frustration in behalf of designers in case of specification changes. Market risks can be managed by iterations that allow market feedback during the design process. This feedback includes elicitation of customer demands and information about competitor actions. High-priority risks are assigned to specific planned iteration cycles and reviews. Iteration cycles provide feedback which reduces risk in the next round. The result is a product development process that addresses a project's major development risks.

There are tradeoffs underlying the point of specification lock-in. The point of specification lock-in should be based on market dynamics, the flexibility of the design process, and available time. Also, because the competitiveness is based more on the performance of the products rather than the speed of development, the actions of competitors in the market should be monitored closely, and thus later specification lock-in should be allowed.

Delaying specification lock-in can improve the product's attractiveness and sales potential. Delaying specifications lock-in can also translate into an opportunity cost of lost profits. Delaying the finalization of

specifications would leave less time available for integration based on finalized specifications. Since in the integration phase product design details are optimized resulting in better quality or lower unit variable cost, delaying commitment results in opportunity costs.

INNOVATION TO COMPLETELY NEW MARKETS

When innovation is launched into completely new business the innovator must learn quickly about segments or customer needs and preferences. Iterative development may encourage and enable rapid experimentation and knowledge acquisition to innovate into this uncharted area. When market risk is included it is important to get feedback from the markets and customers early on; it's not just making sure the product works correctly in technical perspective, it's about making sure the right features are incorporated, and customer needs and wants are satisfied.

When satisfying customer needs and wants implicit assumptions may cause problems for new market entrants. Implicit assumptions are not stated explicitly when requirements are gathered from customers, instead customers make the assumption that the provider of some product naturally takes these things into consideration. This is especially common in software development. Some things are excluded in requirement statements because assumptions are made that these are naturally included in the software code, or when some things are thought to be so fundamental people don't see the need to elaborate on them. Another problem is caused by the IKIWISI syndrome. IKIWISI is an acronym from "I'll know it when I see it". It means customers are unable to specify their needs, until they get a hold of a physical product.

When a company enters new markets, it is important to address these issues rapidly. They are addressed the easiest way when the customer gets a hold of a physical product and is able to test it in real use environment. This is why the need for rapid piloting is highlighted when market risk is present. This is also in accordance of lean value adding methods, where you add touch points to the customers to test what they actually want. The purpose of rapid piloting is to rapidly gain feedback from the markets about how the features of the product satisfy customer needs and to detect any unmet needs. Pilots may also uncover implicit requirements within the company.

7.3.6 Set-based approach

Critical risks that cannot be mitigated through iterative development and testing in the concept development phase can be mitigated through set-based designing. Set-based approach is used to make the design flexible for changes by allowing a sub-concept to be changed to another without major design modifications if a risk associated with the preliminary sub-concept is realized. If a technical risk is realized in a later phase there are other design options that can be used through small changes to the design.

Set-based approach can also be used for the purpose of gaining a second source for the primary component. Although second source strategy is greatly emphasized in the company there are situations where two components are compared against each other in terms of quality, cost and applicability and an exclusive design choice is made, i.e. the other option is excluded from being a second source. This doesn't have to be the case. The design could be made flexible enough to include the unselected component as a second source, if wanted. In some cases this would require mechanical design to allow the use of either design option, and thus the design should be a bit more spacious.

In case of market risk set-based approach can be used for generating multiple concept options. After enough market feedback has been gained, one concept is chosen for further development.

8 CONCLUSIONS

This master's thesis started out with a very comprehensive theoretical background. The theoretical part starts with general product development theory, covers Lean Product Development (LPD), process modeling and then moves on to cover theory about task level iterations. The theoretical part presents tools for modeling and analyzing the detailed design process through a Design Structure Matrix (DSM) representation, network analysis or a matrix representation, and presents task overlapping principles. It also presents the concurrent engineering model and the spiral development model. Modeling the detailed design process even on a rough level would have required so much time and effort that it was out of the reach of this study. System dynamics approach was then chosen.

Main causes for iterations were identified in the empirical part of the study as well as many other areas needing improvement. Resolving some of the causes for iterations could be individual topics for other master's thesis, and thus solutions to some of the causes are only slightly covered in this thesis. The proposed framework addresses the main causes for iteration in the company. The main parts of the framework were verified by discussing them through with project managers and seasoned designers, and a document containing events from past projects was also made to give examples of situations where the framework would have proved to be useful in retrospective investigation.

The achievements of this thesis, besides of classifying the causes of iteration and presenting insights in how to manage them, was describing how product development projects in the company proceed from planning to sales release, documenting events from present and past projects, and pointing out other issues that need addressing.

This thesis was based on theoretical, conceptual and qualitative analysis. The next step would be to try to quantify different iteration related phenomena. This would allow, for example, cost-worth analysis; what improvements would be expected from different development efforts and what would these development efforts require. Key Performance Indicators (KPI's) would reveal the actual improvements. They could then be compared to the expected ones to analyze any inconsistencies.

9 SUMMARY

This chapter starts by reflecting back on the research questions made in the beginning of this study.

Main research question:

- **How to manage iterations in NPD to diminish their impact on project lead time and resource consumption?**

Auxiliary research questions:

- **What different kinds of iterations are there?**
  - **What causes them?**
  - **What are their features?**
  - **Which iterations are beneficial and which iterations are to be avoided?**
  - **What are their impacts on project lead time and resource consumption?**
- **Which tasks/phases should be done through iterations and which through the straight forward way?**

The main research question has been answered on most parts by the framework. The framework doesn't address all the causes for iterations comprehensively, because many of the causes for iterations and solutions to these issues could be individual topics for entire master's thesis. Nevertheless the main issues are addressed.

Auxiliary research questions "What causes them?" and "What are their features?" have been answered by dividing the causes of iteration into four classes (process control mechanisms, organizational/ human related factors, design related factors and external factors) and elaborating on all of them. Also their impacts on project lead time and resource consumption are addressed by classifying them according to an ABC-analysis. The question "Which iterations are beneficial and which iterations are to be avoided?" is answered by the iterative development model. It states that large scale iteration occurring late in the product development process is to be avoided whereas early short-cycled, feedback providing iteration is aimed for.

The question "Which tasks/phases should be done through iterations and which through the straight forward way?" has been answered by stating that the process structure should reflect project uncertainties. The more uncertainties a project has the more iterative the process should be. If major uncertainties are not included in the project the development can proceed more straightforwardly.

Topics for future master's thesis could include

- Specifications development and requirements engineering
- Improving communication in the PD environment
- Modeling the detailed development process

Perhaps the most interesting topic from above would be "Modeling the detailed development process", and not to include only task dependencies but also information flows (making a network analysis) in the model. Detailed development process including information flows would clarify the roles in the development process and it would point out which information sources should give input to which tasks. Nevertheless, this cannot be done in a single master's thesis because it would require so much effort and knowledge about the way products are developed, the technical issues and the nature of the product development environment in the company. It would simply lack the critical mass required for this kind of challenge. This is why this kind of master's thesis could be a part of a bigger development project which would aim at modeling the process, and would be supported by the process development team in the company.

FUTURE CHALLENGES

To be able to confront its future challenges the mindset in the company has to be proactive instead of reactive. That is why these last paragraphs are devoted for predicting future challenges.

It can be seen that there are three trends in the company's product development:

- Growth and increasing product scope
- Increasing focus on global processes
- A strong pressure on time to market

In the past product scope was smaller, the development environment was far less complex, and the knowledge resided in fewer hands. Growth and increasing product scope has brought problems to the company and forced it to change and develop its processes and information sharing. Product development and application engineering knowledge is increasingly been scattered around the globe. Product information is also being globalized, and information sharing in this respect is being harmonized. This doesn't mean that products are not localized anymore, though. Global product master data is collected after which localization can be commenced in respect to regional demands.

These before mentioned factors lead to increase in global processes and puts emphasis on communication and coordination procedures and information sharing systems. Shortcomings in these areas may mitigate knowledge accumulation, and propose problems in respect of quality and reliability.

Strong pressure on time to market requires a better understanding of the real specifications of a product from the perspective of the end-user already in the beginning of the development process. This puts an emphasis on requirements engineering and specifications development. Strong pressure on time to market also requires fast and efficient feedback methods to ensure product reliability. Fast and efficient feedback systems should be developed and used in the design process, especially in the case of strongly innovative products. Emphasis should not be solely on gaining feedback fast, but also with greater quality information. Early feedback can be enabled in the process by facilitating iterations with respect to performance, quality and reliability early in the design process. These iterations may come in the form of prototyping, virtual prototyping (simulating), and by confronting end users earlier with rapid piloting. Especially simulation has great potential in speeding up development by providing design information early in the design process.

10 REFERENCES

1. Kessler, Eric H., and Paul E. Bierly. 2000. "Internal vs. external learning in new product development: effects on speed, costs and competitive advantage" R&D Management 30, no. 3: 213.

2. Paul Trott (2005) Innovation management and New Product Development 3rd ed., Prentice Hall ISBN-10: 0-273-68643-7 ISBN-13: 978-0-273-68643-9

3. Nichols, G., (1990) Getting engineering changes under control Journal of Engineering Design 1 (1)

4. Peter B. Luh, Feng Liu, Bryan Moser (1997). Scheduling of design projects with uncertain number of iterations European Journal of Operational Research, 113, 575-592

5. IBM, Rational method composer – Rational unified process, version 7.1, 2006

6. Smith, R.P., Eppinger, S.D. (1997). A Predictive model of sequential iteration in engineering design. Management Science, 43(8), 1104-1120

7. Browning, T.R., Eppinger, S.D. (2002). Modeling impacts of process architecture on cost and schedule risk in product development. IEEE Transaction on Engineering Management, 49(4), 428-442

8. Eppinger, S.D., (2001) Innovation at the speed of information. Harvard Business Review, 79(1), 149-158

9. Kantomaa, T., Halonen, T., Developing the planning of product development projects in SAC-profit center (2011)

10. Ulrich, K.T., Eppinger, S.D. Product Design and Development (2008) 4th ed. McGraw-Hill ISBN: 978-007-125947-7

11. McGrath, M.E. Product Strategy for High-Technology Companies, New York, Irwin Professional Publishing

12. Wheelwright, S.C. and Clark, K.B. (1992) Revolutionizing Product Development: quantum leaps in speed, efficiency and quality, Free Press, New York

13. John Gruber Apple's constant iterations Macworld Apr2010, Vol. 27 Issue 4, p100-100 ISSN: 07418647

14. Abbie Griffin, The Effect of Project and Process Characteristics on Product Development Cycle, Time Journal of Marketing Research, Vol XXXIV (February 1997), 24-35

15. S. Minderhoud, P. Fraser, "Shifting paradigms of product development in fast and dynamic markets", Reliability Engineering & System Safety, 88 (2005) 127-135 doi: 10.1016/j.ress.2004.07.002

16. New Product Dynamics website http://www.europa.com/~preston/stage_gate.htm cited 9.1.2012 and 10.1.2012

17. Robert G. Cooper, Perspective: The Stage-gate® Idea-To-Launch Process – Update, What's new, and NexGen Systems, Product Innovation Management 2008;25;213-232

18. Jeffrey B. Schmidt, Kumar R. Sarangee, and Mitzi M.Montoya, "Exploring New Product Development Project Review Practices", J Prod Innov Manag 2009;26;520-535 ©PDMA

19. Dr. Robert G. Cooper, Dr. Scott J. Edgett, Dr. E.J Kleinschmidt, Optimizing the Stage-gate® process: What Best Practice Companies Are Doing, Part 1, Working paper No. 14 ©2006 The Product Development Institute

20. Product Development Institute, website http://www.prod-dev.com/stage-gate.php cited 9.1.2012

21. Dr. Robert G. Cooper, Doing it Right: Winning with New Products, working paper No. 10, The Product Development Institute Inc. ©2006

22. Cooper, R.G., "The invisible success factors in product innovation", Journal of Product Innovation Management, 16, 2, April 1999, 115-133 ©2006 The Product Development Institute

23. Don Reinertsen, Product Development – Best Practices Report (ISSN 1049-8400) ©2002 Management Roundtable Inc.

24. Dr. Robert G. Cooper, "The seven principles of the latest Stage-gate® method add up to a streamlined, new-product idea-to-launch process", MM March/April 2006

25. Brombacher AC, de Graef M., den Ouden E., Minderhoud S., Lu Y., "Reliability of technical systems and changes in organizational processes", Bedrijfkunde 2001;73(2):49-58 (in Dutch)

26. Naveem Gautam, Dissertation: A Design reuse based framework on Lean Product Development, doctoral thesis (philosophy), Wayne State University, Detroit, Michigan

27. Barry Jaruzelski, Richard Holman, and Omar Daud, "Next-Generation Product Development", Strategy+business, May 30, 2011 © Booz & Company Inc. (a global management and strategy firm)

28. Hoppmann J., Rebentisch E., Dombrowski U., Zahn T., "A Framework for organizing Lean Product Development", Engineering Management journal, Vol. 23, No. 1 March 2011

29. "Thinking Tools for Agile Software Development Leaders" © Poppendieck. LLC Last updated February 2, 2003

30. Cusumano, Michael A., and Kentaro Nobeoka, Thinking Beyond Lean: How Multi-Project Management is Transforming Product Development at Toyota and Other Companies, Free Press (1998)

31. Morgan, James M,. and Jeffery K. Liker, The Toyota Product Development System: Integrating People, Process, and Technology, Productivity Press (2006).

32. Thomke, Stefan, and Takahiro Fujimoto, "The Effect of 'Front-loading' Problem-Solving on Product Development Performance", Journal of Product Innovation Management, 17:2 (2000), pp. 128-142

33. Fiore, Clifford, Accelerated Product Development: Combining Lean and Six Sigma for Peak Performance, Productivity Press (2004).

34. Ballé, Freddy, and Michael Ballé, "Lean Development", Business Strategy Review, 16:3 (2005), pp. 17-22.

35. Sobek, Durward K., Allen C. Ward, and Jeffrey K. Liker, "Toyota's Principles of Set-Based Concurrent Engineering", Sloan Management Review, 40:2 (1999), pp. 67-83.

36. Clark, "The Power of Product Integrity" (1994) pp. 278

37. Brooks, "Mythical Man Month", (1995), pp. 255

38. Dr. Josef Oehmen, Dr. Eric Rebentisch, "Waste in Lean Product Development", LAI Paper Series "Lean Product Development for Practitioners", July 2010 © Massachusetts Institute of Technology, 2010

39. McManus, H. (2005) Product Development Value Stream Mapping (PDVSM) Manual, Cambridge, MA, Lean Advancement Initiative (LAI) at MIT

40. Oppenheim, B. W., (2004) Lean Product Development Flow. Systems Engineering, 7, 362-376

41. Morgan Swink, Michael Song, "Effects of marketing-manufacturing integration on new product development time and competitive advantage", Journal of Operations Management, 25 (2007) 203-217

42. Eisenhardt, K.M., Tabrizi, B.N., 1995. Accelerating adaptive processes: product innovation in the global computer industry. Administrative Science Quarterly 40, 84-110

43. Terwiesch, C., Loch, C.H., 1999. Measuring the effectiveness of overlapping development activities. Management Science 45 (4), 455-465

44. Jiyao Chen, Fariborz Damanpour, Richard R. Reilly, "Understanding antecedents of new product development speed: A meta-analysis, Journal of Operations Management, 28 (2010), pp. 17-33 doi:10.1016/j.jom.2009.07.001

45. Donald Gerwin, Nicholas G., Barrowman, "An evaluation of Research on Integrated Product Development", Management Science, Vol. 48, No. 7, July 2002 pp. 938-953

46. Ajay Menon, Jhinuk Chowdbury, Bryan A. Lukas, "Antecedents and outcomes of new product development speed; An interdisciplinary conceptual framework", Industrial Marketing Management, 31 (2002), pp. 317-328

47. Eric H. Kessler, and Alok K. Chakrabarti, "Speeding up the Pace of New Product Development", J Prod Innov Manag, 1999;16:231-247

48. Simon, H.A. Administrative Behaviour. New York: Free Press. 1976

49. Wheelwright, S.C., and Clark, K.B. Accelerating the design-build-test cycle for effective product development. In: Strategic Management of Technology and Innovation. R.A. Burgelman, M.A. Maidique and S.C. Wheelwright (eds.). Chicago: Irwin, 1996, pp. 859-868

50. Pattikawa, L.H., Verwaal, E., Commandeur, H.R., 2006. Understanding new product project performance. European Journal of Marketing 40 (11/12), 1178

51. Henard, D.H., Szymanski, D.M., 2001. Why some new products are more successful than others, Journal of Marketing Research, 38 (3), 362-373

52. Cooper, R.G., Edgett, S.J. and Kleinschmidt, E.J, Portfolio Management for new products. Reading, Mass; Addison-Wesley, 1998

53. Cooper, R.G., Winning at New Products: Accelerating the Process from Idea to Launch, 3rd edition. Reading, Mass: Perseus Books, 2001

54. Cooper, R.G. & Kleinschmidt, E.J, New Products: The key Factors in Success. Chicago: American Marketing Assoc., 1990, monograph

55. Crawford, C.M., "The hidden costs of accelerated product development", Journal of Product Innovation Management, vol. 9, no. 3, Sept. 1992, pp. 188-199

56. Cooper, R.G. & Kleinschmidt, E.J, "Major New Products: What distinguishes the winners in the chemical industry", Journal of Product Innovation Management 2, 10, March 1993, 90-111

57. Montoya-Weiss, M.M. & Calantone, R.J., "Determinants of new product performance: a review and meta-analysis", Journal of Product Innovation Management 11, 5, Nov. 1994, 397-417

58. Peters, T.J., Thriving on Chaos. New York; Harper & Row, 1988.

59. Dr Robert G Cooper, Dr. Scott J. Edgett, "Overcoming the Current Crunch in NPD resources", Working paper No. 17 ©2006 Product Development Institute

60. D. Nobelius, "Towards the sixth generation of R&D management", International Journal of Project Management, 22 (2004), pp. 369-375

61. Soumitra Dutta, Jean-Francois Manzoni (1999), Process Re-engineering, Organizational Change and Performance Improvement, McGraw-Hill

62. Asbjørn Rolstadås (1995). Business process modeling and reengineering. Performance Management: A Business Process Benchmarking Approach. p. 148-150

63. Auringer, A. (2009), Meeting the challenge: The 2009 Higher Education CIO agenda, Gartner, Stamford, CT.

64. Slack, N., Chambers, S., Harland, C., Harrison, A. and Johnson, R. (2004) Operations Management, 4th edition, Pitman, London

65. R. Mayer, M. Painter and M. Lingineni, Information Integration for Concurrent Engineering (IICE): toward a method for business constraint discovery (IDEF9), Knowledge Base Systems, INC (1995)

66. Gregor Zellner, A structured evaluation of business process improvement approaches, Business Process Management Journal, Vol. 17, No. 2, 2011, pp. 203-237

67. Sharp, A. and McDermott, P. (2001), Workflow modeling: Tools for process improvement and application development, Artech House, Boston, MA.

68. Braun, C., Wortmann, F., Hafner, M. and Winter, R. (2005) "Method construction – a core approach to organizational engineering", in Liebrock, L.M. (Ed.) Proceedings of the 2005 ACM Symposium on Applied Computing, SAC '05, Santa Fe, NM,ACM, New York, NY, pp. 1295-1299

69. Brinkkemper, S. (1996), "Method engineering: engineering of information systems development methods and tools", Journal of Information and Software Technology, Vol. 38, No. 4, pp. 275-280

70. Baumöl, U. (2005), "Strategic agility through situational method construction", in Reichwald, R. and Huff, A.S. (Eds), European Academy of Management (EURAM) Annual Conference, Munich.

71. Shawn T.Collins, Ali A. Yassine, and Stephen P.Borgatti, Evaluating Product Development using Network Analysis, Systems Engineering, DOI 10.1002/sys.20108

72. Elmaghraby, S.E.. (1995). Activity Nets: A guided tour through some recent developments. European Journal of Operations Research, 82(3), 383-408

73. Eppinger, S.D., Whitney, D.E., Smith, R.P., and Gebala, D.A. (1994) A Model-Based Method for Organizing Tasks in Product Development, Research in Engineering Design, 6: 1-13

74. Farhad Dilmaghani (2008). Critical Chain Project Management (CCPM) at Bosch Security Systems (CCTV) Eindhoven, Master's Thesis, University of Twente, School of Management and Governance.

75. Martinez, C., (2010) "A Coordination Mechanism for Lean Product Development", Department of Industrial Engineering, Texas Tech University, Lubbock, Texas

76. Pritsker, A.A.B., and Happ, W.W., 1966, "GERT: Graphical Evaluation and Review Technique Part I Fundamentals", The Journal of Industrial Engineering, 17(5), 267-274

77. Whitehouse, G.E., 1970, "GERT, A Useful Technique for Analyzing Reliability Problems", Technometrics, 12(1), 33-48

78. 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, 48(3), 292-306

79. Browning, T.R., 2002. "Process Integration Using the Design Structure Matrix", System Engineering, 5, 180-193

80. Juite Wang, Yung-I Lin, An overlapping process model to assess schedule risk for new product development Computers & Industrial Engineering 57 (2009) 460-474

81. Sander PC, Brombacher AC. "Analysis of quality information flows in the product creation process of high volume consumer products", J Prod Econom, 2000;67;37-52

82. Amabile, T. (1998). How to kill creativity, Harvard Business Review, Vol. 76, No. 5,pp. 76-87.

83. Collins, Shawn T., Bradley, Joe A., Yassine Ali A. Analyzing Product Development Task Networks to Examine Organizational Change, IEEE Transactions on Engineering Management Aug2010, Vol. 57 Issue 3, p513-525

84. Collins, Shawn T., Yassine, Ali A., Borgatti, Stephen P. (2008). Evaluating Product Development Systems using Network Analysis, Systems Engineering, DOI 10.1002/sys

85. Maqsood Sandhu and Petri Helo, A Network approach to project business analysis, Logistics systems research group, university of Vaasa, Vaasa, Finland

86. Wasserman, S., Faust, K. (1999), Social network analysis methods and applications, Cambridge University Press, Cambridge

87. Luh, D.B., Ko, Y.T., Ma, C.H. (2011): A Structural matrix-based modeling for designing product variety, Journal of Engineering Design, 22:1, 1-29

88. Browning, T.R., Sources of Schedule Risk in Complex System Development, Systems Engineering 3:129-142 ©1999 john Wiley & Sons, Inc. CCC 1098-1241/99/030129-14

89. Costa, Ramon, and Durward K. Sobek II, "Iteration in engineering design: inherent and unavoidable or product of choices made?" DETC2003/DTM-48662, ASME 2003 Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Chicago, Illinois, September 2-6, 2003

90. Dr Herrmann, J.W., "Managing Problem-Solving Iterations in Product Development", July 7th 2005, University of Maryland, USA

91. Shah, J. J., & Mäntylä, M. (1995). Parametric and feature-based CAD/CAM: Concepts, techniques, and applications. John Wiley and Sons, Inc.

92. Otto, H. E., (2001). From concepts to consistent object specifications: Translation of a domain-oriented feature framework into practice. Journal of Computer Science & Technology, 16 (3), pp. 208-230

93. Cooper, R.G., Edgett, S., 2008. Maximizing productivity in product innovation. Research-Technology Management, 51 (2), 47-58

94. Erdogmus, H. and Williams, L., 2003. The economics of software development by pair programmers. The Engineering Economist, 48 (4), pp. 283-319

95. Boehm, B., edited by Hansen, W.J. "Spiral Development: Experience, Principles, and Refinements", Spiral Development Workshop, February 9, 2000, Carnegie Mellon, Software Engineering Institute

96. Darian Unger& Steven Eppinger (2011) Improving product development process design: a method for managing information flows, risks and iterations Journal of Engineering design, 22:10, 689-699

97. Unger, D., 2003 Product Development process design: improving company response to market pressure, regulation, and changing customer needs. Thesis (PhD). Cambridge, Massachusetts Institute of Technology.

98. Krishnan, V., Eppinger, S.D., Whitney, D.E., "A Model-Based Framework to Overlap Product Development Activities", Management Science, Vol. 43, No. 4, April 1997

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

100. Lee, S.G., Ong, K.L. and Khoo, L.P. Control and monitoring of concurrent design tasks in a dynamic environment Concurrent engineering, Vol 12 number 1 March 2004

101. Smith, R.P., and Tjandra, P. (1998). Experimental Observation of Iteration in Engineering Design, Research in Engineering design, 10:107-117

102. Browning, T.R., S.D. Eppinger 2002 Modeling impacts of process architecture on cost and schedule risk in product development. IEEE Transactions on Engineering Management 49(4) 428-442

103. Browning, T.R., Modeling and analyzing cost, schedule, and performance in complex system product development, Ph.D. Thesis (TMP), Massachusetts Institute of Technology, Cambridge, MA, 1998.

104. Smith, R.P. and Eppinger, S.D., Deciding between sequential and parallel tasks in engineering design, Concurrent Engineering Res Appl 6 (1998), 15-25

105. Eckert, C. M. and Clarkson, P. J., 2003 "The Reality of Design Process Planning", ICED'03 Stockholm

106. Sterman, J. D., 2000. "Business Dynamics: Systems Thinking and Modeling for a Complex World." McGraw-Hill, Boston, Massachusetts, USA.

107. LE, H.N., WYNN, D.C. and CLARKSON, P.J. (2010) "Evaluating the impacts of iteration on PD processes by transforming task network models into system dynamics models" in ASME 2010 International Design Engineering Technical Conference & Computers and Information in Engineering Conference (IDETC/CIE 2010), Montreal, Quebec, Canada

108. Lyneis J.M., Ford D.N. (2007) "System dynamics applied to project management: A survey, assessment, and directions for future research." System Dynamics Review 23(2-3): 157-189

109. Joglekar N.R., Yassine A.A., Eppinger S.D., Whitney D.E. (2001) "Performance of coupled product development activities with a deadline." Management Science 47(12): 1605-1620

110. Krishnan V., Eppinger S.D., Whitney D.E. (1997) "A model-based framework to overlap product development activities." Management Science 43(4): 437-451

111. Roemer T.A., Ahmadi R., Wang R.H. (2000) "Time-cost trade-offs in overlapped product development." Operations Research 48(6): 858-865

112. Ford D.N., Sterman J.D. (2003) "The liar's club: Concealing rework in concurrent development." Concurrent Engineering-Research and Applications 11(3): 211-219

113. J. Lin, K. H. Chai, Y. S. Wong, and A. C. Brombacher, "A dynamic model for managing overlapped iterative product development," European Journal of Operational Research, vol. 185, no. 1, pp. 378–392, 2008

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

115. Graham, A.K. 2000. Beyond PM101: "Lessons for managing large development programs." Project Management Journal 31(4): 7–18.

116. Martinez, C., (2010) "A Coordination Mechanism for Lean Product Development", Department of Industrial Engineering, Texas Tech University, Lubbock, Texas

117. Pritsker, A.A.B., and Happ, W.W., 1966, "GERT: Graphical Evaluation and Review Technique Part I Fundamentals", The Journal of Industrial Engineering, 17(5), 267-274

118. Whitehouse, G.E., 1970, "GERT, A Useful Technique for Analyzing Reliability Problems", Technometrics, 12(1),33-48

APPENDIX A

Some of the topics and related questions of the interviews are presented in this appendix.

CAUSES OF ITERATIONS

What causes harmful iterations? What are the effects and occurrence frequencies of different kinds of iterations causes? How could the effects of harmful iterations be diminished?

DETAILED DESIGN PROCESS

What are the interdependencies between design areas? What inputs are needed in which point from other design areas in the design process? What kind of task sequence changes would help different design areas? In which ways could design flexibility be increased? What problems are there in the detailed design process?

CONCEPT GENERATION PHASE

How are concepts created? Who are involved in this process and is enough time and resources spent in this phase? Are there alternative concepts that are pursued?

SPECIFICATIONS

How are specifications for product development projects formed? What deficiencies are there in forming specifications? What causes specification changes? How does the collaboration between product development and product management work?

KNOWLEDGE ACCUMULATION

Are there problems in accumulating knowledge in the company? Do project managers and designers know the events of past projects and the events from projects in other profit centers? Are there examples of re-inventing the wheel in different PD projects?

COMMUNICATION

Are there communication problems in the company? What causes them? How could these problems be avoided in the future?

PROTOTYPING

Is more prototyping needed? What could and should be tested more in PD projects? Past examples of situations where testing has been neglected which has led to problems later on?

COLLABORATION BETWEEN PRODUCT DEVELOPMENT AND OTHER FUNCTIONS

LEAN PRINCIPLES

EVENTS FROM PAST PROJECTS

APPENDIX B

Questionnaire on how designers spend their work time

Circle your design area

main circuit mechanics electrical engineering printed circuit boards

Evaluate how much of your work time you spend on (altogether 100%).

1 not work related issues (coffee breaks, waiting in projects…)
_____(%)

2 concrete designing (besides of actual design work, includes participating in tests, working on test equipment, productizing components with suppliers etc.)
_____(%)

3 abstract designing – not actual design work, but work that enables actual design work (meetings, conversations with colleagues about work related issues etc.)
_____(%)

4 other kinds of work related issues, that are not actual design work nor are they actual design work enabling work. These are mainly done to aid other functions (sales, sourcing, logistics, production, manuals etc.) These include, for example:

- **BOM (configurable BOM to ERP system), entering material codes and the data behind them in to the ERP system, going through data sheets of components**
- **inbound logistics/purchase – making sure BOM's work**
- **Service Plan (service makes the service plan, but designers are also involved; designers have to choose spare parts and define them in kits. This also includes making maintenance tables (maintenance period tables))**
- **Technical data table**
- **manuals, brochures (to fairs etc.), side stickers**
- **Drive-Size sizing tools attributes (sales tool)**
- **software rating database**
- **working on model equipment (form only equipment for fairs, not actual prototypes)**
- **working on training material (training center, product managers own shows) – service managers and product managers make the material, but they need information from designers**
- **Participating in and preparing for gate reviews and design reviews**
- **supporting sales (sizing cases)**
- **that part of sourcing activities which could be done by some else than the designer itself**
- **SOMETHING ELSE, WHAT?**

_____(%)

What are the five most time consuming activities from part 4 (starting with the most time consuming)?

1
2
3
4
5

List 3 of the most time consuming activities from part 4, that are in most cases (not always) made iteratively. This means information needs to be updated because the initial work is not based on finalized design information.

1
2
3

How much of your work is something that could be performed by someone else, rather than a competent designer (%)? There could be a handyman in your team who would take care of such work, which does not need actual design competence.
_____(%)

How much of your total work time do you spend on sourcing activities?
_____(%)
How much of this could be performed by sourcing? Meaning that if there was a component engineer, who would have enough technical knowledge on components, how much could he lessen your time spent on sourcing activities (percentage of time spent on sourcing activities, value between 0-100%)?
_____(%)